CARNEGIE MELLON UNIVERSITY

School of Architecture
College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

Master of Science in Computational Design

TITLE:

Conversational Form-Generation: An Application of Interactive Genetic Algorithm to Architectural Design

AUTHOR:

Yingxiu Lu

ACCEPTED BY ADVISORY COMMITTEE:

[redacted]

Professor NAME	Principal Advisor	DATE

CARNEGIE MELLON UNIVERSITY

School of Architecture
College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

Master of Science in Computational Design

TITLE:

Conversational Form-Generation: An Application of Interactive Genetic Algorithm to Architectural Design

AUTHOR:

Yingxiu Lu

ACCEPTED BY ADVISORY COMMITTEE:

Professor NAME	Principal Advisor	DATE

Conversational Form-Generation: An Application of Interactive Genetic Algorithm to Architectural Design

by

Yingxiu Lu

Submitted to the School of Architecture
on May 9, 2018, in partial fulfillment of the
requirements for the degree of
Master of Science in Computational Design

Abstract

The design process can be regarded as a recursive conversation between representation and interaction, where designers recursively "generate representations of their interactions, and by interacting with representations simultaneously, designers generate relations with the representations of which they can then interact and repeat this process recursively." (Maturana and Varela, 1991)

This recursive conversation is quite typical and essential in the form-generation stage. This thesis shows how this recursive conversation can be boosted using Interactive Genetic Algorithm(IGA). With the assistance of IGA, the form-generation software can support an interactive conversation between designers and computational tools, by taking designers' evaluation and modification into consideration during computation process. Based on this interactivity of IGA, this thesis proposes an alternative working relationship between designers and computational tools, where they work in different scopes parallelly and share the outcome regularly. This alternative relationship offers an opportunity where designers could maximize their creativity, and computational tools could apply their computation capability to optimize the design idea.

A form-generation platform is implemented grounded on the proposed working relationship, for form study in the early stage of design process.

Thesis Supervisor: Daniel Cardoso Llach
Title: Assitant Professor

Acknowledgments

I'm genuinely indebted to the following individuals for their support and inspiration in the past two years.

To my thesis advisor, Daniel Cardoso Llach for encouraging me to delve into the discourse of the relationship between human and computational tools. His insightful vision and attentive guidance enabled me to preserve the exploration in this field.

To my cousin, Pingting Wei for all the unreserved criticism and inspiring discussions.

To my friends, Tianshu Cheng and Jiamin Sun for being with me whenever I feel lost or baffled.

To all my fellow MSCD students, especially to Yuqian Li and Weiwei Chi for the long conversations about design and future career.

To my parents for their support and understanding as always.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The design process could be regarded as a recursive conversation between representation and interaction.[1] In the design process, designers recursively "generate representations of their interactions, and by interacting with representations simultaneously, designers generate relations with the representations of which they can then interact and repeat this process recursively."[2]

This recursive conversation is quite typical in the form-generation stage. To be specific, an architect has his/her unique attitude towards space, form, material, culture, nature, and so forth. These understandings are his/her representation of the design. During the form-generation process, the architect interacts with his/her representation, by testing out different organization of components. When the architect evaluates the outcome, he/she will generate new understanding towards form, design, and architecture. This new understanding will lead to the modification of the outcome, which is the interaction based on the updated representation. This process will

[1] "....the observer-system can generate representations of its own interactions. When the system recursively interacts with these representations, it becomes an observer. The system can then recursively generate representations of these representations and interact with them, as when an observer thinks...." Hayles, N. Katherine. *How we became posthuman: Virtual bodies in cybernetics,literature, and informatics.* University of Chicago Press, 2008. p. 143

[2] Maturana, Humberto R., and Francisco J. Varela. *Autopoiesis and cognition: The realization of the living.* Vol. 42. Springer Science & Business Media, 1991. p. 14

last until the outcome meets specific standards or the architect is satisfied.

As architecture becomes increasingly complex, designers have to take more factors into account in the form-generation process, which raises the difficulty of the interaction in this recursive conversation. In the 1990s, the rise of computational tools helps designers to get over this difficulty.[3] The computation capability of these tools enables them to figure out an optimized form under multiple constraints much faster than humans.

Such benefit does require some trade-offs. The computational tools are experts in handling objective functions, which means that the constraints they take have to be explicitly defined. However, when we evaluate the form, we always have some expectations and requirements in mind which could not be explicitly defined, such as personal preference and aesthetics. This tension between objective function and subjective function has led to a series of studies on refining the performance of computation algorithms, from Genetic Algorithm (GA) to Interactive Genetic Algorithm (IGA).

This thesis offers another perspective on reconciling this tension, where we evaluate this tension regarding the co-working mode between computational tools and architects, rather than the performance of the tools.

1.2 Related Works

Interactive Genetic Algorithm(IGA) is a particular type of Genetic Algorithms(GAs), which are one of the best-known Evolutionary Algorithms invented for optimization in the 1960s. Different from other algorithms dealing with optimization problems, IGA takes the user's feedback into consideration during the computation process.

IGA turns out to be an effective method for designers and engineers to create a better computational design platforms, as it can keep refining the user request by inquiring for the user's preference. It could take both subjective functions from de-

[3]Fasoulaki, Eleftheria. *"Genetic Algorithms in Architecture: a Necessity or a Trend?".* 10th Generative Art International Conference, Milan, Italy. 2007.

signers and objective functions from computational tools into consideration, by integrating the designer as the fitness function or activating designer's access to adjusting parameters during computation process. Based on IGA, a series of design software has been developed based on IGA, serving as a handy tool for different design fields, such as graphic design, UI design, and fashion design.

1.2.1 Designers serving as fitness function

One typical way to get designers involved in the computation process is asking designers to serve as the fitness function. After a series of operations (crossover, mutation, and evaluation), a portion of the population will be presented to the designer. According to their preference, the designer assigns a score to each candidate or select some of the candidates to keep and eliminate the others for next generation. This approach fits with 2D graphics design tool quite well, since the features could be fully articulated through images.

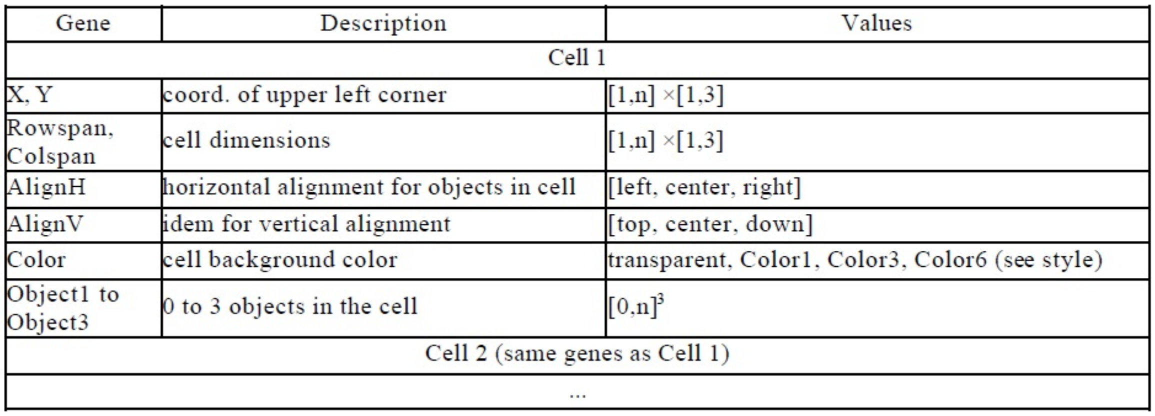
In early 2000, a group of researchers from Laboratory of Computer Science, Univerity of Tours developed a software generating style and layout of web pages with IGA.[4] They have fourteen genes which encoded the style of the web pages, including font, size, color, border, and so forth. The other six genes control the layout, regarding alignment, color, and location. After each iteration, twelve options with the highest scores will be shown to the user. The checked options will be kept for the next generation, and the others will be discarded and replaced with newly generated web pages.

In this interactive web page design software, the twenty genes cover millions of possible web page layout, and the interaction process of checking favorable options is intuitive and straightforward. However, the user fatigue is a severe drawback. In fact, the user fatigue problem is a significant challenge in most design software which takes users' preference as the fitness function in each iteration. Since genetic algorithm needs a considerable amount of iterations before grasping the target features, the

[4]Oliver, Antoine, Nicolas MonmarchAl, and Gilles Venturini. *Interactive Design of Web Sites with a Genetic Algorithm.* ICWI. 2002.

user has to repeat the evaluation process tediously to get a desirable result. In 2017, researchers from University of Isfahan improved the effectiveness and efficiency of this interaction process by adding candidate elimination algorithm to learn users' inclination.[5]

Figure 1-1: The encoded genes used to represent the features of a webpage.



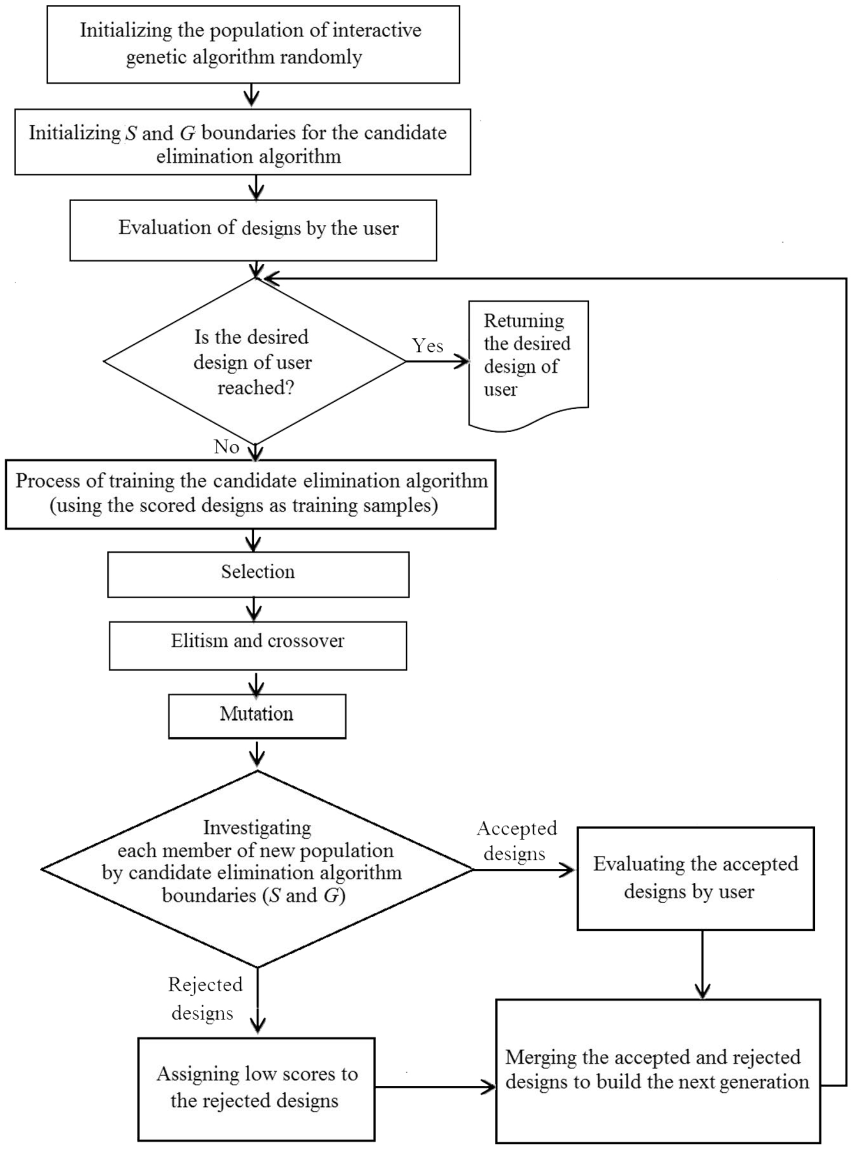
Gene	Description	Values
Cell 1		
X, Y	coord. of upper left corner	[1,n] ×[1,3]
Rowspan, Colspan	cell dimensions	[1,n] ×[1,3]
AlignH	horizontal alignment for objects in cell	[left, center, right]
AlignV	idem for vertical alignment	[top, center, down]
Color	cell background color	transparent, Color1, Color3, Color6 (see style)
Object1 to Object3	0 to 3 objects in the cell	$[0,n]^3$
Cell 2 (same genes as Cell 1)		
...		

The candidate elimination algorithm will take the scores given by the user as training samples to train the upper boundary and lower boundary which be used to classify if a solution from the population is qualified to be presented to the user in the next generation. By pre-filtering the options rather than showing all of them to the user, the proposed algorithm sharply reduces the average evaluation time and the number of generations in comparison with classic genetic algorithm and interactive genetic algorithm, which considerably decreases user's workload.

In both cases, users act as the fitness function and drive the direction of the computational design process. However, they do not have access to directly adjust key features, which could accelerate the computation process and reach the desirable outcome quickly.

[5]Darani, Zahra Sheikhi, and Marjan Kaedi. *Improving the interactive genetic algorithm for customer-centric product design by automatically scoring the unfavorable designs.* Human-centric Computing and Information Sciences 7.1 (2017): 38.

Figure 1-2: Flowchart of IGA with candidate elimination algorithm[5]

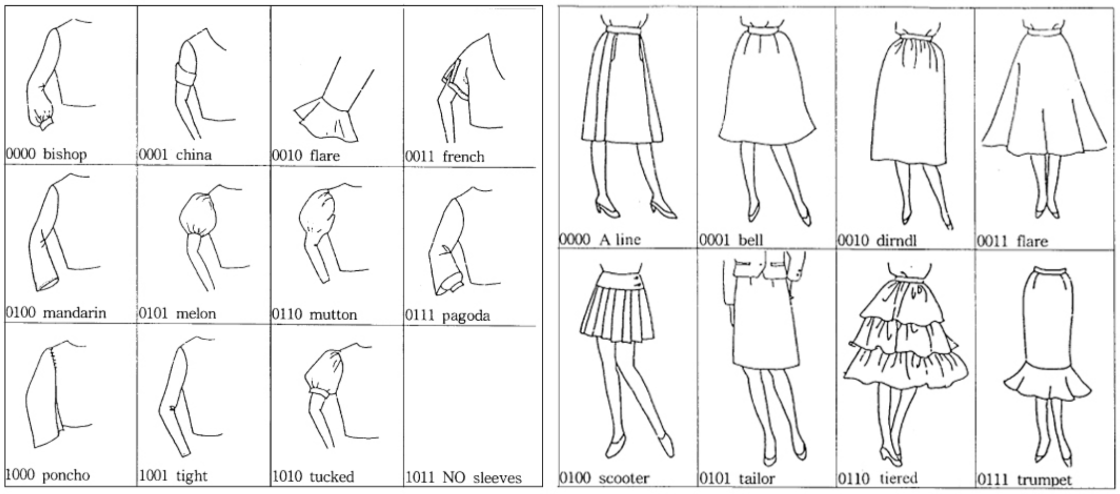

Initializing the population of interactive genetic algorithm randomly
Initializing S and G boundaries for the candidate elimination algorithm
Evaluation of designs by the user
Is the desired design of user reached?
Yes
Returning the desired design of user
No
Process of training the candidate elimination algorithm (using the scored designs as training samples)
Selection
Elitism and crossover
Mutation
Investigating each member of new population by candidate elimination algorithm boundaries (S and G)
Accepted designs
Evaluating the accepted designs by user
Rejected designs
Assigning low scores to the rejected designs
Merging the accepted and rejected designs to build the next generation

1.2.2 Adjusting parameters between loops

The other standard way to impart designer's opinion during the computation process is giving users the access to change pivotal parameters between iterations. Since parameters must be explicitly defined and matched to specific features, IGA design platforms, which allow users to control parameters, usually equipped with a component library and the parameter are related to the phenotypic description of a component. Since each candidate solution consists of components from the library, users could directly control some key features of the solution, by changing the value of parameters.

In the fashion design platform, developed by Hee-Su Kim and Sung-Bae Cho in 2000[6], a pool filling with hundreds of different kinds of 3D clothes detail models, such as neckline, collar, sleeve, and waistline served as the component library. Each candidate solution is a composition of selected detail models. Between iterations, the user could rate each solution and directly modify the model by changing parameters which control the length, width or style of the detail models.

Figure 1-3: Clothes detail library for fashion design platform[7]



A 3D modeling system, developed by Hiroaki Nishino and his team in 2001[7], demonstrated a similar approach. It also equipped with a 3D model library of abstract geometry shapes. The difference is that this 3D modeling system could handle more complex geometry operation, such as boolean and blend, where the former fashion design platform could only support a permutation of the selected components.

The IGA design platforms which give the user direct access to modify parame-

[6]Kim, Hee-Su, and Sung-Bae Cho. *Application of interactive genetic algorithm to fashion design.* Engineering applications of artificial intelligence 13.6 (2000): 635-644.

[7]Nishino, Hiroaki, et al. *A 3D modeling system for creative design.* Information Networking, 2001. Proceedings. 15th International Conference on. IEEE, 2001.

ters fits with non-professional design and fast prototyping for biology research.[8] It enables the user to control the direction of the computation process with parameters. However, this type of design platform is not suitable for professional design, since the built-in component library restricts the possibility of the outcome within a finite hypothesis set.

1.3 Hypothesis

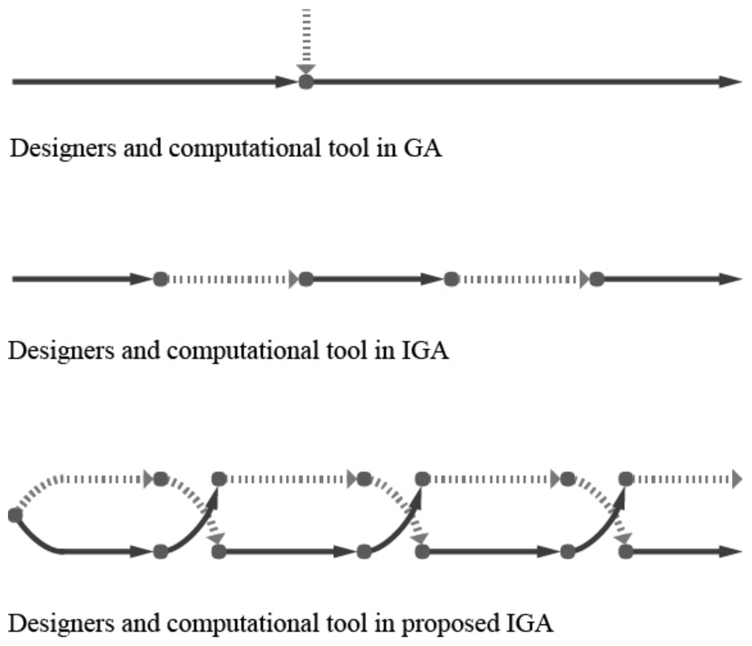
In the previous studies, design platforms using IGA introduce designers into the computation design process, where designers could express their preference and impact the evolving direction of the outcome. However, the tension between the designer and the computational tool remains still. Either designer could not access the candidate solution directly, which slows down the convergence, or designer can straightly modify solutions under a pre-defined frame at the expense of losing more poss possibilities. Evidently, designer and researcher understand the importance of balancing between subjective functions (from designers) and objective functions (from the tool) within the computational design process. The question is how.

The IGA is a powerful tool to achieve such balance and the design platforms discussed in the previous section indicate the progress made towards the balanced relationship between designers and tools. The limitation of these implementations is caused by the following two reseasons. First, the working scopes of both sides (designers and computation tools) overlap. Designers, who should have controlled the big picture with their creativity, had to evaluate and modify the details of the solution, while the computation tools, who should have given full play to the advantages of fast computation on details, had to guess the designers' preference. Second, both sides have to interrupt the other side's working process constantly, which was inefficient.

This thesis proposes an alternative working relationship between the two, where they work in different scopes parallelly and share the outcome regularly. This alterna-

[8] Min, Hyeun-Jeong, and Sung-Bae Cho. *Creative 3D designs using interactive genetic algorithm with structured directed graph.* Pacific Rim International Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2004.

Figure 1-4: A comparision of relationships between designers and GA/IGA.


Designers and computational tool in GA
Designers and computational tool in IGA
Designers and computational tool in proposed IGA

tive relationship offers an opportunity where designers could maximize their creativity and computation tools could apply their computation capability to optimize designers idea. Based on the proposed working relationship, a form-generation platform is implemented for form study in the early stage of design process.

1.4 Intended Contribution

This thesis focuses on exploring a new relationship between designers and computational tools, where both sides could support each other and promote each other through the design process, to raise the quality of the outcome and the efficiency of the process. Based on previous researches, a new workflow between designers and computational tools is suggested, implemented, and tested. The results would prove the feasibility and advantage of using this new workflow during the computational design process, as well as the possibility of applying it for typology research.

Chapter 2

Method

2.1 Interactive Genetic Algorithm(IGA)

Genetic Algorithms (GAs) are one of the best-known Evolutionary Algorithms.[1] GAs were invented for optimization and machine learning by John Holland in the 1960s.[2] After decades of development, GAs became the backbone in evolutionary computation area. In the 1990s, due to the increasing complexity in architecture, GAs were introduced into architecture design process as optimization tools and form-generation tools, because of their effectiveness in dealing with optimization problems under multiple constraints.[3]

As evolution-inspired algorithms, the structure of GAs is closely related to genetic variation and natural selection law. Typically, GAs consist of genetic operators (including crossover, mutation, and reproduction), the fitness function and the selection function.[4]

A particular type of GA, IGA dates back to the 1980s when Richard Dawkins demonstrated it to create a visualization tool to model an evolutionary system, called

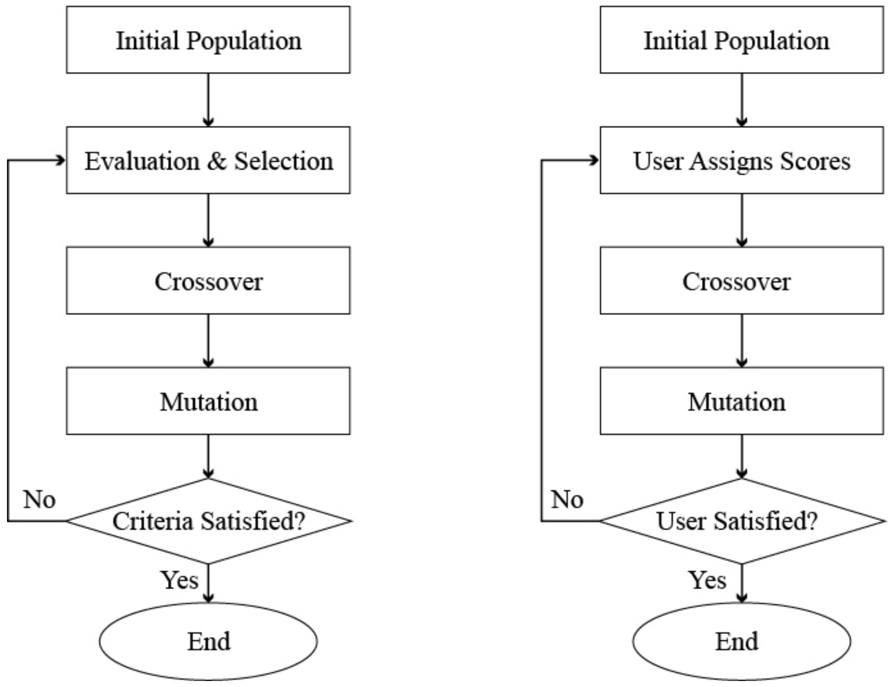
[1]Banzhaf, Wolfgang, et al. *Genetic programming: an introduction.* Vol. 1. San Francisco: Morgan Kaufmann, 1998.

[2]Mitchell, Melanie. *An introduction to genetic algorithms.* MIT press, 1998.

[3]Fasoulaki, Eleftheria. *"Genetic Algorithms in Architecture: a Necessity or a Trend?".* 10th Generative Art International Conference, Milan, Italy. 2007.

[4]Banzhaf, Wolfgang, et al. *Genetic programming: an introduction.* Vol. 1. San Francisco: Morgan Kaufmann, 1998.

Figure 2-1: GA and IGA algorithm flow



bimorphs.[5] The difference is that IGA could take subjective function into account, where Genetic Algorithm (GA) could only deal with objective function, by letting the designer acting as the fitness function to select the parents for next generation.[6] [7] In early 2000, IGA was introduced into the field of design, where a bunch of modeling tools were developed for fashion design[8] and plant morphology simulation.[9]

For both GA and IGA, each iteration consists of three main parts.[10]

- Crossover operation chooses two individuals from current populations as par-

[5]Farooq, Humera, and Muhummad Tariq Siddique. *A Comparative Study on User Interfaces of Interactive Genetic Algorithm.* Procedia Computer Science 32 (2014): 45-52.

[6]Darani, Zahra Sheikhi, and Marjan Kaedi. *Improving the interactive genetic algorithm for customer-centric product design by automatically scoring the unfavorable designs.* Human-centric Computing and Information Sciences 7.1 (2017): 38.

[7]Oliver, Antoine, Nicolas MonmarchÃČÄ¿, and Gilles Venturini. *Interactive Design of Web Sites with a Genetic Algorithm.* ICWI. 2002.

[8]Kim, Hee-Su, and Sung-Bae Cho. *Application of interactive genetic algorithm to fashion design.* Engineering applications of artificial intelligence 13.6 (2000): 635-644.

[9]Min, Hyeun-Jeong, and Sung-Bae Cho. *Creative 3D designs using interactive genetic algorithm with structured directed graph.* Pacific Rim International Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2004.

[10]Banzhaf, Wolfgang, et al. *Genetic programming: an introduction.* Vol. 1. San Francisco: Morgan Kaufmann, 1998.

ents. By swapping a part of one parent with a part of the other, the crossover operation will produce two children each time. For IGA, only individuals selected by the user are favored to be parents for next generation.

- Mutation operation can only be applied to one individual. Typically, after the crossover operation, each child produced from that operation will be passed into mutation operation with a low probability. Some bits of their genes will be flipped.

- Evaluation operation decides whether an individual could be kept in the population or not. In GA, the evaluation function has to be explicitly pre-defined by the user, while in IGA there is no explicit evaluation function. The users will act as the selection function. Based on their preference, users will pick out the individuals which will be kept.

To enhance the interaction part in the computational design process and retain a balanced relationship between designers and computation toools, three adjustments are made based on IGA.

2.1.1 More than acting as fitness function

More than acting as fitness function. Traditionally, the designer is involved in the IGA serving as a fitness function. The algorithm is in charge of computing explicit defined objective constraints, while the designer is responsible for picking the parents for next generation. This cooperation between computational tools and designers could balance the subjective and objective constraints to some extent. Whereas this implementation limits designers access to the computational process, by confining designers' role within passive choosing, rather than active interacting.

For the sake of designers' active involvement, during the computation process of IGA, designers should not only act as the fitness function but have access to modify and revise the form.

2.1.2 Working in different scopes

Designer's creativity enables the designer to control the big picture of the outcome, while computational tool's computation capability empowers it to take all the details into account and figure out an optimized solution. Therefore, the designer and the computational tool should work in different scopes to make the computation process more efficient. Designers should work in higher level to control the evolving direction of the outcome. Whereas the computational tool should focus on optimization of details, to support higher level's decision.

2.1.3 Working in parallel

Working in parallel. To maximize the creativity of designers and computation capability of IGA, this thesis suggests a new relationship between designers and IGA in the form-generation process, where the two work in parallel and support each other. Designers and IGA take the same initial models and develop them separately. During the computation process, the two swap outcomes regularly. After each swap, they carry on the computation process based on the other's outcome.

2.2 Recording the think-out-loud

As an interaction-oriented design platform, the evaluation of this platform should emphasize the interaction process between the designer and the computational tool. Besides the final 3D model, we should also assess how this interactive conversation affects designers, regarding identifying design problems, polishing design strategies, evaluating their own solutions, and so forth.

Donald Schőn has conducted a lot of observations and experiments about designer's design process[11]. His observation and documentation method could be applied to demonstrate the interaction between the designer and the design platform.

[11]Schőn, Donald A. *"Designing as reflective conversation with the materials of a design situation."* Knowledge-based systems 5.1 (1992): 3-14.

In his documentation, he first described how students elaborated their design process, including what design decisions they made and why they made such decisions. Then, he analyzed the connection between their motivations and actions.

In this thesis, similar documentation approach will be applied, as the designer using this design platform. The major difference is that the recorder is the designer, rather than a third party. The designers are supposed to document their observation of in consulting mode, especially the unexpected result, and their design strategies, if changed.

This documentation will help designers to trace back the development of their design strategy and reflect on the decisions they made, after they finished the design process.

Chapter 3

Experiment

3.1 Prototype - a form generation platform

Based on the proposed working pattern between designers and computational tools, a form generation software is developed based on IGA. It offers designers a interactive design platform to explore forms with given components defined constriants in the early stage of design process. This prototype aims at demonstrating the feasibility and advantage of applying the hypothesis in the design process with a modification on standard IGA algorithm.

This interactive design platform could be applied in a residential building design scenario. Provided with different room type, this platform will generate a form, which is a result of cooperation between designer and computational tool. The designer could control the skeleton of the form, while the IGA will help designer optimize the form in higher resolution by evaluation possible permutations of given room types and their accessibility to the public spaces.

3.2 Algorithm Flow

After the designer creates an initial form using provided parameters, IGA will help designers to optimize this form under two constriants. First, each component should have access to the public space. Second, the relative proportion of three given types of component should apporaching the desired relative proportion controlled by the user. During the computation process, the designer could navigate through the model and make modification on the form. IGA will take designer's adjustment into account in the following iterations. The following is the pseudocode of IGA in the optimization stage of form generation.

```
Data: Initial cube matrix M and default relative proportion R.
Result: Optimized cube matrix M'.
Generate a group of random packing solution with given components based
  on M.;
while no stop request or solutions haven't converged do
    if user updated relative proportion then
        R ← updated relative proportion;
    end
    for model ∈ population do
        score ← evaluate model based on overall accessibility of each
          individual component to public space & difference between actual
          relative proportion and desired relative proportion;
    end
    delete model with low score;
    for each pair of models ∈ population do
        produce children;
        mutate children with Russian roulette;
        add children into population;
    end
end
M' ← model with highest score;
return M' ;
```

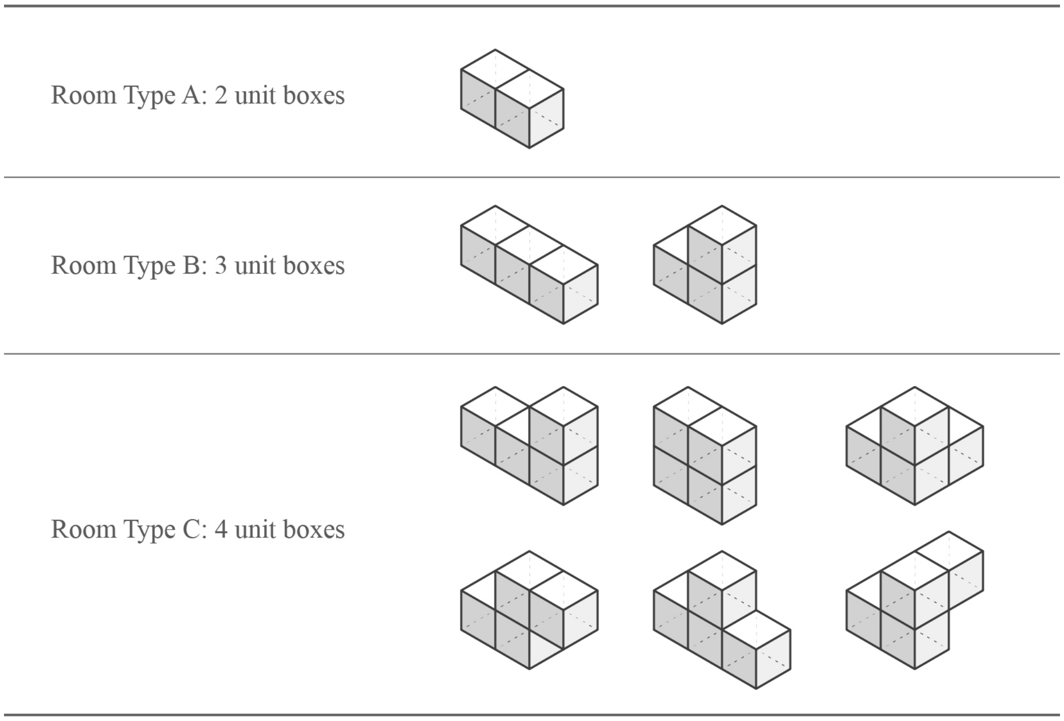
Algorithm 1: IGA for form generation

3.3 Implementation

In this design platform, the designer and the computational tool concentrate on different levels. Designers focus on a higher level, where they could control the relative proportion of all room types and modify the form boundary. On the other hand, the computational tool aims to figure out an optimized permutation of given room type which satisfies the requirement of accessibility and relative proportion. A conversation between designers and computational tools is generated, where the latter updates its selection function based on former's updated input and the former keeps adapting their design strategy in response to the solution offered by the latter.

3.3.1 Components

Figure 3-1: Three room types as basic components.


Room Type A: 2 unit boxes
Room Type B: 3 unit boxes
Room Type C: 4 unit boxes

This interactive design platform provides users three room types, containing different number of unit boxes. Figure 3-1 shows all possible configuration of each room type. In residential building design scenario, each configuration represents a particular layout of a living unit. The IGA will take these room units as basic components and pack them into the shape given by the user, while taking the accessibility of each unit and the relative proportion of them into account.

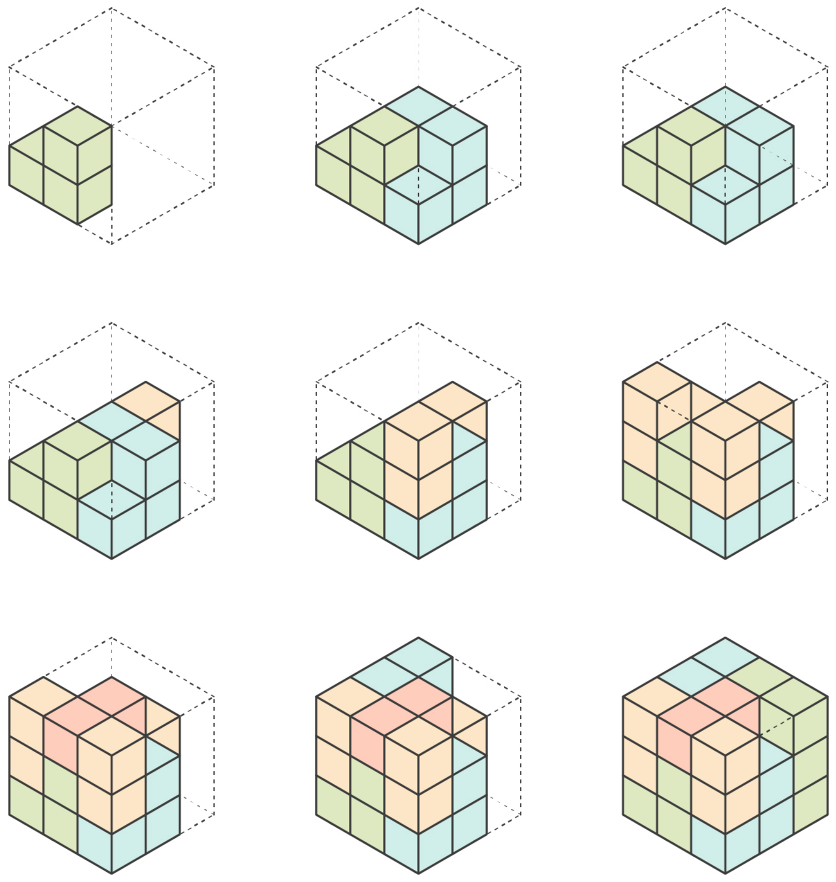
3.3.2 Initialize random packing solution

The user could create shapes by adding or eliminating cubes. After receiving the latest shape created by the user, the IGA will generate a population of random packing solution of the room units. For each cube in the given shape, the program will loop through all 3x3x3 spots. If the spot is not occupied, the program will randomly choose a room type and position the room unit into the available spots. Otherwise, the program will skip the current spot, leaving it empty, and check next spot's availablity.

```
Data: Cube matrix M.
Result: A packing solution matrix M.
for each cube $\in$ M do
    if cube is activated by the user then
        for each unoccupied spot $\in$ cube do
            random choose a room type $T$;
            search for available spots which could fit the room unit;
            if there exist available spots then
                mark all available spots as occupied;
            end
        end
    end
end
return M ;
```

Algorithm 2: Random packing for 3x3 cube

Figure 3-2: Process of packing solution generation.



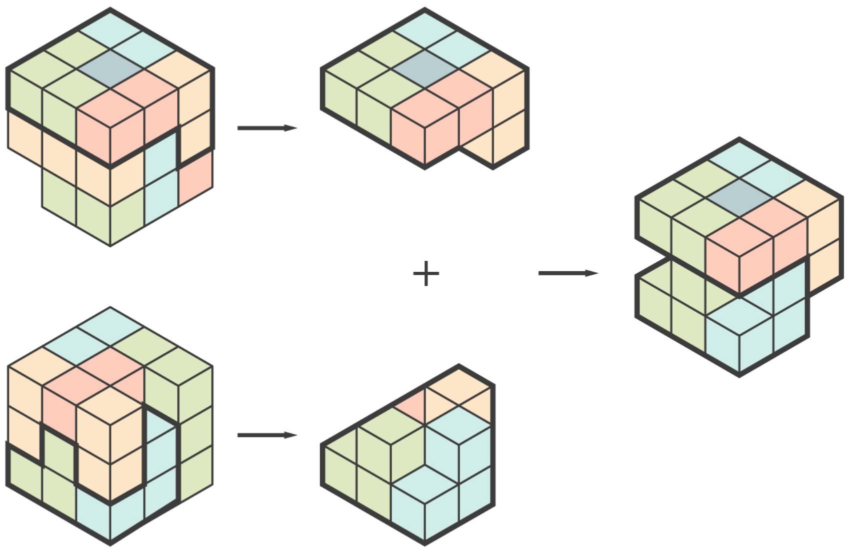
3.3.3 Crossover

In the crossover operation, for each pair of individuals in the population, the program will generate a child which inherit half "genes" from each parent. Contrary to the conventional crossover operation in IGA, it is not reasonable to simply split each parent in the middle and assemble the two halves together, since it would break the components in the middle into pieces and change the "genes". Thus, the child could not inherit as many "genes" from both sides.

3.3.4 Mutation

For each generated child, it might take the mutation operation with considerably low probability. The mutation operation contains insert, delete, break, and connect,

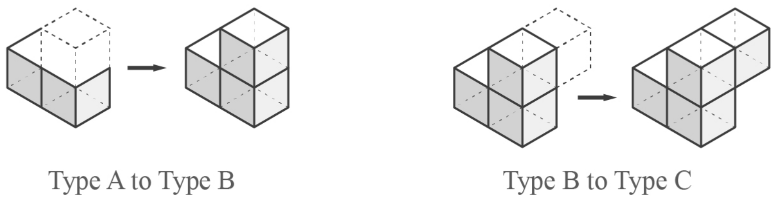
Figure 3-3: Example of crossover operation.



which aims at increasing variance and diversity among the children.

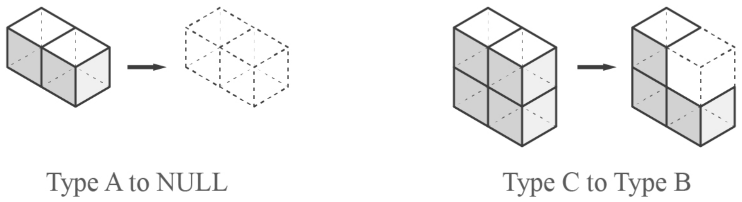
- Insert operation. The insert operation can only be executed between empty spot and empty spot, between empty spot and Type A unit, or between empty spot and TypeB unit. For each empty spot in the cube, if one of its neighbors satisfies the requirement, the empty spot will be unioned with its neighbor and the neighbor will switch from empty spot to Type A unit, Type A unit to Type B unit, or Type B unit to Type C unit.

Figure 3-4: Example of insert operation.



- Delet operation.The delete operation, removing one of the boxes, can be executed among all type of room units. After the delete operation, Type A unit will become two empty spots, because room unit with only one box is not defined as valid room unit. Type B unit will become Type A unit. Type C unit will become Type B unit.

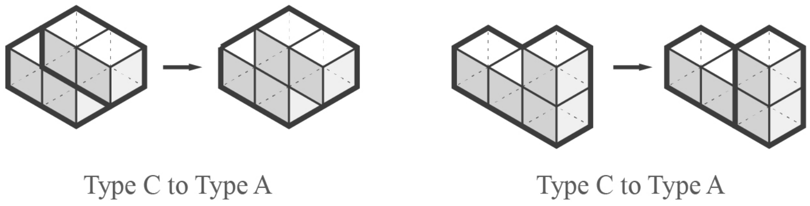
Figure 3-5: Example of delete operation.



- Break operation.The break operation can only be executed on Type C unit, which consists of 4 boxes. After the break operation, the Type C unit will be splited into two Type A units.

- Connect operation.The connect operation can only be executed between two Type A units. The connect operation will union these two Type A units and produce a Type C unit.

Figure 3-6: Example of break and connect operation.

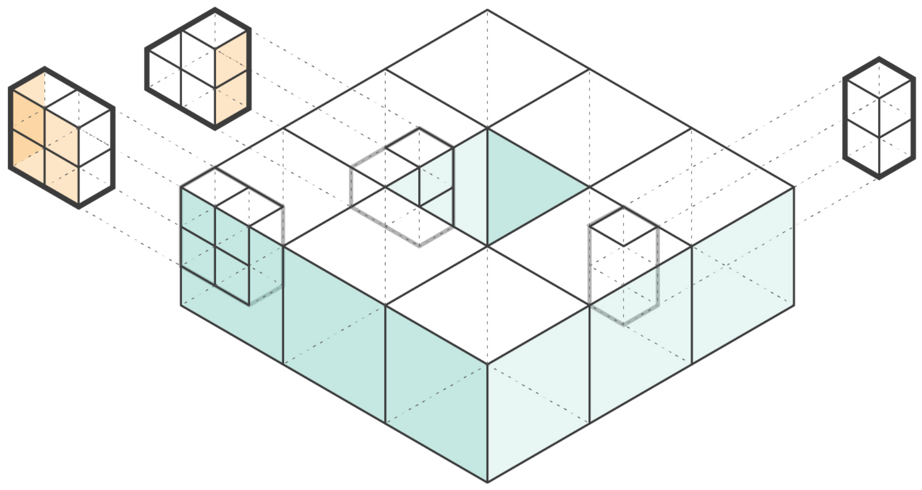


3.3.5 Select function

The selection function will examine each packing solution from two aspects. First, the relative proportion of three room types should be as close as the desired relative proportion set by the user. Second, each box in each room unit should have one or two faces facing public space, courtyard or street. The selection function will give a score to each candidate solution and eliminate solutions with low scores.

Figure 3-7 shows an example, where the Type C unit will get a high reward, since each box in it has at least one face facing the street. Type B unit will get a low reward, since two of the boxes have one face towards the courtyard. Type A unit will get penalty, as none of the boxes have access to courtyard or street.

Figure 3-7: Evaluation of accessibility to public space.



3.3.6 User Interface

The user could interact with IGA with two different approaches. The first approach is direct and intuitive. By changing the relative proportion of room types in the control panel and modifying the shape in the modeling window, the user could adjust the constraints and parameters of the evaluation function in IGA. The second approach is more subtle but vital. In the modeling window, the user could switch between editing mode and consulting mode. The editing mode allows the user to focus on the creation of shapes, while the consulting mode offers the user opportunities to evaluate the current solution and design strategy by inspecting the design problem from the perspective of defined constraints.

Figure 3-8: User interface: control panel (left) and modeling window (right).

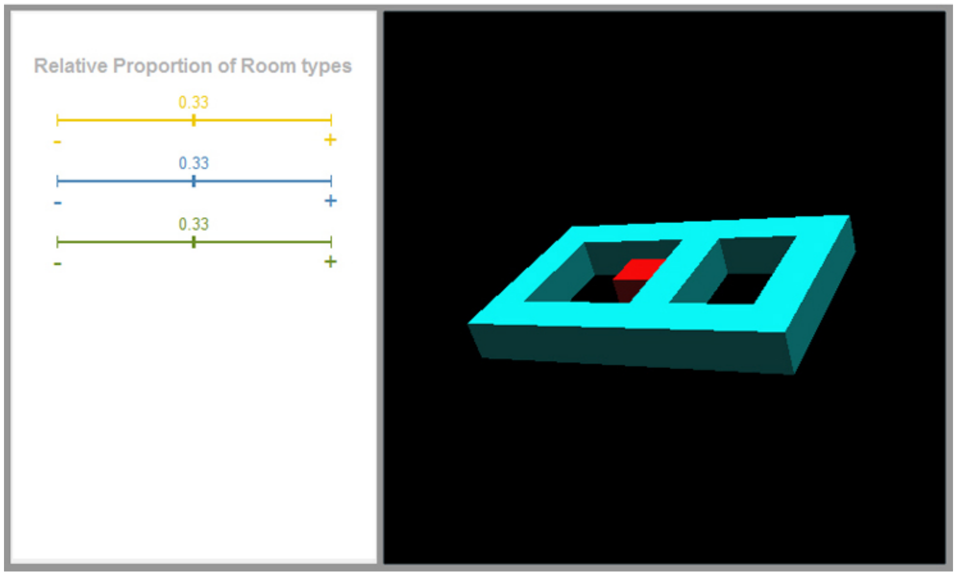
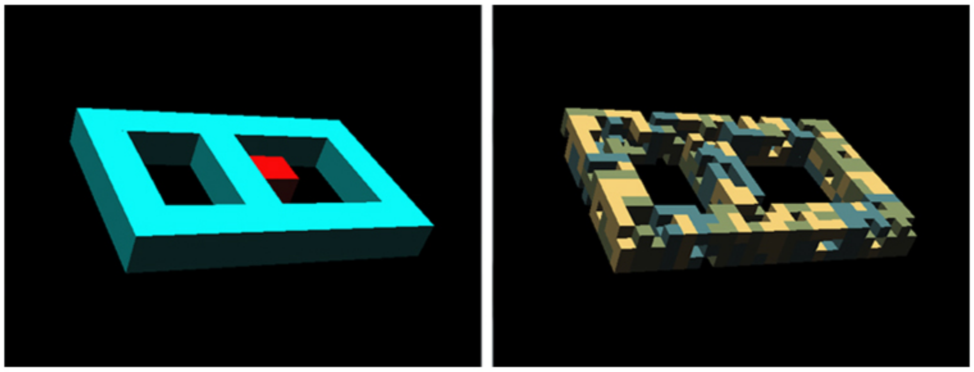


Figure 3-9: Editing mode (left) and consulting mode (right).



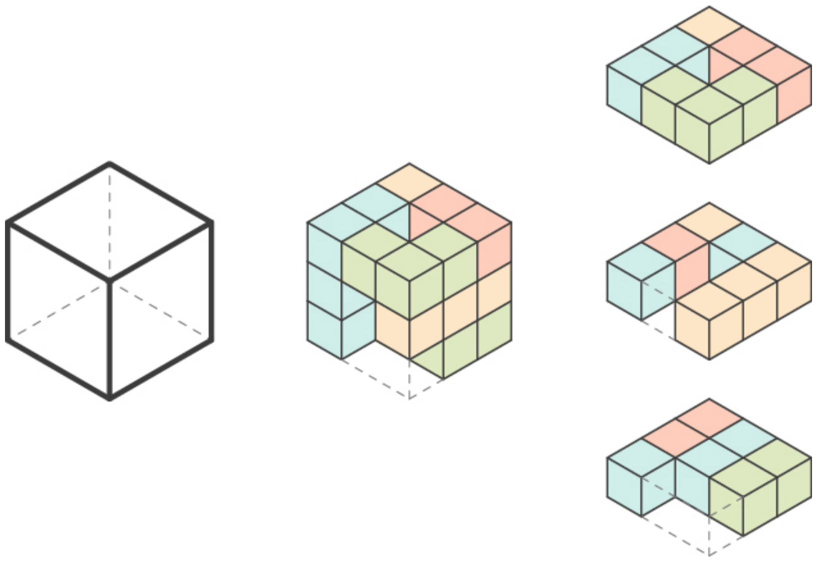
3.4 A toy example

This section provides a toy example to demonstrate how this design platform work.

In the editing mode, the designer could modify the shape by adding or removing 3x3x3 cubes. After the designer adds a cube unit, the IGA will work in 1x1x1 scale, packing the room type components into the cube unit.

Supposing that the designer placed only one cube unit, the IGA would first randomly pack room type components into the cube unit and then iterate crossover, mutation and evaluation operations on the packed components. It finally converged to a solution shown in Figure 3-10. The center of the cube unit is hollow, since the room unit in the center cannot access to any facade of the cube unit directly (no sunlight for that room unit) and would be eliminated through evaluation function.

Figure 3-10: Example of packing components in one cube unit.



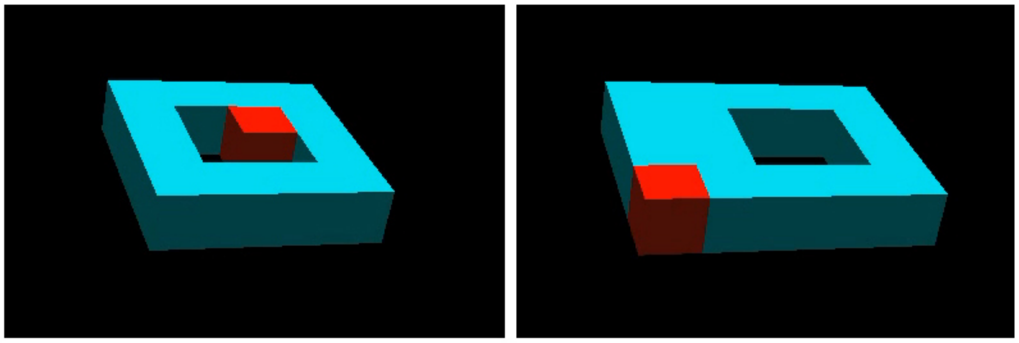
3.5 An application scenario

This section proposes an invented scenario to show how this interactive platform could help the designer to understand the design problem, reconstruct strategies of action based on the feedback provided by the computational tool.

May, an architecture student, decided to use this interactive design platform to create the form for her residential building design project in the early stage of design process. Based on her analysis of residents' demands and preferences, she chose three room types, which she would like to use in her residential building.

For the sake of dwelling quality and community development, she determined to let each living unit have direct access to public space, which means that each living unit will have at least two faces facing the courtyard or street. Besides, May thought it would be nice to have evenly distributed number of units for each room type, so that the building could evenly solve the demands for different room types.

Figure 3-11: First idea: expand the width of one branch to make shape more interesting.

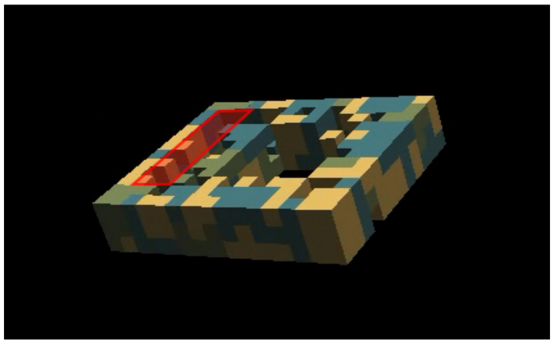


After defined these constraints, May launched the program.

She started with a basic courtyard shape. In the beginning, May expanded the width of one side of the courtyard, to increase the capacity and break the evenly distributed mass.

However, after she checked the consulting mode, she found this move resulted in a wired and narrow gap on the width branch. After several trials, she recognized that she should avoid wide width in the residential building, because of the accessibility constraint she set for this design problem. The living units in the middle of the mass are not able to access either courtyard or street directly, which will get a penalty in the selection function and be eliminated in the next several generations.

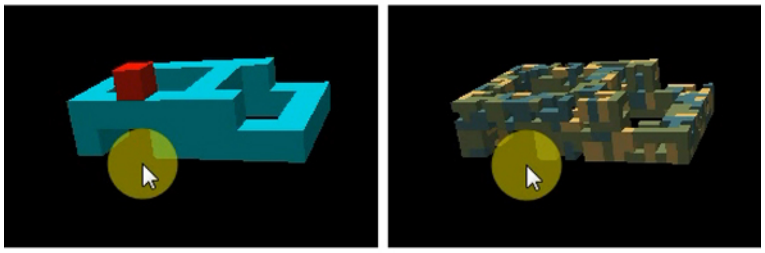
Figure 3-12: Consulting mode: a narrow gap appears on wide width branch.



Realizing this rule, May adjusted her strategy.

Instead of increasing the width of the branch, she decided to keep the width the same and create courtyards with different scales to make the shape more interesting.

Figure 3-13: New strategy: keep the width same and create courtyards with different scales.



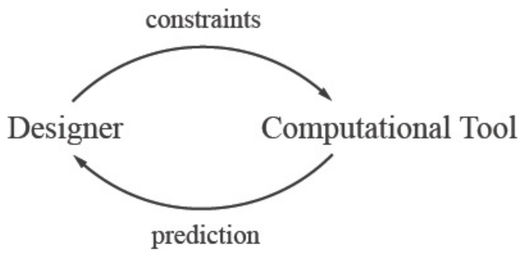
Through checking the consulting mode, May also noticed that the number of room type I (with two unit boxes) was always much lower than other two room types, even though she set the relative proportion of three room types the same. After examining this phenomenon, May found out that there was only one configuration for room type I to avoid getting a penalty.

Thus, May determined to reduce the relative proportion of room type I, since the valid spots for this room type were always less than the other two room types.

Before arriving at the satisfactory shape, May would keep modifying the shape and parameters, and refer to consulting mode to evaluate the current shape from the perspective of computational tools and reconstruct strategies of action. This is an interactive conversation between the designer and the computational tool. The designer keeps passing new constraints (new shape and updated parameters) to the computational tool, and the computational tool offers the designer a prediction based on the given constraints, which will help the designer to reflect on his/her design strategy and knowledge.

In this specific case, May adjusted her design strategy twice based on her observation of the prediction provided by IGA. First, she changed her strategy towards the width of the shape according to the explicit feedback from the IGA, a narrow gap on the wide branch. Second, she changed the parameter for the relative proportion of room types grounded on an observation of the implicit feedback from the IGA.

Figure 3-14: Diagram of interactive conversation.



In this interactive and recursive conversation, the IGA plays the role of a sort of "speedwriter" recording the decision of the designer and offering a prediction about the expectable outcome under the real world constraint upon each decision. Designers concentrate on reflection and creation parts.

Chapter 4

Discussion and Next Steps

The relationship between designers and computational tools has always been a topic that is worth studying. The design platform proposed in this thesis offers a solution to mediate the confliction between the designer (subjective function) and the computational tool (objecctive function), by letting them interact with each other. Thus, it is worth to discuss about the significance of this interaction between designers and computational tools and what this interaction process offers besides generating 3D form.

4.1 From reaction to interaction

Interaction is the key feature of this proposed interactive design platform, which distinguishes it from conventional modeling software. In the conventional modeling software, the software will execute and display exactly what the designer asks it to do. As Usman Haque argued, this is *reaction*, instead of *interaction*. He pointed out that "the process of clicking on a link to summon a new webpage is not 'interaction'; it is 'reaction'. The client-server system behind the link reacts automatically to input, just as a supermarket door opens automatically as you step in front of it."[1]

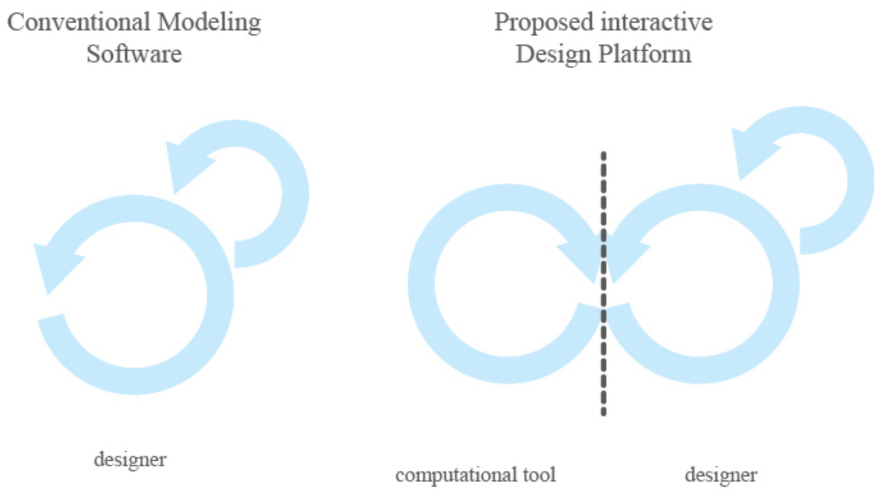
In the conventional modeling software, the function which takes user input and

[1]Dubberly, Hugh, Paul Pangaro, and Usman Haque. *"ON MODELING What is interaction?: are there different types?."* interactions 16.1 (2009): 69-75.

emits the output is fixed, while in the proposed IGA based design platform the function is dynamic. In the consulting mode, the evolutionary algorithm conducts a dynamic stochastic solution searching process. When the program received updated constraints from the designer, it will overlay the latest constraints onto the current solution. The relationship between user input and the output is not a simple linear relationship.

The relationship in proposed design platform is characterized as Managing and Entertaining, "the engagement of a learning system"[2], which consists of a self-regulating system[3] and a learning system[4]. The IGA is the self-regulating system which takes input from the designer and emits output to designer. The designer is the learning system which passes input into IGA and reflects upon the output from it.

Figure 4-1: Relationships between designer and computational tool.


Conventional Modeling Software
Proposed interactive Design Platform
designer
computational tool
designer

[2]Dubberly, Hugh, Paul Pangaro, and Usman Haque. *"ON MODELING What is interaction?: are there different types?."* interactions 16.1 (2009): 69-75.

[3]"A self-regulating system has a goal. The goal defines a relationship between the system and its environment, which the system seeks to attain and maintain." Dubberly, Hugh, Paul Pangaro, and Usman Haque, 2009.

[4]"Learning systems nest a first self-regulating system inside a second self-regulating system....pursues its goal and tests options, it learns how its actions affect the environment." Dubberly, Hugh, Paul Pangaro, and Usman Haque, 2009.

4.2 What does interaction offer us?

The significance of the interaction between designers and computational tools could be summarized as learning, coordinating and collaborating[5].

- Learning. Learning from the computational tool's output and learning from the interaction process. The former focuses on how to solve the current specific design problem. In the previous example, the designer learned that she should bound the width of the building based on the prediction of packing configuration by IGA. The latter contribute to building up the design thinking, including clarifying the design problem, polishing the design strategy, and handling unexpected outcomes. These skills will benefit the designer in a longer term.

- Coordinating. In the architecture design context, coordinating could be interpreted as converging on an agreement. It is about how to formulate the constraints, so that the optimized solution makes sense to the designer. It is also about how to position the design strategy, to make it consistent with the defined constraints. The two participants in this design platform concentrate on the different aspects of the same problem. The interaction between the two sides helps them to understand the other side's intention and make the decisions which fit into the other side's *coordinate.*

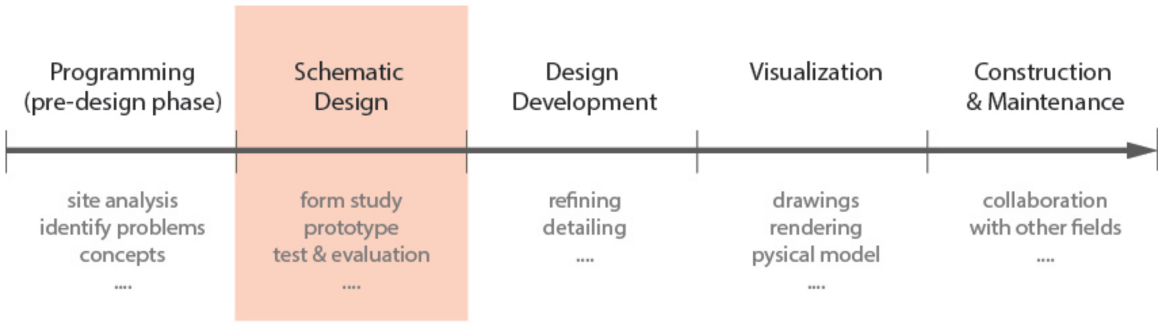
- Collaborating. Collaboration is the process where the two particpants approach the same goal from different directions. The design is responsible for the creative works, while the computational tool focuses on solid real-world constraints. The two approaches sometimes conflict with each other and sometimes support each other. The interaction process enables the two participants to negotiate with each other and achieve a result which satisfies both sides.

[5]Dubberly, Hugh, and Paul Pangaro. *"ON MODELING What is conversation, and how can we design for it?."* interactions 16.4 (2009): 22-28.

4.3 Potential application

This interactive design platform serves a particular stage in the architecture design process, schematic design. After gathering and analyzing information of the site, client's needs, and the surrounding community, the designer already forms a preliminary strategy. This interactive modeling software will be a capable and efficient tool, which could help the designer inspect their solution and polish their strategies through a recursive conversation.

Figure 4-2: Diagram of architecture design process.



This design platform is not merely a simulation or evaluation software, but a partner or consultant who helps the designer to understand the design based on real-world constraints and clarify the design problem. Therefore, the outcome of this design platform is more than a 3D model. It also yields a polished and burnished design strategy together with the designer, which will affect the following design stages.

This interactive design platform could be applied in the design workshop under the following settings, to study the design methods and design patterns.

In preparation phase,

- Determine the elementary building blocks according to the site analysis, regarding scale, shape, quantity, and so forth.
- Define the rules and constraints for the configuration of these building blocks. For instance, building block A has to adjacent to building block B. For another example, all building blocks must have direct access to the courtyard.

- Encoding rules and constraints in the evaluation function.

In interactive phase,

- Edit shape in the editing mode. Document the current design strategy.
- Switch to consulting mode. Document the observation. If an unexpected result occurs, speculate the reason.
- Repeat the previous two steps until reaching a satisfactory shape.

In discussion phase,

- Compare the design strategy in the last iteration with the design strategy in the first iteration. Figure out the difference between the two strategies.
- Trace back the documented design strategies. Explain how the strategy transformed through the modeling process.
- Evaluate the shape in the first iteration. Explain the initial design strategy and discuss how to approach similar design problem in the future.

The goal of this series of modeling, documenting and discussing activities is to help the designer to scrutinize the design process beyond reaching a satisfactory shape.[6] Similar action-oriented approach for reasearch in design theory and methods was proposed in the 1980s, as *the Silent Game*.[7]

In *the Silent Game*, participants alternately make patterns and conjecture patterns in making forms. The patterns always remain implicit, and participants are not supposed to explain the pattern they create until the game ends. This game serves as a tool to couch study in design theory and methods.

Even though the protocol of *the Silent Game* and the settings for applying this interactive design platform in design workshops are quite similar, the two differ significantly regarding the number and type of participants, roles, rules and the goal.

In the proposed design platform, only the designer's side evolves through the process, regarding the understanding of the design problem and design strategies. In

[6] "design is not just steering towards a goal; design is also a process of discovering goals, a process of learning what matters." Dubberly, Hugh, and Paul Pangaro. *"How cybernetics connects computing, counterculture, and design."* Hippie Modernism: The Struggle for Utopia. Walker Art Center, Minneapolis MN (2015): 1-12.

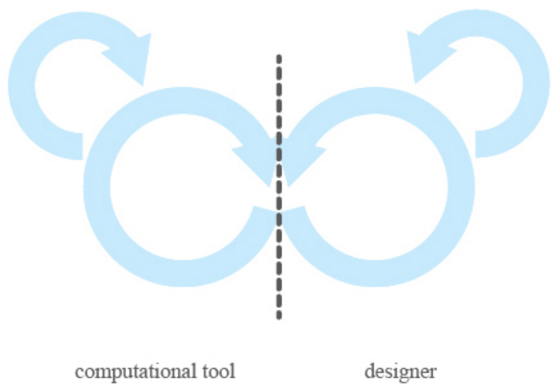
[7] Habraken, N. John, and Mark D. Gross. *"Concept design games."* Design Studies 9.3 (1988): 150-158.

Table 4.1: Comparision of settings.

	the Silent Game	proposed design platform
Participants	At leats two participants (designers).	Two participants. One designer and one computational tool
Roles	Creating patterns and guessing patterns.	Creating shapes and optimizing the packing components under constraints.
Rules	Patterns remains implicit.	Constraints are explicit defined by the designer.
Goals	Research in design theory and methods.	Assist designer to scrutinize the design problem and polish design strategy.

the future work, machine learning will be integrated with IGA, which will enable the computational tool to evolve its evaluation function based on the analysis of given constraints, rather than designer's preference. Under this setting, this interactive design platform could support more sophisticated interactions between designers and computational tools.

Figure 4-3: Conversing relationship between designer and computational tool.



Bibliography

[1] Hayles, N. Katherine. *How we became posthuman: Virtual bodies in cybernetics, literature, and informatics.* University of Chicago Press, 2008.

[2] Maturana, Humberto R., and Francisco J. Varela. *Autopoiesis and cognition: The realization of the living.* Vol. 42. Springer Science & Business Media, 1991.

[3] Schőn, Donald A. *Educating the reflective practitioner: Toward a new design for teaching and learning in the professions.* Jossey-Bass, 1987.

[4] Negroponte, Nicholas. *Soft architecture machines.* Cambridge, MA: MIT press, 1975.

[5] Schőn, Donald A. *"Designing as reflective conversation with the materials of a design situation."* Knowledge-based systems 5.1 (1992): 3-14.

[6] Dubberly, Hugh, and Paul Pangaro. *"ON MODELING What is conversation, and how can we design for it?."* interactions 16.4 (2009): 22-28.

[7] Dubberly, Hugh, Paul Pangaro, and Usman Haque. *"ON MODELING What is interaction?: are there different types?."* interactions 16.1 (2009): 69-75.

[8] Banzhaf, Wolfgang, et al. *Genetic programming: an introduction.* Vol. 1. San Francisco: Morgan Kaufmann, 1998.

[9] Mitchell, Melanie. *An introduction to genetic algorithms.* MIT press, 1998.

[10] Habraken, N. John, and Mark D. Gross. *Concept design games.* Design Studies 9.3 (1988): 150-158.

[11] Schőn, Donald A. *"Learning to design and designing to learn."* NA 6.1 (2015).

[12] Dubberly, Hugh, and Paul Pangaro. *"How cybernetics connects computing, counterculture, and design."* Hippie Modernism: The Struggle for Utopia. Walker Art Center, Minneapolis MN (2015): 1-12.

[13] Cameron, Mark. *"Review Essays: Donald A. SchÃűn, The Reflective Practitioner: How Professionals Think in Action. New York: Basic Books, 1983. ISBN 0âĂŤ465âĂŤ06874âĂŤX (hbk); ISBN 0âĂŤ465âĂŤ06878âĂŤ2 (pbk)."* Qualitative Social Work 8.1 (2009): 124-129.

[14] Mols, Ine, Elise van den Hoven, and Berry Eggen. *"Technologies for everyday life reflection: illustrating a design space."* Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction. ACM, 2016.

[15] Kun, P. E. T. E. R. *Designing for Reflection* Diss. Ph. D. thesis, Chalmers Univ. of Technology, 2013.

[16] Fasoulaki, Eleftheria. *"Genetic Algorithms in Architecture: a Necessity or a Trend?"*. 10th Generative Art International Conference, Milan, Italy. 2007.

[17] Farooq, Humera, and Muhummad Tariq Siddique. *A Comparative Study on User Interfaces of Interactive Genetic Algorithm.* Procedia Computer Science 32 (2014): 45-52.

[18] Darani, Zahra Sheikhi, and Marjan Kaedi. *Improving the interactive genetic algorithm for customer-centric product design by automatically scoring the unfavorable designs.* Human-centric Computing and Information Sciences 7.1 (2017): 38.

[19] Oliver, Antoine, Nicolas MonmarchÃľ, and Gilles Venturini. *Interactive Design of Web Sites with a Genetic Algorithm.* ICWI. 2002.

[20] Kim, Hee-Su, and Sung-Bae Cho. *Application of interactive genetic algorithm to fashion design.* Engineering applications of artificial intelligence 13.6 (2000): 635-644.

[21] Min, Hyeun-Jeong, and Sung-Bae Cho. *Creative 3D designs using interactive genetic algorithm with structured directed graph.* Pacific Rim International Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2004.

[22] Nishino, Hiroaki, et al. *A 3D modeling system for creative design.* nformation Networking, 2001. Proceedings. 15th International Conference on. IEEE, 2001.