Parametric Tools and Digital Fabrication for the Design of Luminous Ceilings

by

Rita Saad

Bachelor of Architecture (2002)
Lebanese American University

Submitted to the Department of Architecture
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Architecture Studies
at the Massachusetts Institute of Technology
June 2004

Signature of Author ....
Department of Architecture
May 19, 2004

Certified by ....
Lawrence Sass
Assistant Professor of Computation and Design
Thesis Advisor

....
William Lyman Porter
Muriel and Norman Leventhal
Professor of Architecture and Planning
Thesis Advisor

Accepted by ....
Julian Beinart
Professor of Architecture
Chair, Committee on Graduate Students

Readers of this Thesis

....

William J. Mitchell

Professor of Architecture and Media Arts and Sciences

....

Maria Thompson

Ph D candidate in Design and Computation

Parametric Tools and Digital Fabrication for the Design of Luminous Ceilings

by
Rita Saad

Submitted to the Department of Architecture
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Architecture Studies
May 2004

Abstract

The digital phenomena constitute a fundamental change in how designers accomplish a wide range of the complex processes of design. This thesis investigates the use of computation in the context of architectural lighting design. It particularly looks into how cutting edge computational tools -- such as digital fabrication and parametric tools -- can be combined with the Light Emitting Diodes (LED) technology to create luminous architectural elements. Work in this field is of most relevance in a moment when the implementation of LED systems is expected to establish a new paradigm in architectural illumination. Results from recent technology roadmaps show that by the year 2020 LEDs will be replacing incandescent, halogen and fluorescent lamps and will become the primary choice for general lighting applications. Because LED architectural applications are not widely understood by the industry, a successful implementation process will be highly dependant on multidisciplinary design research, where many design experimentations will have to occur. New approaches are needed where the technical advantages of LEDs – they are more efficient, have longer life of operation, are rugged and compact, produce the entire color spectrum, and are fully controllable – are used to promote better lighting design quality. It is in this context that my research takes place, utilizing advanced computational tools to explore innovative design possibilities for lighting systems with embedded LEDs.

This thesis describes a sequence of experiments to design and build a system of luminous ceiling tiles made of acrylic pieces and equipped with embedded LEDs. First, I use programming to generate parametric 3D models of the ceiling tiles. A series of variations of an initial design of the tiles are accomplished through the manipulation of control parameters. After the first set of 3D models is created, I use digital fabrication techniques to build prototypes of the models, which are tested with LEDs and evaluated in terms of their lighting performance. Finally, I develop the experiments to create an entire luminous ceiling area, and the design achieves an overall result rather than being restricted to individual elements.

Advanced lighting systems enhance the quality, flexibility and cost effectiveness of light, and digital fabrication techniques improve the optimization of computer-based methods of design. The results of my experiments show that lighting systems can greatly benefit from the testing of the design and the technical performance before installation in the architectural space. In this context, parametric tools and digital fabrication technologies demonstrate exceptional wealth for both the conceptual and the optimization phases of lighting design in architecture.

Thesis Supervisor: Lawrence Sass
Title: Assistant Professor of Computation and Design

Thesis Supervisor: William Lyman Porter
Title: Muriel and Norman Leventhal Professor of Architecture and Planning

Acknowledgements

I would like to express my gratitude for the generosities and indulgences of all those who helped in the past two years

My thesis committee, Larry Sass and Bill Porter, Bill Mitchell and Maria Thompson for their great inspiration and advice

Dr. Makarand Chipalkatti and Joe Laski from R & D LED Lab OSRAM Optosemiconductors, for making this thesis possible

Professor James Gips for believing in my work

Axel Kilian, Loai Naamani and Stelios Dritsas for their precious help and generous assistance

Khalil Khouri for his beautiful mind

My fellow SMArchS students for their great friendship

My parents, brothers and Jad for their constant support

My outstanding gratitude for Maria whose collaboration in my research made me benefit from her sponsorship by OSRAM Optosemiconductors lighting manufacture. She has always enlightened me with her good advice, guidance and inspiration; and has constantly shared with me her expertise.
To her I dedicate this work, the fruit of what we shared in spirit and soul.

For the One through whom Light was

"While you have the light, believe in the light,
so that you may become children of the light"
(John 12:36)

Table of Contents

Parametric Tools

and

Digital Fabrication

for the

Design of Luminous Ceilings

CHAPTER 1: INTRODUCTION

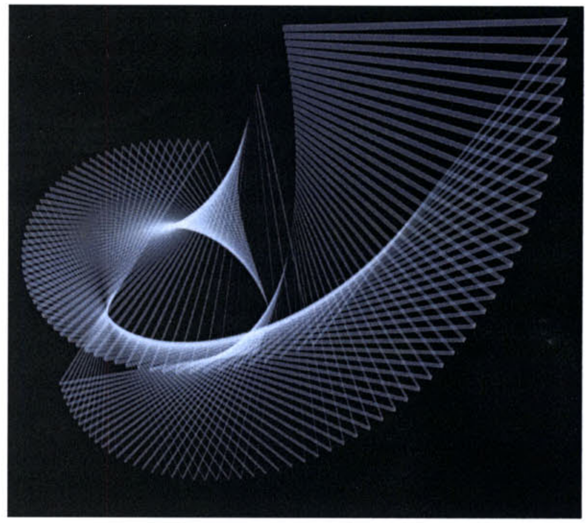
1.1 Overview

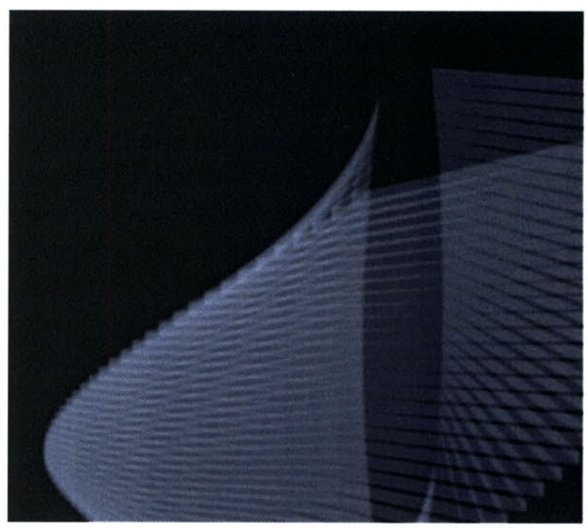
1.1.1 Scope

In this research I develop a systematic method of designing and constructing luminous ceilings using parametric tools and digital fabrication techniques.

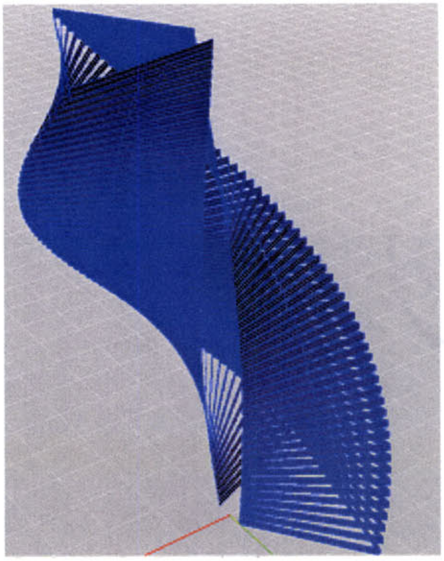
I started by defining a process to design and build luminous ceiling tiles equipped with embedded light emitting diodes (LEDs) as light sources. Through this process I first generated, modeled and evaluated various samples of luminous objects by creating digital and physical models from parametric tools and digital fabrication techniques. Then, the objects developed into luminous ceiling tiles as a result of the manipulation of parameters in the code (a Rhinoscript). The design variations created in the tiles produced different ceiling geometries. In comparison to the traditional flat luminous ceilings, these tiles can be perceived as architectural elements expressing aesthetic possibilities which may offer different illumination performances.

After an elaborate design investigation in luminous ceilings, I used Shape Grammars to capture the principles of shape rules that define a design and thus to rationalize the design process. The shape rules of different designs can be recombined together in different sequences to generate new and unpredictable designs. I finally built an input interface following the logic of Shape Grammars. Through the interface, users can manipulate the design in terms of the parameters and of the rules. During the process of building the interface, I realized many meaningful features of Shape Grammars in producing different ceiling configurations.

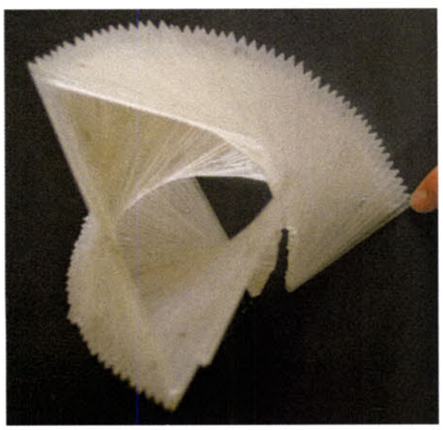




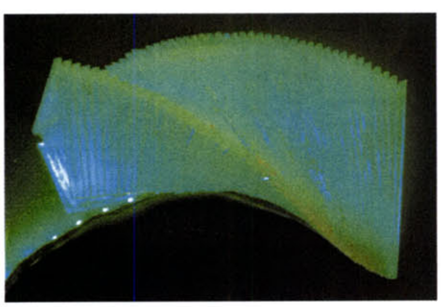
Simulations of a parametric model produced in Rhinoceros and rendered in 3D Studio VIZ



parametric model in Rhinoceros



physical prototype of the model



model tested with light

The study of parametric and digital fabrication tools originates from my experience in the workshop "Generative and Parametric Tools for Design and Fabrication" [1], in Spring 2003 in collaboration with Foster and Partners [2]. During this workshop I explored parametric design tools combined with rapid prototyping techniques to generate and evaluate design solutions for design problems focused on light. In this exploration, the parametric tool I used was a three-dimensional (3D) modeling software: "Rhinoceros", with scripting support in Visual Basic (VB) programming language. The workshop's computational model initiated this research work.

1.1.2 Structure

In Chapter 2, I describe a series of computational exercises in parametric modeling, which constitute the different stages of the design investigation to reach a luminous tile. Then, I evaluate the different designs by observing and comparing the experiments I built; and I show how physical models produced by digital fabrication techniques enable the testing of the design and the performance of lighting systems.

In chapter 3, I introduce a theoretical framework for the design of luminous ceilings using Shape Grammars to generate various design solutions. Finally, I propose a method of design for luminous ceilings and I express this method in an input interface which demonstrates the design rules of shape grammars and parametric modeling.

The appendices are derivations of luminous ceilings created by the input interface. They

---

[1] http://architecture.mit.edu/subjects/sp03/4182.html, Design and Computation Group, Department of Architecture and Planning, MIT

[2] Foster and Partners - www.fosterandpartners.com

also include examples in Rhinoscripting using VB codes to design the objects, the tiles and the ceilings; and they contain a sample code written in C# programming language for the creation of text boxes to build the interface used in Rhinoceros.

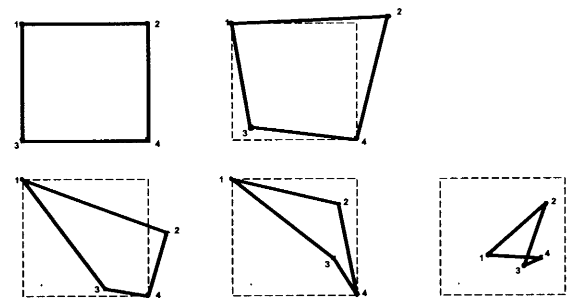
1.2 Background

1.2.1 Parametric Design

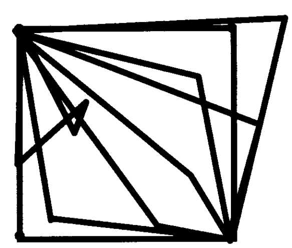
"Modeling buildings in fully parametric 3D computer-aided design (CAD) systems offers numerous benefits in terms of productivity, the ability to rapidly generate design alternatives at different levels and elimination of errors that result from the disparity between different drawings in current practice."[3]

Parametric design constitutes a rational design process defined in a set of numerical relationships between variables and parameters. It empowers the designer with the ability to create variations in a design by changing the value of parameters or variables.
Therefore, a design is parametric when it is based in parameters and variables.

By expressing parametric relationships between elements, it becomes possible to describe a whole family of possible outcomes from an original design. The parametric relationships are defined by a parametric set-up, a series of instances of a parametric element, each a transformation of the initial shape with a different value for the parameter.



series of instances of a parametric element, each with a different value for the parameter (each a transformation of the initial shape)



the repetition of a sequence of transformed elements defines the geometry of the final piece

$$f(x) = ax$$

A variable (a) explains the formal relationships in a given state. It refers to the initial state. While a parameter (x) is defined as a variable to which other variables are related. It explains the transformation

[3] Sacks Rafael, Eastman Charles, and Lee Ghang, "3D Parametric Modeling in Building Construction with Examples from Precast Concrete", Automation in Construction 13 (2004) 291-312

The sequential repetition of the transformed elements defines the geometry of the final shape, and is a result of a generative process. The ability to represent a geometry in parametric relationships enables a systematic analysis of the geometry and the variation in size, position and shape of its various elements because a parametric expression can represent a set of geometrical forms instead of just one.

The concept of parametric design was first anticipated in the 1963 MIT Ph.D. thesis of Ivan Edward Sutherland "Sketchpad, a Man-Machine Graphical Communication System". Sketchpad was a computer program that introduced many computational techniques including recursive methods for geometric transformations, and an object oriented programming style. The main idea was to have master drawings which one could instantiate into many duplicates. If the master drawing was changed all the instances would change as well. Another major invention in Sketchpad pertaining to parametric design is to let the user easily constrain selected geometrical properties in the drawing. For instance the length of a line or that of two lines should have a specific angle between them. This work provided a foundation to what is today known as "parametric design".[4]

Today, the most prominent examples of parametric methods of design can be found in Norman Foster's and Frank Gehry's architecture. The study of parametric design has constituted the scope for many academic researches including Yanni Loukissas 2003 MIT SMArchS thesis ("Rulebuilding: Exploring Design Worlds Through End-User Programming", Master of Science in Architecture Studies, MIT 2003); Dennis

---

4 http://www.sciencedaily.com/encyclopedia/sketchpad

Shelden 2002 MIT PhD thesis ("Digital Surface Representation and the Constructibility of Gehry's Architecture", Doctor of Philosophy in the Field of Architecture: Design and Computation, MIT 2002); and Mark Burry 2002 research paper ("Rapid Prototyping CAD/CAM and Human Factors", Automation in Construction 11 (2002) 313– 333).

1.2.2 Digital Fabrication

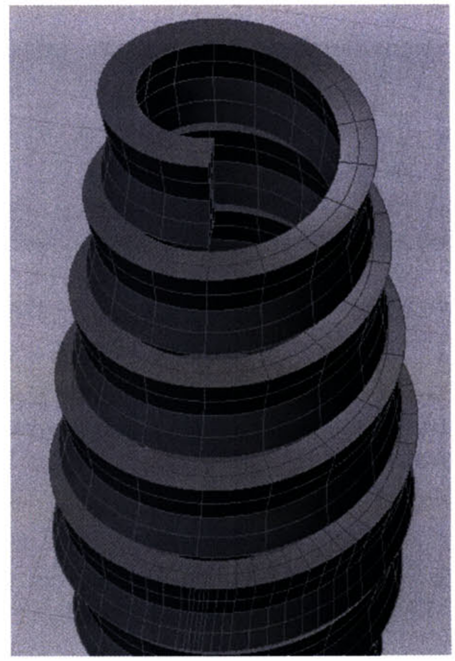
Digital Fabrication or Rapid Prototyping (RP) is a technology used to:

(1) Build physical models and prototype parts from 3D computer-aided design (CAD) data. RP and CAD solid modeling are complimentary technologies, so they help to justify one another.

(2) Construct virtually any shape that can be modeled on a CAD system. RP parts are used to verify form, fit and function, and to evaluate a fixture design before a final product is manufactured.

(3) Shorten the design and the manufacturing cycles, as well as improve the quality of a design, by allowing designers and engineers to create a solid model directly from CAD data.

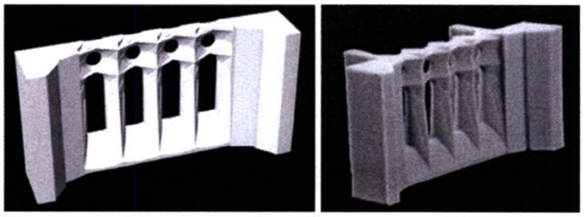
Rapid Prototyping techniques are applied in this investigation for the creation of concept models and physical prototypes of ceiling tiles to verify their form, fit and function and to evaluate various performance characteristics, such as the brightness and the uniform distribution of light in the tiles.



digital model of a helix shape drawn in Rhino

physical prototype of the helix model produced by a 3D-printing (RP) machine

In a lighting system, various aspects of the design and the fabrication of the system, such as material properties and position of the lighting source can be decided upon based on the analysis of a 3D physical model. The development of new lighting technologies, such as LED (Light Emitting Diodes), demonstrates the value of rapid prototyping capabilities. Today, with the advent of solid-state lighting technology and its flexibility in performance control and design integration, Rapid Prototyping techniques offer a platform to experiment and to investigate the potentials of this technology in architectural applications. The scenario for the evaluation of light performance becomes a complementary partnership between parametric modeling and digital fabrication techniques for the design of lighting systems.

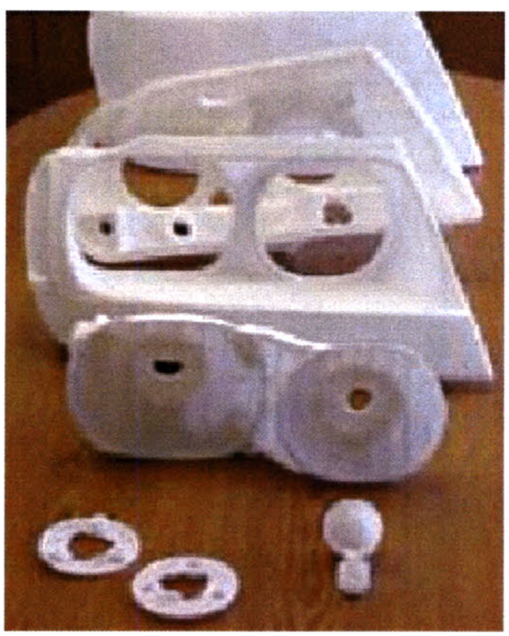


digital model and physical prototype of a façade from the Sagrada Familia Church project

from: Mark Burry 2002 research paper ("Rapid Prototyping CAD/CAM and Human Factors", Automation in Construction 11 (2002) 313– 333)

A good example of a Rapid Prototyping investigation in Architecture is Mark Burry's work on the Sagrada Familia Church project[5]. In this work, RP techniques are used to analyze and test different aspects of the design. Together with parametric design tools, RP techniques prove their value to describe the formal complexities of Gaudi's architecture and to reproduce its details. As expressed by Mark Burry in his paper on Rapid Prototyping:

"Once the computer model has been sent to the design office on site, it became apparent that although the model could be 'read' once rendered, it was a confusing mass of lines as a wire-frame for anyone not involved directly in its production. It was therefore difficult to evaluate whether the computer model matched the original without a physical prototype. Given the fiddly nature of making this particular piece in gypsum plaster, it was of interest to check the viability of rapid prototyping. The result was judged to be of real benefit".



external automotive lighting rapid prototypes built in a 3D-printing machine
(image by Valeo Sylvania)

The use of Rapid Prototyping techniques for the testing of the performance of lighting systems has been successfully applied for more than ten years in the automotive lighting industry, to test equipment such as headlamps and tail light.
Rapid Prototyping tools are used by lighting manufactures such as OSRAM SYLVANIA[6] in the product development process of automotive lighting equipment to optimize and expedite the design process.

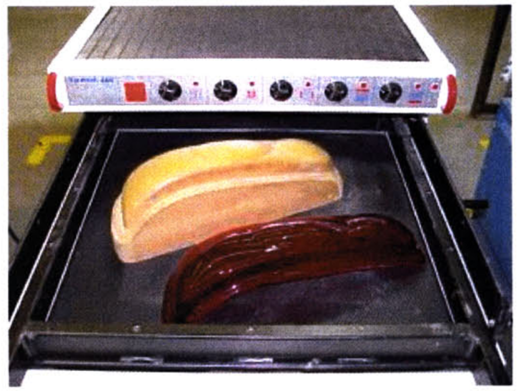
---

[5] mburry@deakin.edu.au (M. Burry), "Rapid Prototyping, CAD/CAM and Human Factors", Automation in Construction 11 (2002)

[6] Valeo Sylvania Rapid Prototype Group, OSRAM SYLVANIA Products Inc., Automotive Lighting (http://www.sylvania.com/press/980818.html)

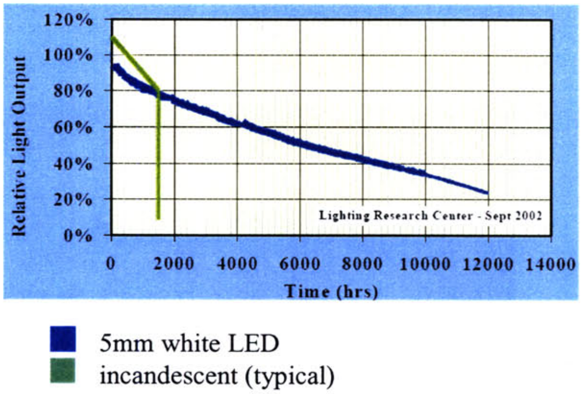
Various pieces such as optical lenses, reflectors, and housings are prototyped to evaluate design and functionality. The performance of these components is also tested in terms of their aesthetical properties. After a performance pattern is determined, physical prototypes of components are built to ensure that the design features of a vehicle support the components performance. With a variety of fabrication techniques including 3D printing, photolithography and vacuum forming, the automotive industry tries to approximate the cosmetic appearance of optic lens and reflectors to the required reflectivity of light. These methods have made possible the insertion of optical patterns in mock-ups to reach an optimal performance of light.
In addition to significant time-savings on the product development phase, which also results in substantial cost reduction, Rapid Prototyping techniques make prototype styling reviews possible, and promote 'healthy' and efficient lighting products, the ends of which are ground to the required geometric shape. It gains momentum with the production of scaled presentation that can fit in context and is tested on the vehicle to validate style, technical properties, and optical design of the light fixture.

The application of Rapid Prototyping technology in automotive lighting industries provides a significant example for architectural lighting design. This technology supports the creation of prototype components and assemblies for the evaluation and the verification of both design and performance.

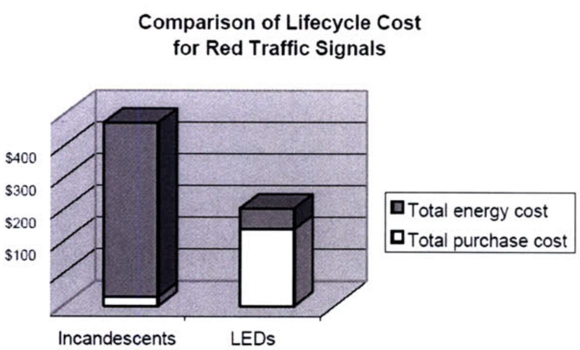


formech 660 vacuumformer used to produce physical prototypes of automotive lighting components (image from Valeo Sylvania)

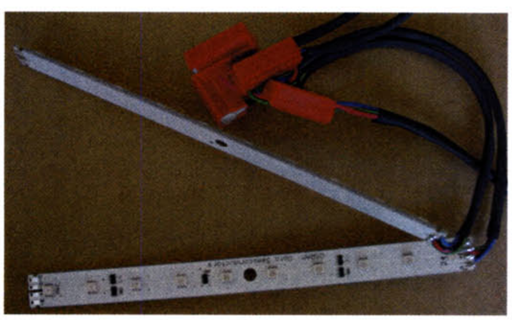
fit & finish rapid prototype for design verification (image from Valeo Sylvania)



efficiency of LED in comparison to that of an incandescent bulb
(Source: Lumileds Lighting LLC)



Energy Savings
(Source: CEE Council for Environmental Education)



strips of color-mix LEDs
rugged Solid State Construction

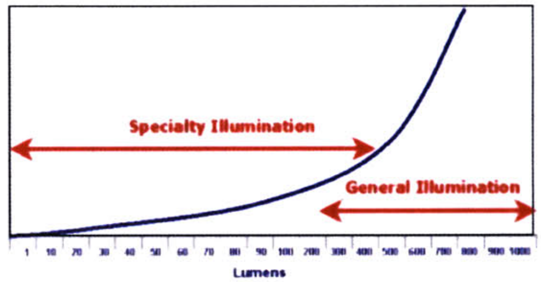
1.3 Significance

This work constitutes an unprecedented example of the use of parametric tools in lighting design. It also opens a path of investigation for the use of LEDs in architecture. Although LED technology has been broadly applied in different markets such as automotive, communications and signs; it hasn't yet become recognized in architectural applications. As for today, the general properties of LEDs have not reached the necessary development for illumination in architecture. However, LEDs offer many advantages over the traditional light bulb:

(1) Long lifetime
(100,000 hours as opposed to 1,000 hours for incandescent lamp and 10,000 for fluorescent lamp)
(2) High efficiency
(The luminous efficiency[7] of LEDs is expected to reach in 2007 over 80 Lumens/Watt that is approximately 6 times more than one incandescent lamp)[8]
(3) High reliability
(Rugged Solid State construction)
(4) Energy savings
(LED signal consumes approximately 88 percent less energy than a comparable incandescent signal in the same application)
(5) Controllability
(Illumination over a full color spectrum with color-mix strips of LEDs, used in the experiments of this work)
(6) Heat reduction

[7] the efficiency is the Light Output Ratio (measures how much light is given out of a fixture)
[8]http://www.crd.ge.com/cooltechnologies/pdf/2002grc078.pdf

Table 1 shows the advantages of LEDs in comparison with conventional light bulbs. Because of the great advantages presented by LEDs, research institutions in academic, corporate and governmental spheres are making significant investments to further the development of this technology. Results from recent technology roadmaps show that by the year 2020 LEDs will be replacing incandescent, halogen and fluorescent lamps and will become the primary choice for general lighting applications (table 2). While some applications of LED technology may be as simple as replacing one light bulb with another, far more innovative changes may involve entirely new mechanisms for utilizing light. For example, architectural elements such as walls, columns or ceilings, could become interactive luminous surfaces with various digitally controlled performances. Despite the great potential and promises of LED technology, new approaches are needed where the technical advantages of LEDs are used to promote a better lighting quality for architecture.



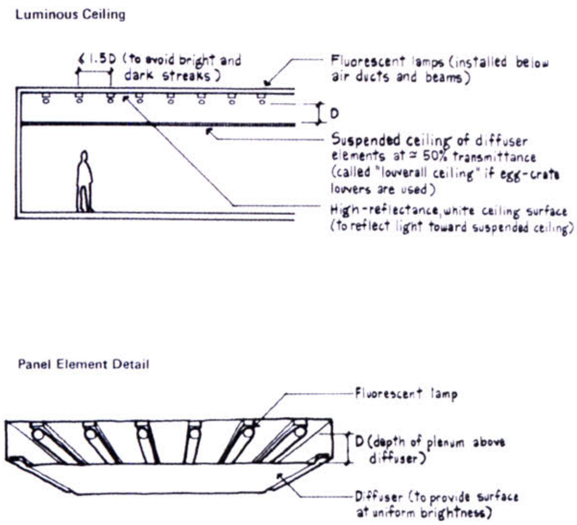
Applications opportunities for LEDs based on Light Output
IEEE 2002

Light Source	**Efficiency**	**Lifetime**
Incandescent bulb	16 lumens/watt	1000 hours
Fluorescent lamp	85 lumens/watt	10,000 hours
Today's white LEDs	25 lumens/watt	20,000 hours
Future white LEDs	up to 200 lumens/watt	100,000 hours

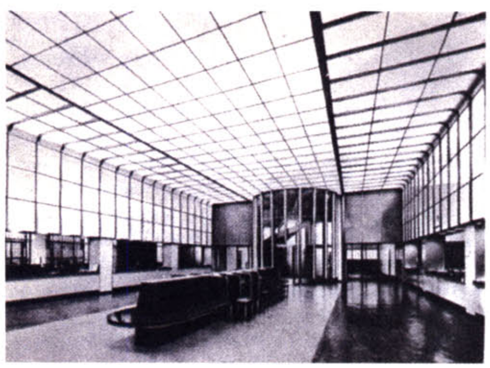
Table 1 - Performance comparison between LEDs, Incandescent and Fluorescent light bulbs -- Sandia National Laboratories

Year	**SSL-LED 2002**	**SSL-LED 2007**	**SSL-LED 2012**	**SSL-LED 2020**
Luminous Efficacy (lm/W)	25	75	150	200
Lifetime (khr)	20	>20	>100	>100
Flux (lm/lamp)	25	200	1,000	1,500
Input Power	1	2.7	6.7	7.5
Lighting markets Penetrated	Low-Flux	Incandescent	Fluorescent	All

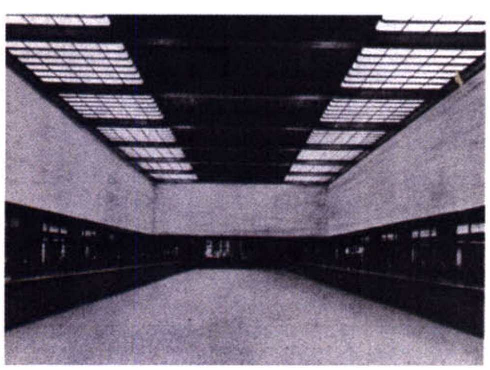
Table 2 - Performance targets for SSL-LEDs -- OIDA Technology Roadmap, 2002



typical detail of a luminous ceiling from 1983 (David Egan, "light and Form" p.107, *Concepts in Architectural Lighting*)



continuous area of surface brightness (1932, Berlin)



luminous side areas (1933, Berlin)

glass ceilings lighted with fluorescent fixtures (Kohler, "The Luminous Ceiling" p.149, *Lighting in Architecture*)

It is in this context that my research takes place, utilizing advanced computational tools to explore innovative design possibilities to create lighting systems with embedded LEDs.

The central point of my argument is that to use LEDs as general lighting it will be necessary to bring these sources to ceiling systems, and to have these systems respond to lighting design problems that have not yet been fully resolved by traditional lighting equipment. Following this approach, I have explored the use of LEDs in architectural illumination.

The history of luminous ceiling technologies originates in the nineteen thirties when fluorescent lamps were installed above a dropped ceiling made of a flat diffuser material. The examples shown here give a view of the principles of luminous ceilings. They are flat, structured and subdivided. They are continuous or interrupted by opaque areas. The ceiling can be illuminated from an indirect lighting; or with an overhead lighting[9].

The following statement describes some of the problems encountered in the early examples of luminous ceilings, which reflects the relevance of today's technologies in solving them:

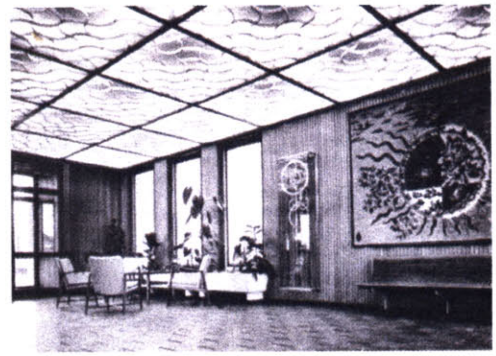
"*Luminous ceilings provide even illumination at low luminance (e.g., less than 250 foot-lambert*[10]*). Many applications of luminous ceilings are oppressive and monotonous, appearing visually as an overcast sky at low elevation. In nature clouds are generally pleasing as they constantly change, providing visual interest.*

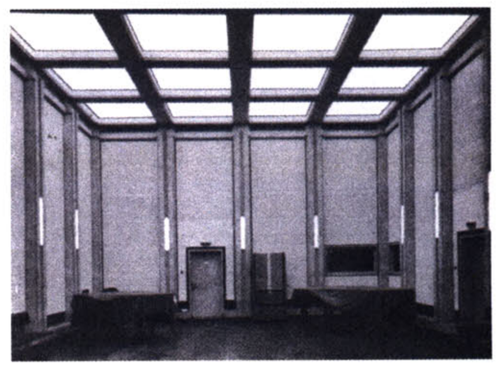
[9] Kohler Walter Dr., "The Luminous Ceiling" p.149, *Lighting in Architecture* (New York: Reinhold Publishing Corporation, 1959)

[10] Luminance unit equal to one Lumen per square foot
1 Ft-Lambert = 3.426 Candela/m2

Luminous ceilings, on the other hand, are static and therefore can be dull and gloomy... In addition, even illumination will provide poor modeling conditions. Luminous ceilings often emphasize maintenance problems by showing dirt and uneven joints.

However, they can be used to conceal mechanical services (pipes and ducts) and structural elements."[11]





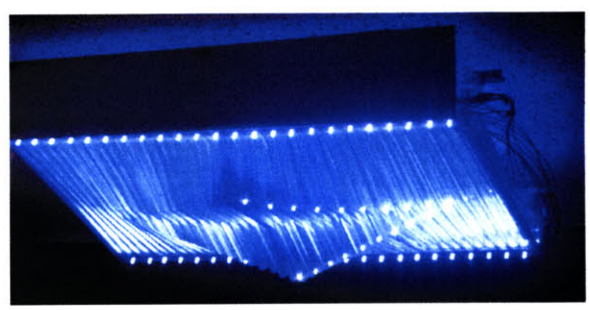
effects of old examples of luminous ceilings (Kohler, "The Luminous Ceiling" p.149, *Lighting in Architecture*)

Today, the use of parametric design tools in architectural lighting may offer a great help for the improvement of luminous ceilings by transforming the flat typology into various geometries as a result of the operation of parameters. The implementation of these geometries and the technical, functional and aesthetic solutions for the design can be handled by the new methods of digital fabrication. Above all, the innovation in luminous ceilings is supported by the Solid State Lighting sources which can be embedded in the system material and become part of the construction process instead of being a manufacture product that is added to the architecture.

The system that I reached in this work attempts to solve the stated problems of traditional luminous ceilings through:

(1) Innovative Shapes:
The creation of non-flat luminous ceilings, continuous surfaces, and different luminous configurations.

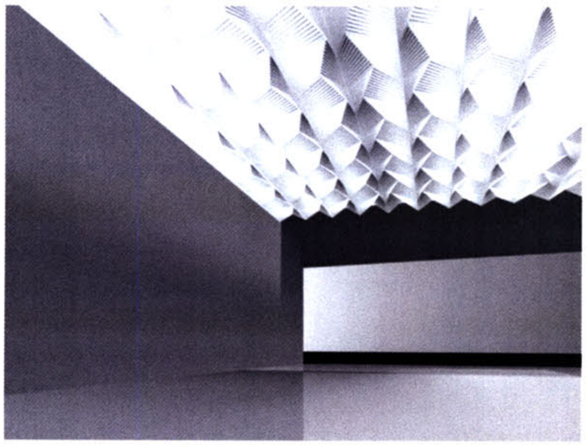
(2) New Lighting Possibilities:
The creation of a system with dual lighting components (diffuse and direct); performance control with an illumination over a full color spectrum; and design integration with embedded light sources.



system of luminous tiles with embedded LEDs for diffuse and direct lighting

[11] Egan M. David, "Light and Form" p.107, *Concepts in Architectural Lighting* (USA, Mc Graw-Hill, Inc.: 1983)

(3) New Lighting Design Process:
The creation of parametric design rules for the accommodation of different needs through different design possibilities.



simulation of a parametrically generated luminous ceiling

With these computational design and fabrication methods I have come to explore LED technology in architectural lighting design through the creation of luminous ceiling tiles.

CHAPTER 2: CONSTRUCTION OF A LUMINOUS TILE

2.1 Parametric Modeling

The first phase of the research presents a sequence of computational exercises that begins with the modeling of 3D objects manually without introducing any rules. Then, one 3D object is selected to be expressed parametrically in a Visual Basic code. The exercise of the parametric model develops the 3D object into a flatter piece which culminates in an architectural tile. It is the result of the operation of control variables and parameters in the parametric code.

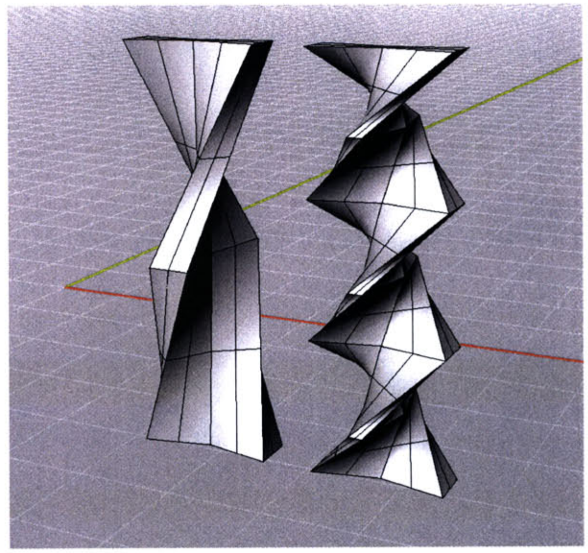
2.1.1 Investigation of the Shape

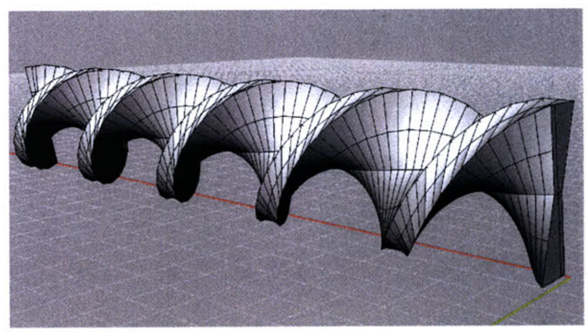
The design exploration of this work starts by creating a 3D digital model of an object with a helix shape in Rhinoceros-3D, a modeling program for designers. This shape is freely manipulated and altered with transformations to investigate new designs. In a next step, the 3D model is parameterized in a code using Visual Basic as a programming language for scripting. The parametric 3D object permits the investigation of various possible outcomes through the manipulation of variables and parameters in the code.

The parametric expression of the design is based on the numerical models described by William Mitchell in his book *Digital Design Media*:

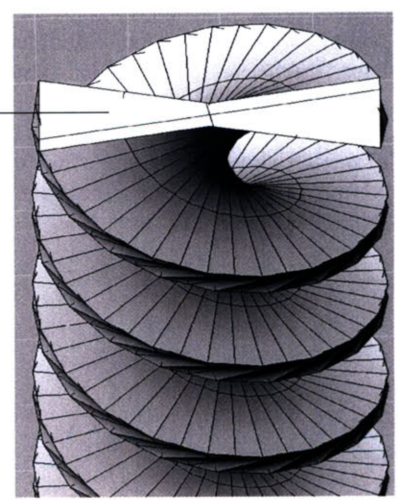
"They describe the behavior of systems they interest us in terms of input variables, output variables, and functions specifying how output variables depend upon input variables." [12]

[12] Mitchell, William, and Malcolm McCullough, "Numerical Models", *Digital Design Media* (Van Nostrand Reinhold, 1991), p.32

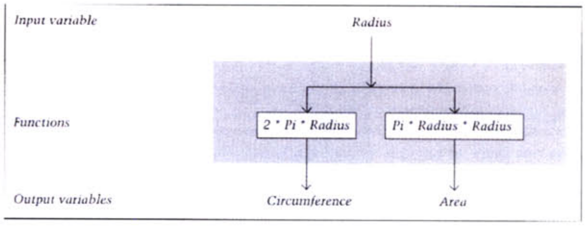




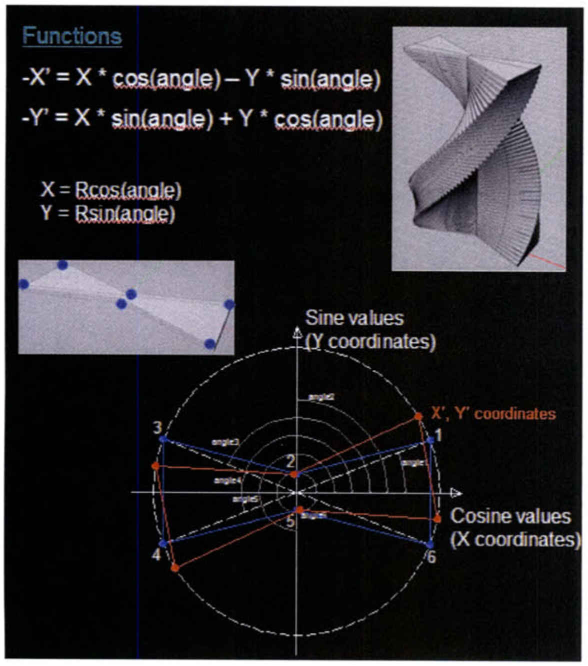
free-hand modeling of 3D objects in Rhino based on a helix shape



first model (3D object) parametrically expressed



numerical model of a circle
(William Mitchell, *Digital Design Media*, 1991)



numerical model of the 3D object (a helix shape) based on the model of William Mitchell

X and Y are the polar coordinates of the edge corners of the initial shape (section of the object)
X' and Y' are the polar coordinates of the rotated shape (it is the Output)

Based on these numerical models the parametric relationships are established through a system of input variables, functions and output variables in a Rhinoscript. The input variables of the 3D object are

(1) The angle of rotation, which determines the shape of the section.
(2) The radius of the circle in which the section of the object is contained. It determines the size of the section.
(3) The polar coordinates of the six corners of the section.

These input variables are entered into numerical functions and the output result of this operation is the new polar coordinates of the rotated section. This numerical model is hence used to draw the section of the 3D object and to transform it through the rotation of the edge corners of the section along its center.

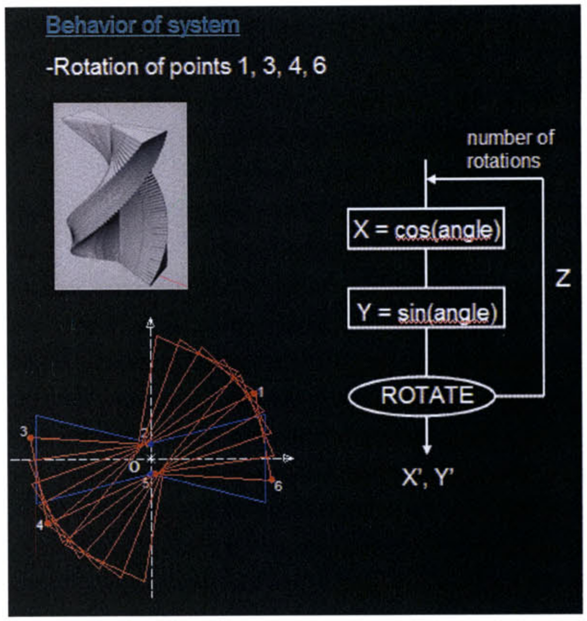
Following the first parametric expression of the design, a loop is created to construct the 3D object like the helix shape in Rhinoceros. The loop is the conditional statement of the procedure that constructs the 3D object through the repetition of the object's sectional profile.

In reference to William Mitchell's "Procedural Programming Language"[13]:

"*A procedure consists of declarations and instructions to perform actions –to accept input values, to perform arithmetic operations, and to store, display, or print output values. It is a sequence of instructions that can incorporate control statements that specify loops, branches and hierarchies*".

[13] Mitchell, William, and Malcolm McCullough, "Modeling with Procedural Programming Language", *Digital Design Media* (Van Nostrand Reinhold, 1991), p.34

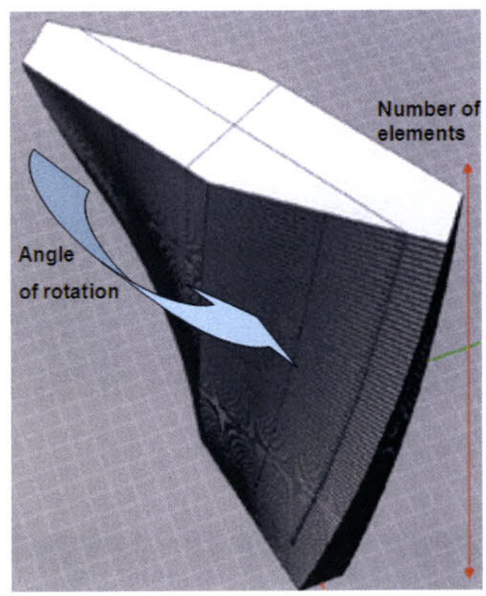
The loop determines the number of rotations of the sectional profile and the procedure behavior is a repetition of this profile with the rotation. The numerical model and the conditional statement of the procedure generate a 3D object from the design of a section and its repetition with a rotation. Therefore, by manipulating the numerical model with the angle of rotation and the polar coordinates of the corners of the initial section, transformations happen on the section, and consequently, the geometry of the final piece is re-defined.
Using a scripting language, the geometry can be manipulated by varying key parameters and variables. Many iterations are performed to develop the design. The parameter of this design is the angle of rotation.



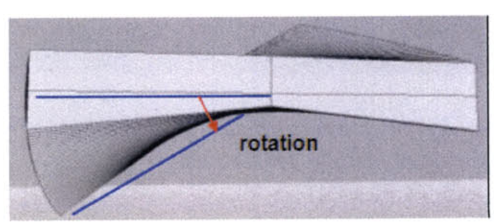
procedure based on a conditional statement (a loop) which repeats the initial shape a number of times to build up the 3D Object

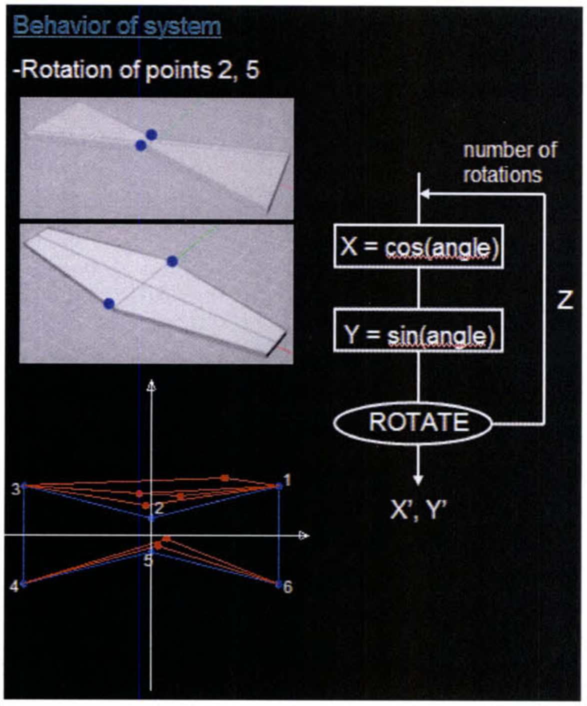
2.1.2 From a 3D Object to a Flat Piece

The next step in the parametric process is a progression from a 3D object to a flat piece in order to develop the design into an architectural element. The flattening of the 3D object is achieved through the control of the angle of rotation -which is a parameter in the code- on the generative process that transforms and repeats the initial section.
The transformation in each repetition of the initial section is the output of the rotation of the center corners of the section while the edge corners are fixed. The recurrence of these transformations with a smaller angle of rotation generates a flatter object.
The different positions that the rotation assigns to the center corners of the initial section determine different profiles and different thicknesses for the generated surface. While these positions can be determined once in the numerical model as it draws the initial shape, they can vary along the repetition of this shape in the generative process, enabling different designs for the tiles.

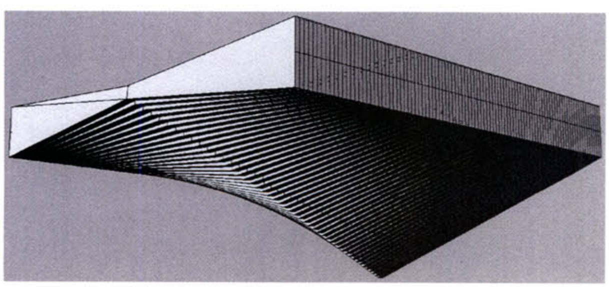


the 3D object becomes flatter as a result of the control of the angle of rotation on the generative process

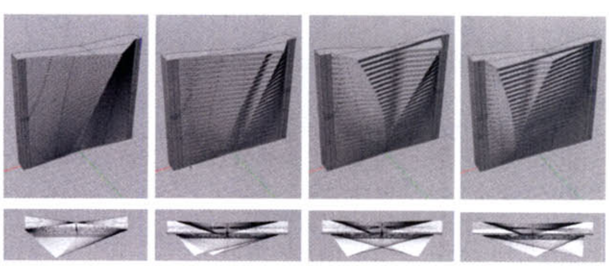




numerical model of the surface (flatter object)
the edge corners are fixed and the rotation is applied on the center corners, which result is an architectural tile



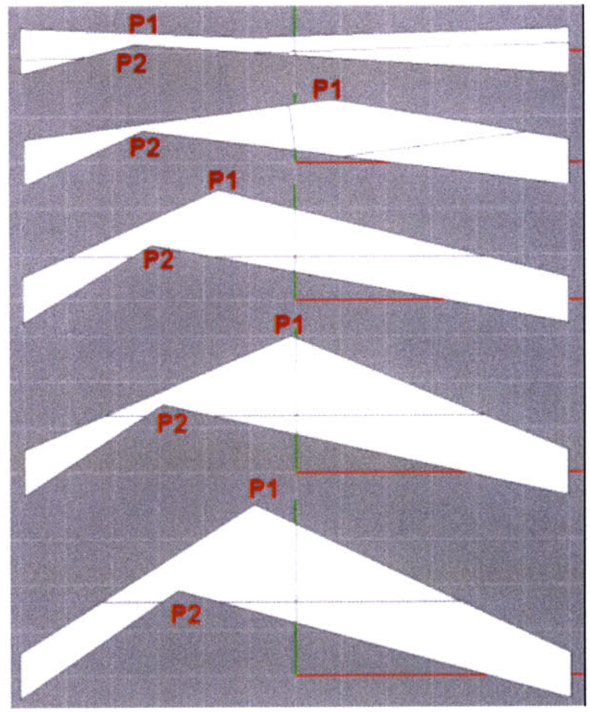
architectural element (a ceiling tile)



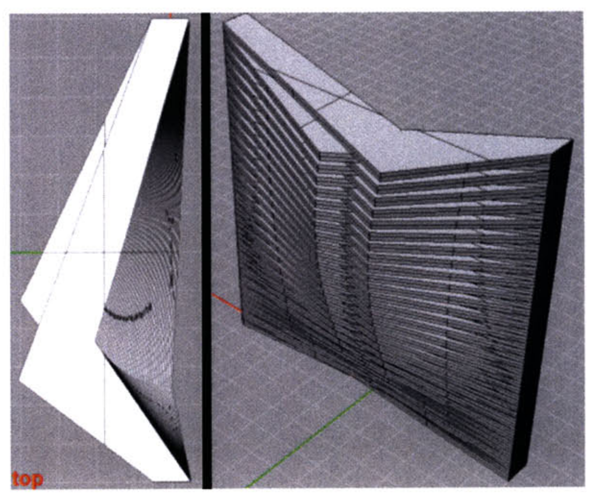
study of alternative design solutions of tiles for different ceiling geometries

2.1.3 From a Flat Piece to an Architectural Element

In a consequent step of the design process, the surface resulting from the flattening of the 3D object develops into a ceiling tile. Once again this is achieved by the manipulation of the code and by the control over the angle of rotation. The resultant surface is also scaled with a scale variable for the dimensional properties of a standard ceiling tile. The various transformations over the initial section result in various designs of the tile through which different geometries of architectural surfaces can be achieved.



parametric set-up of an architectural tile showing instantiations of the transformations of the initial shape, which determines the different sections of the tile and therefore its final design



final shape from the generative process of the tile which is the result of all the above transformations together

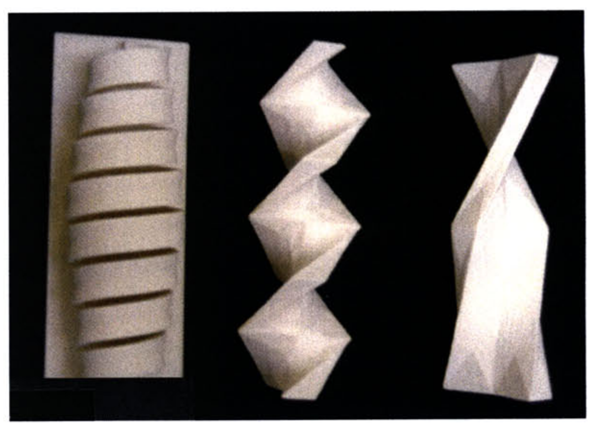
2.2 Digital Fabrication Experiments

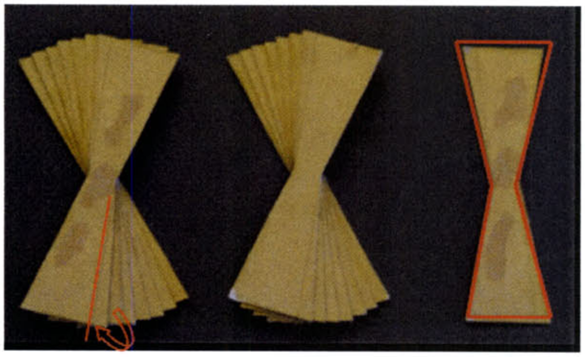
Among the Rapid Prototyping techniques available at MIT, two were most appropriate for this investigation: 3D Printing and Laser Cutting.
The following work is a series of experiments built with these two techniques to test with light the parametrically generated designs.

2.2.1 3D-printing

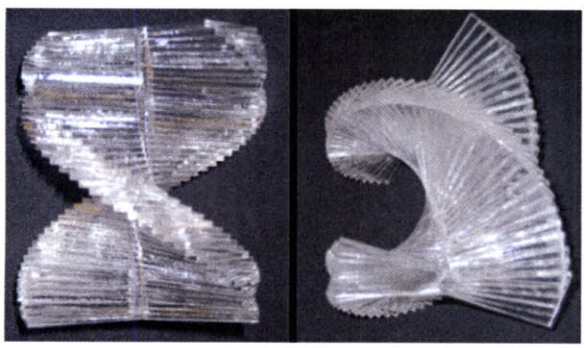
The first experiments in fabrication techniques are 3D models built in the 3D-printing machine. The 3D Printing technology, Z Corp.'s Z402 is a three-dimensional printer that can quickly produce 3D models from 3D CAD files, using a fine powder of starch hardened with a binding agent for material. These models are physical prototypes of the first objects that were drawn in Rhinoceros. They provide the opportunity to touch the design and to examine it more completely in terms of shape and proportion. They serve as a vehicle for visualization, or 3D sketches that help understand and analyze the geometry of the shape and by the same way improve it. With flexible strips of LEDs, these 3D-prints also help the thinking of the location of light sources into the object and are useful to analyze the design that best fits light sources and that presents a potential for a lighting system.

plaster models of 3D objects produced in Rhino, Z-Corp 3D printer

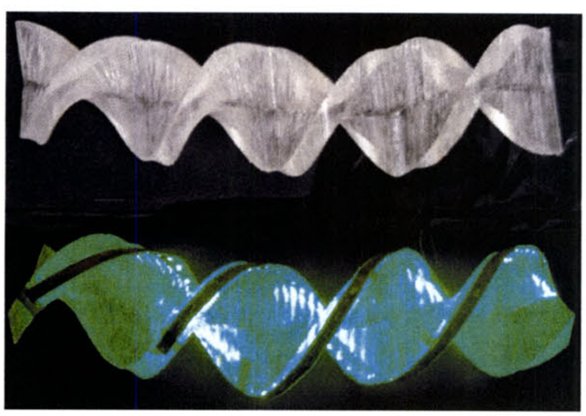




paper model of the 3D object, laser-cutter



acrylic models of 3D objects with different thicknesses of acrylic sheets, laser-cutter



acrylic model of the first parametric 3D object tested with a flexible LED strip along the edges of the helix

2.2.2 Laser cutting

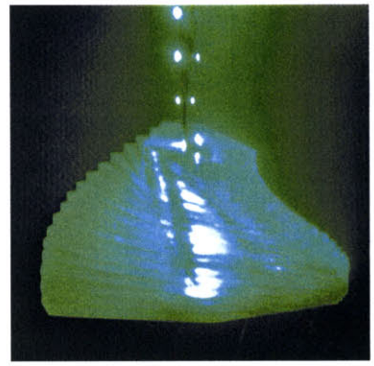
After the analysis of the shape with 3D printed models, the best performing objects are selected and more advanced models are produced in the laser-cutter using clear acrylic as a production material.

In this technique, slices of acrylic are cut and assembled together to form the 3D shapes, and strips of LEDs are placed inside or along the physical models.
This fabrication method evaluates the design in different aspects based on

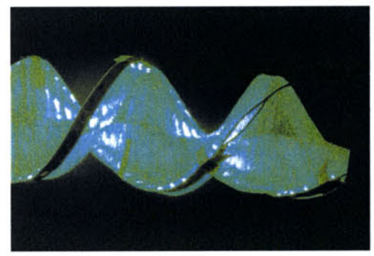
(1) the thickness of each piece of the material in relation to the angle of rotation.
(2) the location of the light source in the object.
(3) the surface treatment of the material used.
(4) the assembly system of the individual pieces that generate the whole shape of the model to test, based on the scripting code.

The focus of the design evaluation is the scattering of the light in the acrylic pieces in order to achieve the best brightness of the object. The first model built in this technique is a Rhino 3D object and is cut in paper. Like the digital model of the code, it is manually assembled by layering up a number of cut sections of the object, each time with an incremental rotation. This method of fabrication gives more freedom for the manipulation of the angle of rotation and consequently of the 3D shape. Indeed, the larger the angle of rotation on the sections, the flatter the object becomes.
Different variations in the design of the 3D object are produced with different thicknesses of acrylic and different angles of rotation, and are tested with LED strips.

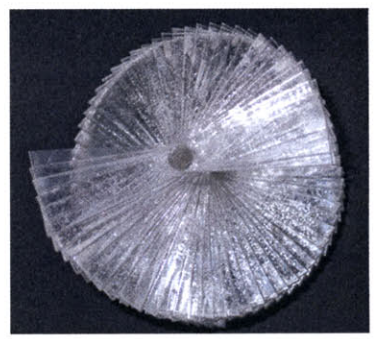
The study of these models for the design evaluation of the 3D object informs the proportional relation of the thickness of the original section to the angle of rotation: as dictated by the code, the thicker the section the bigger the angle of rotation. The performance of these models with light completes the test by observing a stronger brightness of the object with a larger angle of rotation, thus a flatter object.



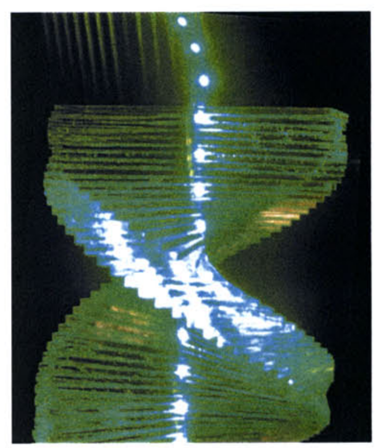
2.2.3 Light sources

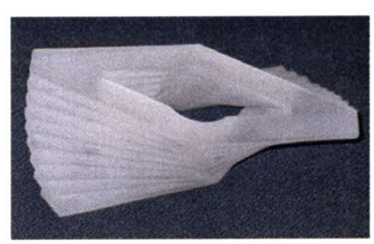


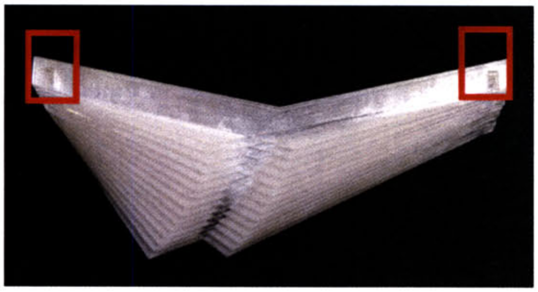
The acrylic models combined with LED strips allow for the testing of the light performance, aesthetically in relation to the design and functionally in terms of brightness. Although this is an empirical experiment, it allows to evaluate how light appears in the models and how it is affected by the different finishes of the acrylic pieces, and thus it guides the design decisions to optimize the brightness of the objects. Moreover, different locations and aiming angles for the LEDs within the models are analyzed and evaluated.



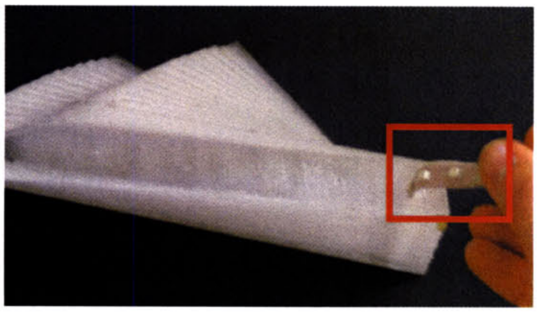
First, the LED strip is tested along the edges of the helix. The limitation of this location was a step in the development of the design and the production of new models with apertures in the center of the 3D object, in which the LED strip can be introduced. Different possibilities and shapes are investigated with the light source located in the middle of the 3D object. Some cases fit more than one strip of light, and accommodate different orientations of the light source into the object. It is observed that when the light source is in the center of the object, this latter does not achieve a good performance as it looks glary in some locations.

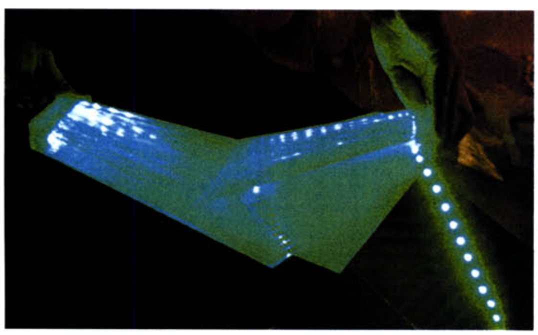


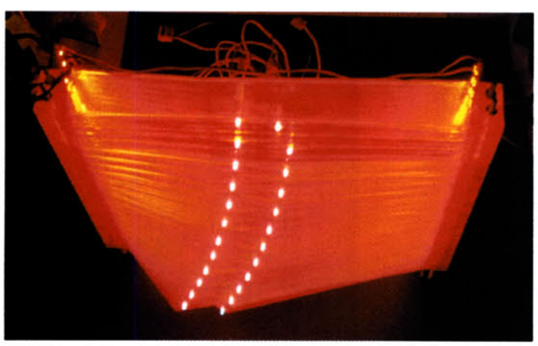




location of the LED strips in slots along the edges of the acrylic tile







two staggered strips of LEDs located along each side of the tile, while individual LEDs are introduced into the acrylic slices along the tips of the angular slices

With the control of the rotation angle, the result is a flatter piece, and the test of the performance for the location of the light source develops into the design of an edge-lit architectural element, namely a ceiling tile.

In the tile, the light source is positioned inside linear slots located along the sides of the tile. Two staggered rigid LED strips are introduced into the slot on each side of the tile facing the inside. The staggering of the LED strips reduces the dark areas caused by the distance between the LEDs in a strip.

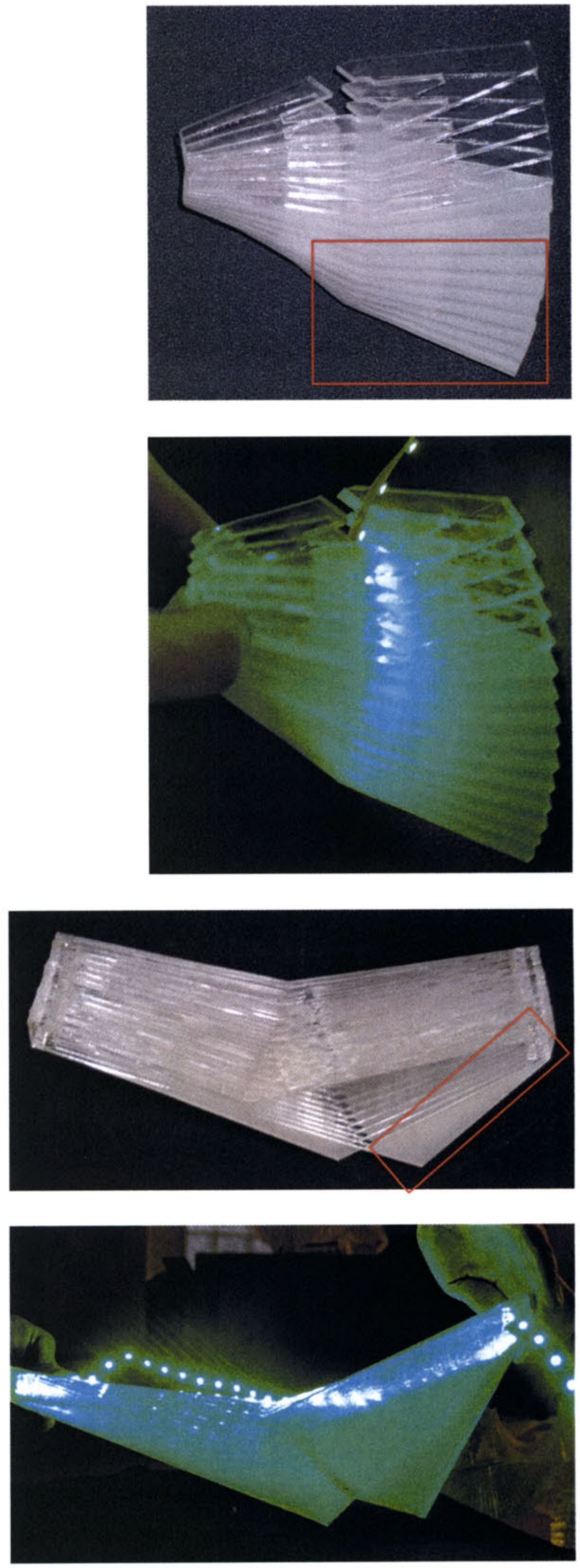
Prototypes of the architectural tile are also evaluated for the effects of light which developed with the use of different sources. In addition to the LED strips inserted into the linear slots along the side edges of the tile, individual white Light Emitting Diodes are incorporated between the double layers of acrylic pieces for illumination. A hybrid system with a dual lighting configuration is now created: it is a diffuse and direct lighting component. The system is at the same time luminous and illuminating.

The distribution of individual Light Emitting Diodes along the pieces of acrylic is another criterion for the evaluation of the light performance. Different prototypes of the hybrid systems are produced to test different patterns in the distribution of individual lights along the edges of the tile, or in the middle emphasizing the lines of the geometry along the protruded angles of the system, or both.

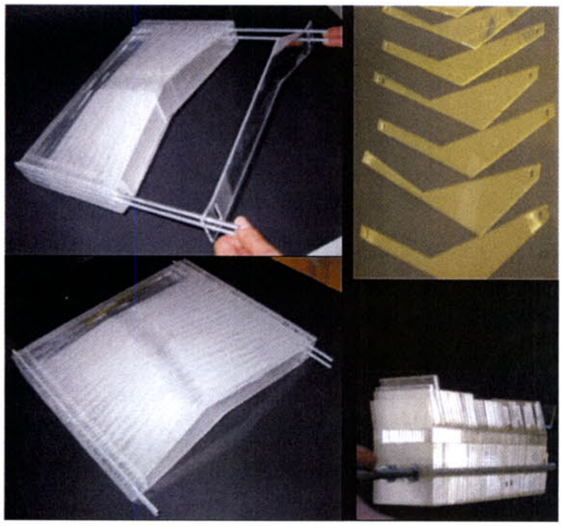
2.2.4 Treatment of Surface

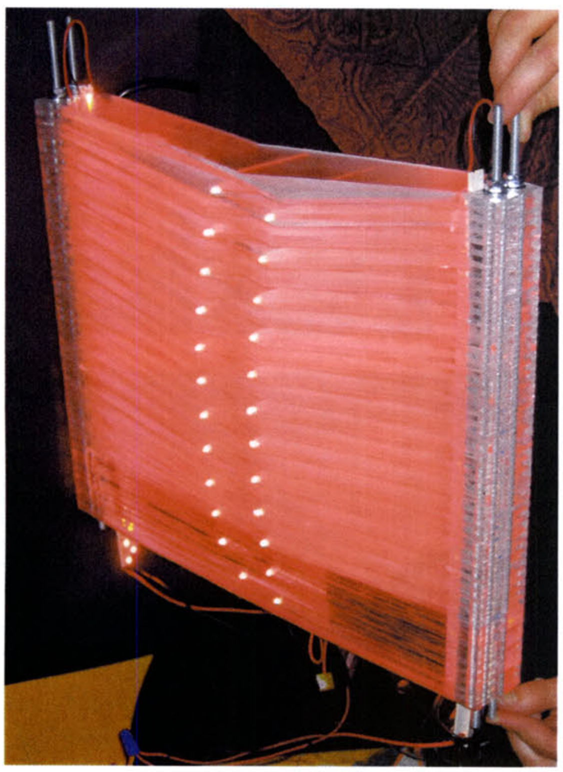
The construction of physical models for the evaluation of the performance creates the possibility to treat the surface of the material used in order to optimize the performance of light through the system. The surfaces of the acrylic pieces are mechanically treated to be translucent as opposed to being transparent. This treatment is first tested on the entire surfaces of each sectional piece of the model, then solely on the edge. It is observed that when light travels through a 'sand blasted' treated surface of acrylic, it is scattered and it propagates in a more efficient way for the brightness of the model. This conclusion engages a new investigation in the treatment of specific areas of the acrylic surfaces in order to have the light scattered only where needed, in this case, in the exposed edges of the object.

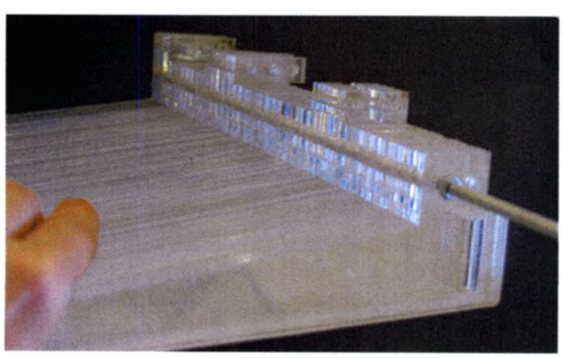
The edges along the face of the tile and the surfaces of the protruded angles of the different sections that form the tile are sand blasted while the inside is left clear with no treatment. Light travels freely through the clear inside of the pieces of acrylic, and it scatters in the treated areas which then look brighter; and the object becomes luminous.



treated areas of the models are marked by a red rectangle







2.2.5 Assembly Systems

Like the 3D object, the tile is constructed with individual pieces that are first glued together with acrylic glue. The existence of the glue between the acrylic pieces disturbed the performance of light as it obstructed the light rays to travel through. A mechanical method of assembly is then used to put the pieces of the tile together. This method consists of two metallic rods that go on each end of the tile behind the light source location to hold the pieces together in compression. Considerations of the tile as a module in a larger system of luminous surfaces drove the conclusion of moving the assembly system of the rods to the back of the tile. The ends of the tile are free to receive adjacent ones. The assembly system is then developed to connect adjacent tiles together while keeping room for the channeling of electrical wires from the individual LEDs that are embedded in the acrylic slices along the edges of the tile.
The assembly of the pieces with rods gives more flexibility for the manipulation of each individual piece in such a way that makes the physical prototype to behave parametrically. Indeed, it offers the possibility to change the order of the pieces, alter the design and the treatment of each piece separately, and consequently change the design and the performance of the whole system.

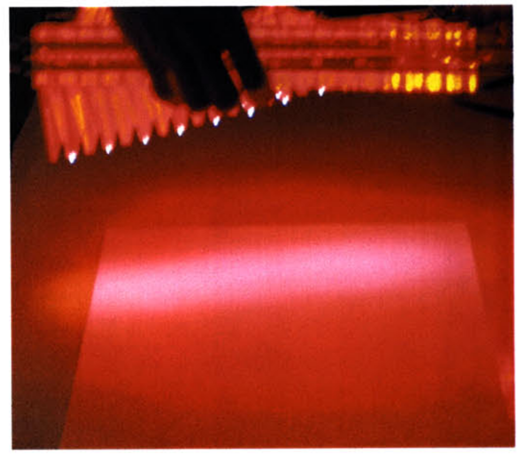
2.3 Evaluation

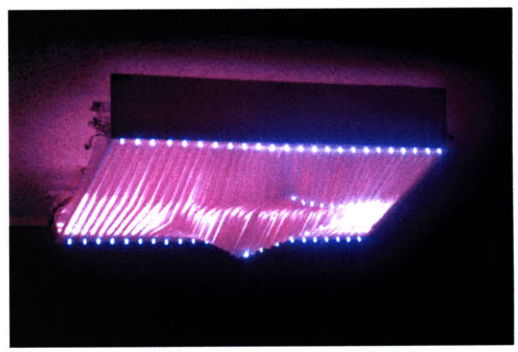
The use of Rapid Prototyping technology in the design of lighting systems initiates implications on the strong potentials of its applications in lighting design.

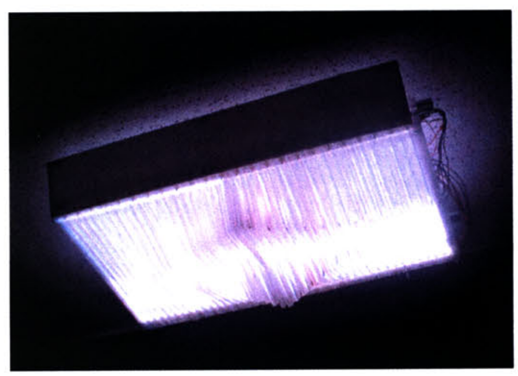
The previous experiments in fabrication evaluate the design in a series of variations achieved through the code. The construction and the observation of these variations highlights design factors that affect the visual and aesthetic performance of the architectural tile, such as the angle of the individual sections in the tile.

The physical prototypes of the architectural tiles demonstrate that the angle in the slices of acrylic benefits the brightness of the tile. The tile looks brighter when it is built with angular slices as opposed to straight pieces of acrylic. The angle also adds to the aesthetics of the tile as it creates a motif in the tile and draws different patterns. When such patterns are created, the tiles can be composed in symmetrical pairs to form a luminous surface.

This is shown by the final physical prototype of luminous tiles that was successfully mounted on a ceiling of a space. The pair of tiles embedded strips of color-mix LEDs. It was the first model that enabled me to observe and analyze the performance of the tiles in a real setting. I was also able to observe the colors which performed better in the tiles resulting in a uniform brightness; and those which rendered the tiles with dark strips.







a pair of ceiling tiles with hybrid lighting and color-mix LEDs, installed on a ceiling

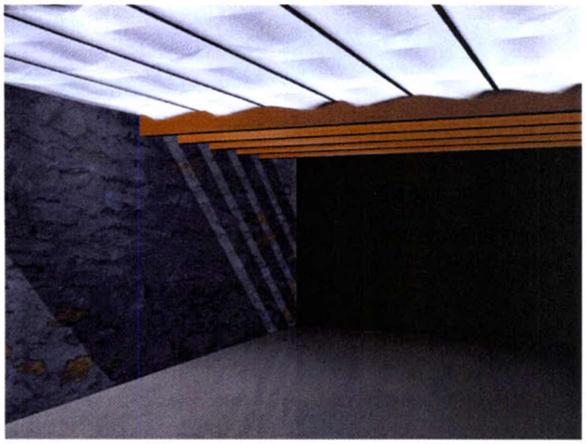
straight acrylic pieces/ edge sandblasted

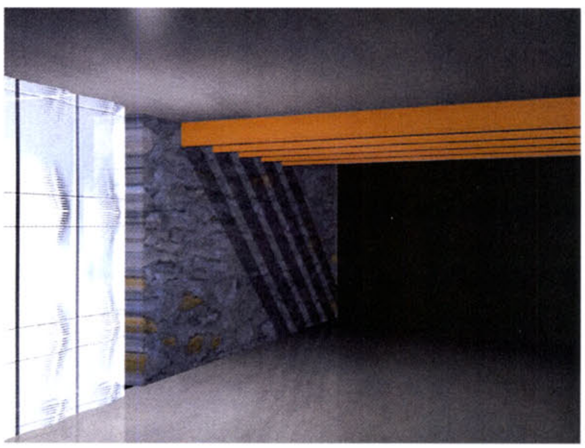
angular acrylic pieces/ whole sandblasted

angular acrylic pieces/ edge sandblasted

the visual comparison of the performance of three tiles with different geometries and treatment of the pieces, reveals the best performance with the angular pieces treated on the edges only

CHAPTER 3: CONSTRUCTION OF A LUMINOUS CEILING





simulations of luminous surfaces with parametrically generated tiles

The code used so far in this work treats individual tiles as self-contained and self-referential pieces. This phase of the research intends to further advance the code to generate a grid of tiles, in which the design of each individual module relates to its respective neighbor. The design variations of the tile that were so far generated by the code largely define all the possibilities of this code, and by the same means its limitations.

This phase develops the code to generate a ceiling system and to achieve an overall result of architectural surfaces rather than being restricted to individual elements. The idea is to attribute to the design an architectural scale by creating a modular system of tiles in the code.

From an individual system, the design is then focused on architectural surfaces to create different luminous configurations. The change in the geometry of the tiles changes the geometry in the ceiling and thus affects the space.

3.1 Design Rules

The design rules discussed here pursue the goal to develop the design of a luminous ceiling that, being parametric, could be modified to accommodate specific needs. For example, the code could be configured to generate the ceiling of a lobby, or the ceiling of an office...
This objective prompted the need to rationalize the design by extracting and understanding the rules that underlie it. Shape grammars [14] are used as a computational method that explains the design by producing a set of shape rules for luminous ceilings. The rules defining the grammars of tiles and ceilings are expressed in the code, and modeled in Rhinoceros, remaining parametric. The design of the tiles and the ceilings relies primarily on the outcomes of shape grammar and secondarily on the manipulation of the code.

3.1.1 Shape grammars

"*Grammar here is used in a formal, technical sense, a grammar being a particular type of mathematical construct. The word was first used in this technical sense by Chomsky (1957), who defined various types of formal grammars in his quest to find a mechanism that would generate exactly the set of 'grammatical' English sentences. Chomsky, of course, developed grammars that generate sequences ('strings') of symbols. Other types of grammars have since been developed that generate arrays, trees, graphs, or shapes. All of these types of grammars are examples of production systems.*"[15]

---

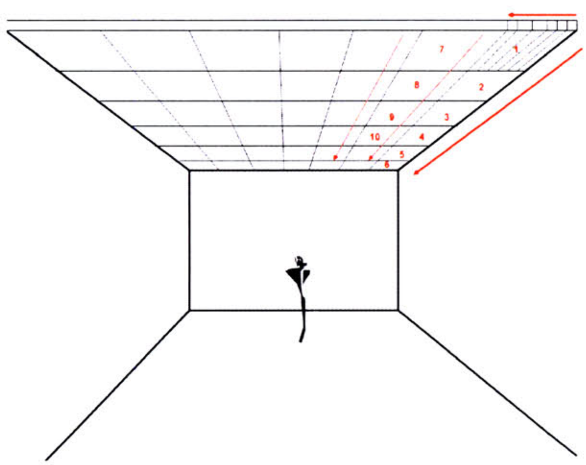
[14] Stiny, George, "Introduction to Shape and Shape Grammars", *Environment and Planning B: Planning and Design* 7 (Great Britain: Pion Publication, 1980), p. 343-351

[15] Gips J, Stiny G, "Production Systems and Grammars: a Uniform Characterization", Vol.7 of *Environment and Planning B* (1980), p.399-408

Many computational methods can be used as a structure of rules to understand a design and develop new ones. I chose to work with shape grammars to describe a language of luminous ceilings, which can generate new designs and which specifies the design solutions with aesthetic performance. The advantage in utilizing this method is in Terry Knight's words:

"... *for creating and understanding designs directly through computations with shapes, rather than indirectly through computations with text or symbols*."[16]

With a language of shape rules ceilings are produced in different configurations. The combination of shape rules with parameters that define the elements of a ceiling creates a systematic method with the possibility to tailor the design for different spaces.



understanding the rules through which a modular system of ceiling tiles can be built

James Gips, one of the inventors of Shape Grammars, defines a ground for the use of a computerized shape grammar model:

"*Shape grammars are intended to form a basis for purely visual computation. The primitives in shape grammars are shapes, rather than symbols. The relationships and operations are all spatial (e.g. similarity, rotation) rather than symbolic*".[17]

Since the design of the tile, and concurrently, of the ceiling is formalistically motivated and spatially driven, the application of Shape Grammars as a computational method with shapes helps both describe and generate the design. Once the designs are depicted with shape rules, these rules can be used to create more ideas and could become useful

---

16 http://www.mit.edu/%7Etknight/IJDC/frameset_abstract.htm, September 2000

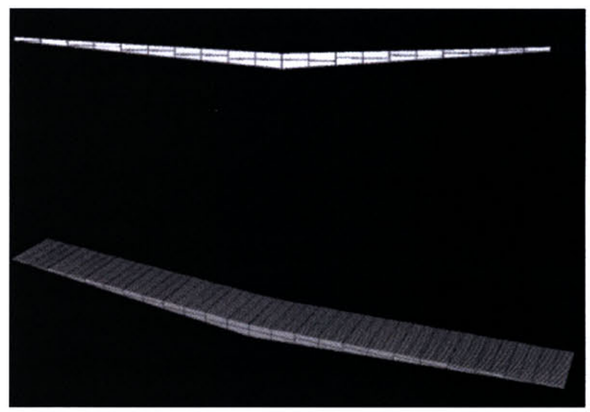
17 Gips, James, "Computer Implementation of Shape Grammars", Workshop on Shape Computation, MIT, 1999

tools for creativity as they help understand the nature of the computational design process.

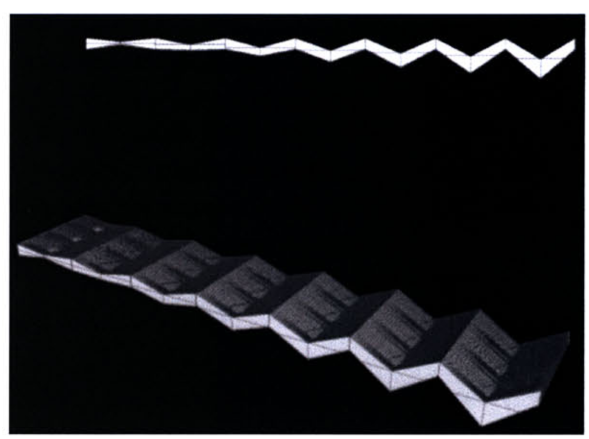
With these anticipated objectives, the definition of design rules using shape grammars is formulated as a means to:

1- Capture the expertise and the knowledge of a design and embed it in rules that can be transmitted and manipulated to produce variations.
2- Identify a method to explore design solutions for specific needs and spatial integration.
3- Define a grammar that maximizes the performance of luminous ceilings in terms of brightness and light distribution.
4- Allow a larger community of users to come up with their own designs of luminous ceilings by playing with shape rules.

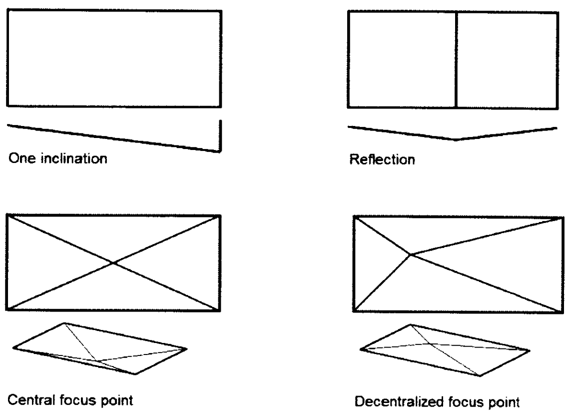
Through an independent study that I took with Professor James Gips (Spring 2004), I was able to develop different design rules and define shape grammars for different designs of luminous ceilings.
In a first step of the design investigation, the same code that was used to create a tile develops to generate two types of ceilings.
In the first ceiling type, the tiles describe one predominant movement. They all incline towards one main convergence point and most of the tiles are built out of straight slices, as opposed to the angular tiles previously discussed.
In the second ceiling type, the tiles describe a repetitive rhythmic movement, and thus they create a texture in the ceiling. They are built out of angular slices.



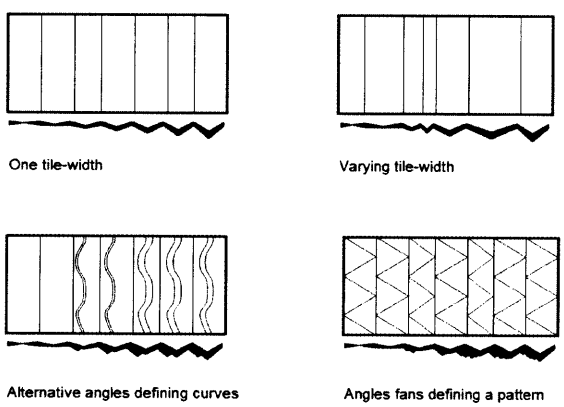
ceiling type 1 describing one predominant movement



ceiling type 2 creating a texture, or a rhythm



variations of ceiling type 1



variations of ceiling type 2

Each ceiling type was a platform for the exploration of possible rules to propose a family of design variations. The variations in the first type of ceiling are:

a. A reflection: the ceiling folds symmetrically creating a sense of balance in space.
b. An inclination in one direction: the ceiling is sloped and guides the vision to one end or one wall of a room.
c. A central focus point: the ceiling is a symmetrical pyramid with a focus point emphasizing the central space of a room.
d. A decentralized focus point: the ceiling is an asymmetrical pyramid with a focus point shifted to designate or direct the attention through light to an important location or feature in a room.

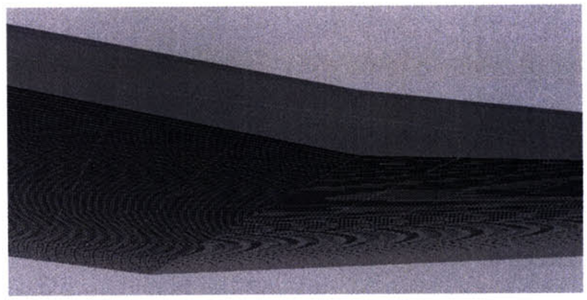
The variations in the second type of ceiling offer greater design possibilities as they are controlled by both the rhythm and the movement of the tiles and those of the slices in each tile. The result of these variations produces intricate motifs and patterns formed by the manipulation of the central angle of the slices.
The pre-defined variations in this type are:

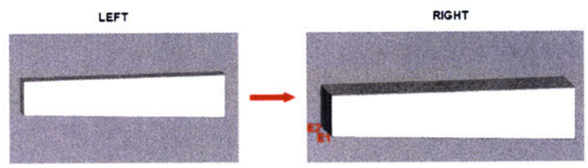
a. A pattern of curves: the ceiling is a composition of parallel curves formed in the center of the tiles.
b. A pattern of lines: the ceiling is a composition of triangular motifs formed in the center of the tiles.

In a second step, a shape grammar is defined to depict the rules underlying each ceiling design.
Three different grammars are presented in the first type of ceiling.

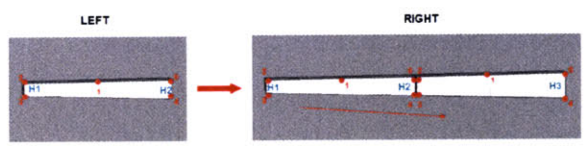
Type One Variation a:

The first design is the 'one movement' ceiling with a reflection. It is based on a five-rule grammar:
Rule 1 uses the shape operation of addition to create a tile. This rule takes the initial shape, which is the first slice of a tile, and repeats it identically a number of times until it generates a tile, which is one module of the whole ceiling surface. The spatial transformation in this rule is a translation.
Rule 2 copies the initial shape, which is the first slice of the first tile, stretches its height in one end and adjacently adds it to the initial shape. Then, applying Rule 1 again, it repeats the new shape identically a number of times until it generates the second tile. These two rules are repeated alternatively until they create a row of tiles, in which each tile is stretched by an increment added to the height of the previous slice of tile.
Rule 3 copies the first slice of the last tile generated by Rules 1 and 2, mirrors it and adds it adjacently to the previous one; the mirrored section is repeated by Rule 1 to create a mirror of the last tile.
This is the beginning of the reflected half of the ceiling, which is built in the same fashion with a fourth rule that decreases the height at one end of the tile by the same decreasing values of the increment added in Rule 2.
After building one row of the entire ceiling with these four rules, Rule 5 repeats this row a number of times to generate the whole ceiling surface.



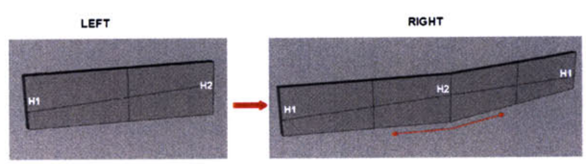


rule 1
parameter E1 = E2 = ... Where E is the extrusion of the shape

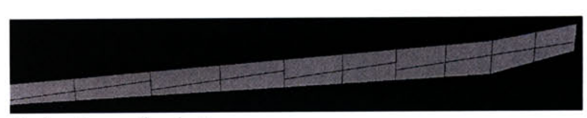


rule 2
parameter H (a+1) = (x/(3-3x)) H (a) where x = 2 to tilecount (number of tiles)

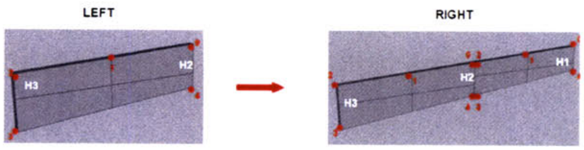
derivation of rule 2



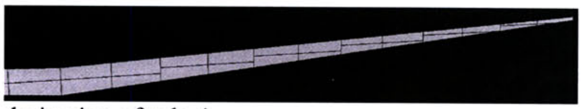
rule 3
reflection of the first slice of the last tile



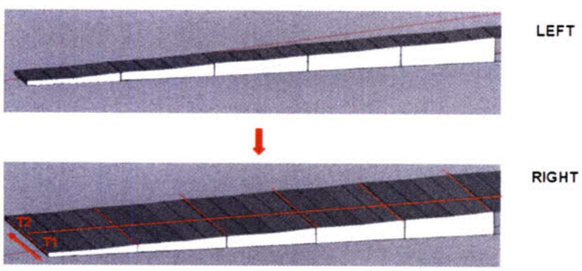
derivation of rule 3



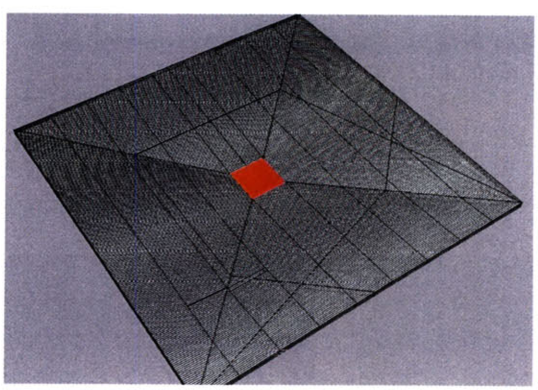
rule 4
parameter H (a) = (x/(3-3x)) H (a+1) where x = 2 to tilecount



derivation of rule 4



rule 5
parameter T1 = T2 … where T is the tile width



Type One Variation b:

The grammar of this design is defined by Rules 1, 2 and 5 of the previous grammar. These three rules build the inclination of the ceiling in one plan without doing any reflection.

Type One Variation c:

The grammars defined for the previous two designs describe linear applications of rules. The following design is based on a grammar that applies shape rules in a parallel way to generate the design of a pyramid ceiling. As defined by Terry Knight[18]:

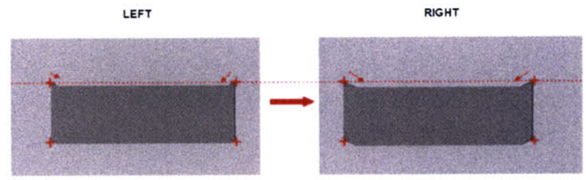
"Parallel grammars allow different properties and representations of designs to be separated into different computations, while allowing for these computations to communicate with and influence one another in appropriate ways".

The rational to define a grammar for this ceiling considers the possibility of the focus point or the tip of the pyramid to be located in and off the center of the ceiling, and thus to allow design variations according to needs. Therefore, the rules start from the central piece of the pyramid and develop towards the edges, and identify the design process as radial, growing from the center outward.

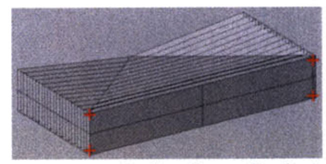
The first three rules of this grammar are addition rules. They add slices to create the central tile of the pyramid. Rule 1 repeats and adds the initial shape and applies a transformation on the central corners of the shape. The transformation is a translation of the corners toward the center of the shape until the two corners meet in one central point. Registration marks show the fixed ends of the slices.

[18] Knight, Terry, "Shape Grammars in Education and Practice: History and Prospects", Vol.2 of *International Journal of Design Computing*, 1999-2000

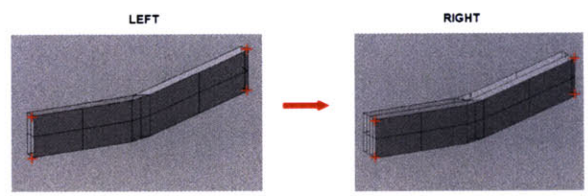
Rule 2 copies the last slice from Rule 1, reflects it and adds it to the previous one. Rule 3 repeats and adds the reflected slice like Rule 1 this time applying the transformation to translate the central corners from the center toward the edges of the slice. These three rules build the first tile of the ceiling which is the tip of the pyramid.



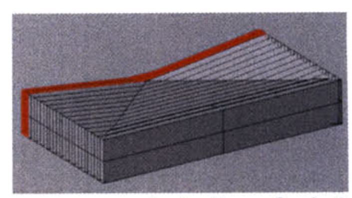
rule 1



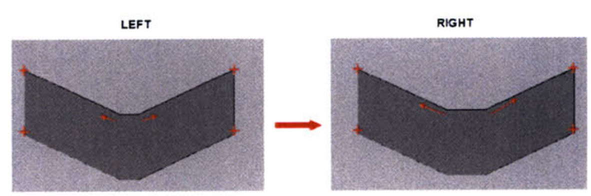
derivation of rule 1



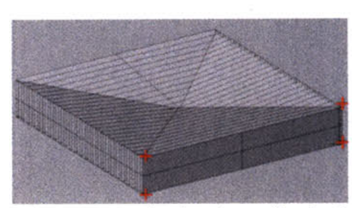
rule 2



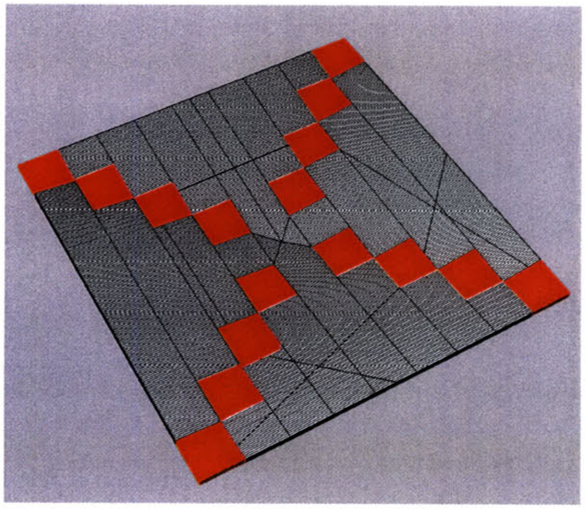
derivation of rule 2



rule 3



derivation of rule 3

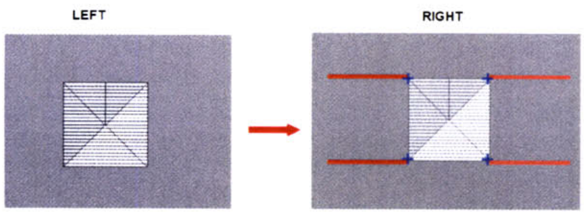


The following rules are applied in a parallel way to build the tiles at the bended edges of the pyramid. These edges connect the tip of the pyramid to the corners of the ceiling and they define the 'spine' of the pyramid.

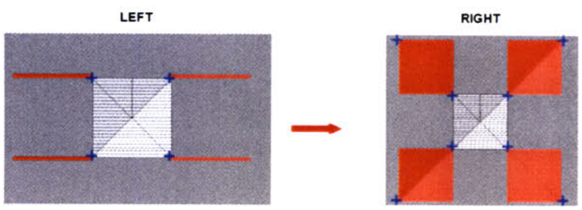
Rule 4 adds to the corners of the tip tile four slices which are the initial shapes of the four tiles to be generated.

Rule 5 repeats and adds these initial shapes a number of times to build the tiles along the edges of the pyramid. These slices are repeated with a transformation that stretches and translates the central corners to create a bend in the middle of the tile.

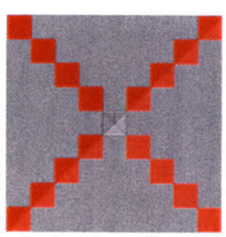
Rule 6 builds the faces tiles of the pyramid. These tiles are formed by straight slices that add and move through a translation downward to create the slope of the pyramid, meaning the depth of the ceiling.



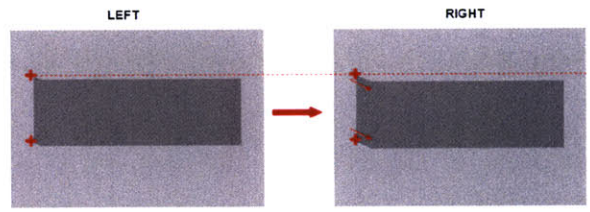
<u>rule 4</u>
rules applied in parallel



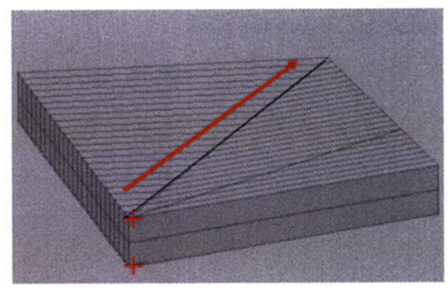
<u>rule 5</u>
rules applied in parallel

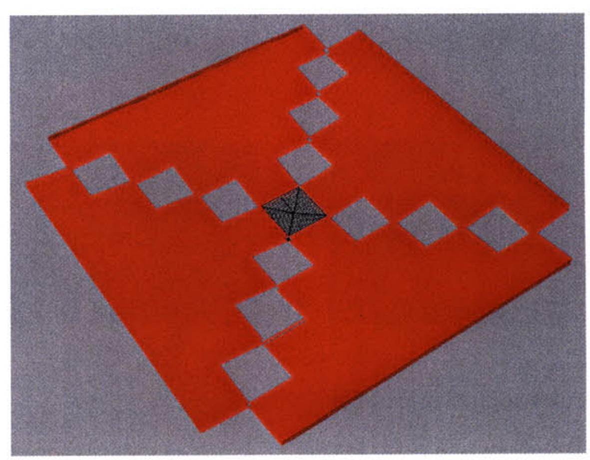


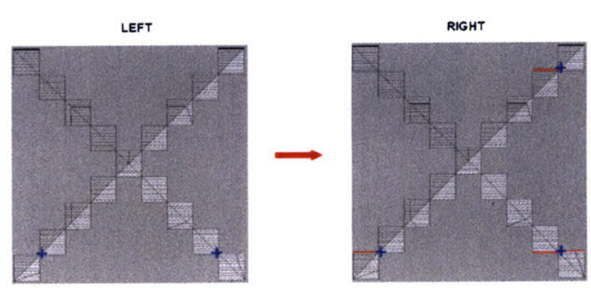
derivation of rule 5



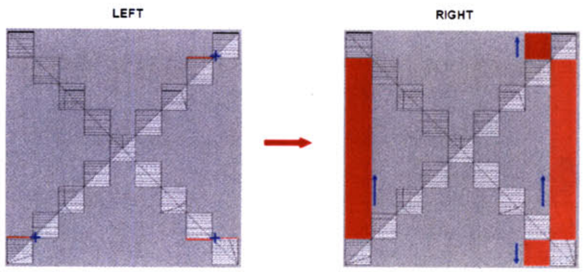
rule 5
rules applied in parallel



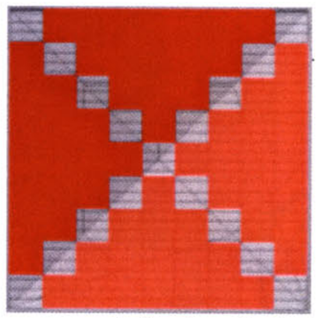




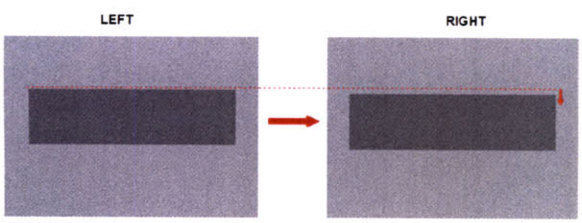
rule 6
rules applied in parallel



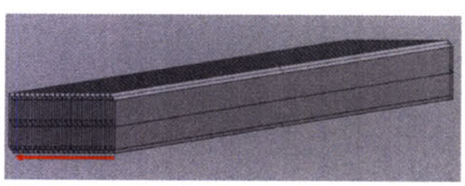
rule 6
rules applied in parallel



derivation of rule 6



rule 6
rules applied in parallel



The two grammars that describe two different designs in the second type of ceiling are also based on rules of addition that apply different transformations on the slices to create variations in the tiles and thus a texture in the ceiling.

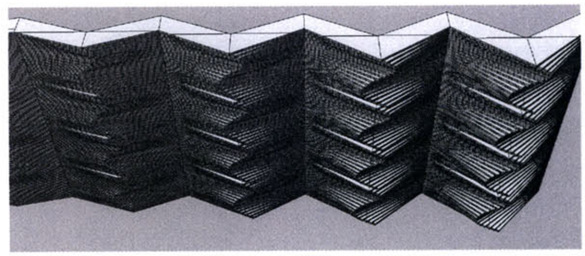
Type Two Variation a:

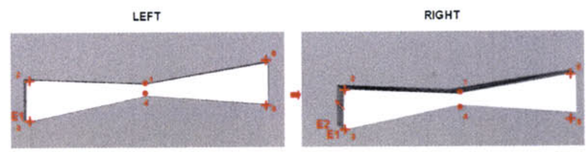
This design is based on a three-rule grammar.

Rule 1 repeats and adds the initial shape to create a tile. As it repeats the shape, it applies a transformation on the central bottom corner of the shape which makes it rotate and describe a curve through the overall number of slices in one tile.

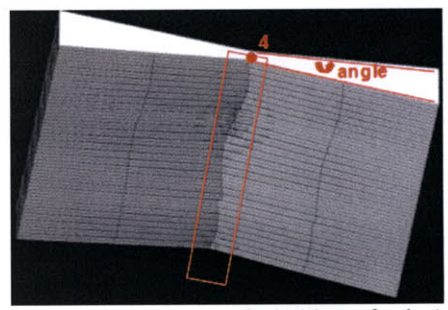
Rule 2 repeats the initial shape and transforms it by adding an increment to the central height of the shape. The alternative repetition of Rules 1 and 2 creates a row of tiles that gradually increase in central height or depth while describing curves in their central bottom corners.

Rule 3 repeats and adds the row of tiles to create a surface.

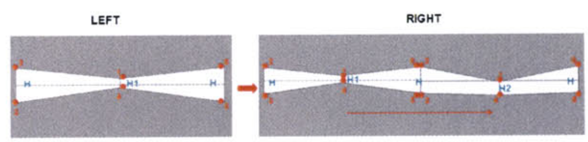




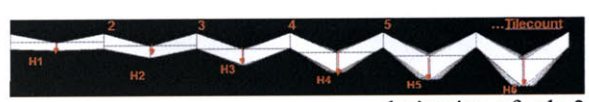
rule 1



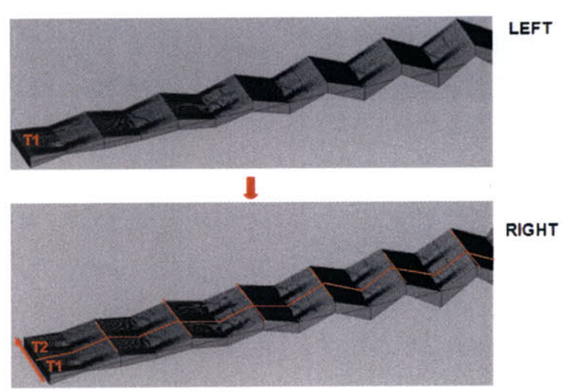
derivation of rule 1



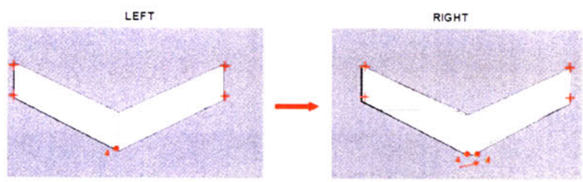
rule 2



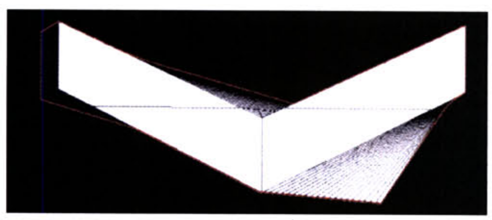
derivation of rule 2



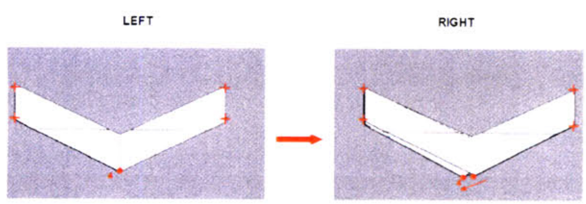
rule 3



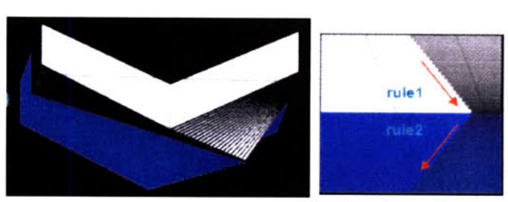
rule 1



derivation of rule 1



rule 2



derivation of rule 2

Type Two Variation b:

The grammar of this design is similar to that of the previous one.

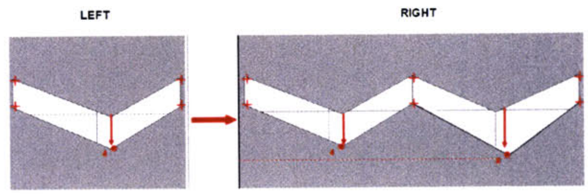
Rule 1 repeats and adds the initial shape to create a tile, and translates the central corner of the shape by an increment to create a straight line.

Rule 2 copies the last slice of the generated tile and applies a translation to the central corner of the tile in a symmetrical direction of the line defined by the corner in the first tile; and adds it to the previous slices of the tile.

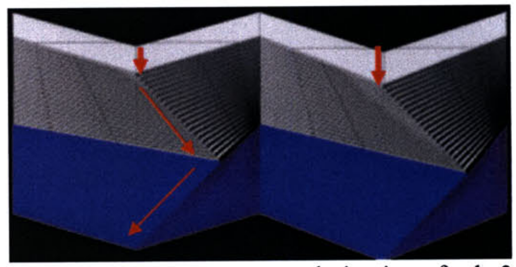
Rule 3 repeats the first slice of the first tile and transforms it by adding an increment to the central height of the slice. Then Rules 1 and 2 are repeated consecutively to create two tiles adjacent to the previous ones.

Rule 4 repeats and adds the row of tiles to create a surface.

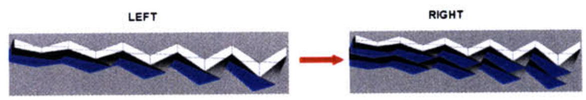
The definition of a shape grammar for each type of ceiling offers many choices of rules and many ways to apply them for multiple design possibilities. It also brings up new design properties and spatial configurations such as the direction of the rows in a room, the depth of the ceiling, the number of tiles and rows in relation to the dimensions of the room in which the ceiling is installed.



rule 3



derivation of rule 3



rule 4

derivation of rule 4

3.1.2 Visual Basic with Shape Grammars

After the grammars are defined and the rules underlying the designs are depicted, the codes become a parametric expression of the grammars; and a tool of interaction with the grammars.

The rules are controlled by the loops that repeat the slices and build the tiles in a specified sequence. This sequence is determined by the functions that draw the initial shapes and that are packaged to call the rules one after the other. The transformations that happen within the rules are defined by the parameters that are declared in the code.

This organization of the design in parameters and rules draws two main conclusions for the generation of new designs:

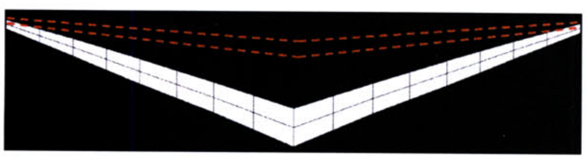
1- Iterations through the functions:

Variations in the design can be created by the manipulation of the parameters or the variables defining the initial shape in the functions. They can either conserve the topology and change the proportion or depth of the ceiling; or change partially or entirely the design.

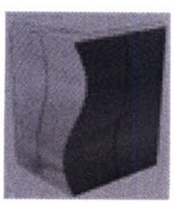
The different motifs of the texture ceiling are an example of this last variation. The parameter that first created a curve in the middle of the tiles along the rows is modified to draw straight lines, continuous or interrupted, each time resulting in a new geometry of the ceiling. In the code, this parameter is either expressed through a cosine function that draws the curve, or through an increment related to the number of slices in a tile to draw a straight or inclined line.

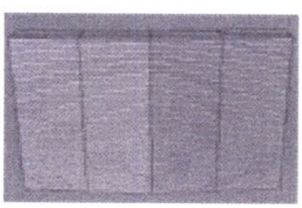
2- Iteration through the rules:

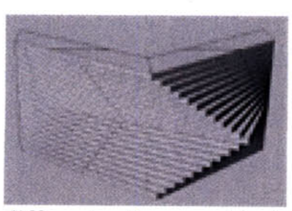
Variations in the design can in this case be created by the choice of rules and the sequence of rules.



variations created by the manipulation of the parameters or the initial shapes of the rules

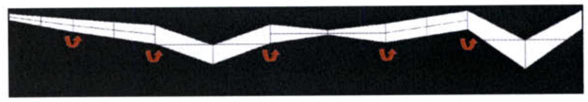






different patterns and textures created in the tiles

The choice of rules happens in the loops that build straight tiles, bended tiles or angular tiles with controlled heights as identified in the previous original designs. Different loops can be called by different functions and vary the movement in the geometry of the ceiling from the initial shape assigned by the used function.
Since different rules can be applied, different sequences or orders for these rules can be determined to generate a new design. For example, if the first two rules of the first ceiling type are called by a function, they will repeat the initial shape defined by this function and stretch one end of the shape to make it inclined. Then, instead of calling the third rule of this grammar, the second rule of the second grammar is called to increment the central height of the created tile; a hybrid topology of the ceiling can be generated and can identify a new design which was not preconceived.



design variations created by the choice and the sequence of rules used from different grammars

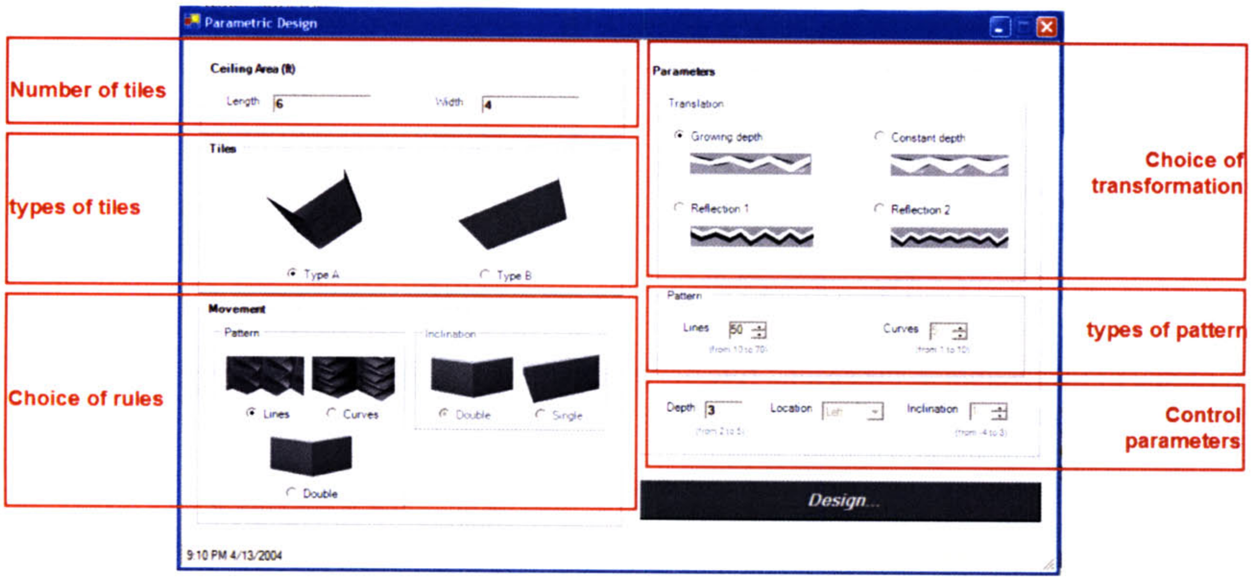
3.2 Control Mechanisms

The scripts and the shape rules built for the design of luminous ceilings are developed into two types of control mechanisms defined in an input interface used in Rhinoceros. These two types take a user input to execute a design and achieve unpredictable results by responding to different parameters or by applying different design rules through computational iterations.

3.2.1 Input Interface

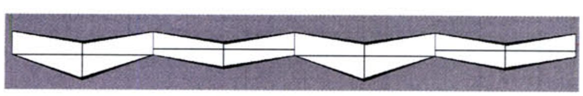
The interface suggests an approach to connecting different grammars together. It is used as an input to run the shape grammars.
It gives control over the pre-defined computational processes in a structured way. The control happens through the input of constraints and parameters, and through the choice of design rules.

It is built in five sequential steps and offers the choice between two main lines of design: a design with straight tiles, or ceiling type one; and a design with angular tiles, or ceiling type 2.

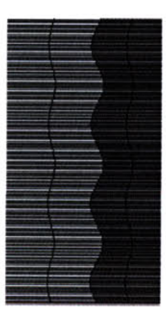


angular type of tile

straight type of tile

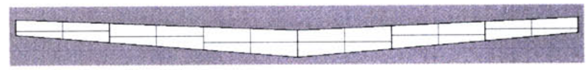


the *Pattern* grammar varies the geometry of the individual tile, consequently the geometry of the ceiling and creates patterns with *Lines* and *Curves*

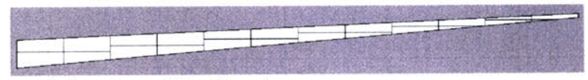


1- CEILING-AREA:
   In a first step of the interface, the values input are the *Length* and the *Width* of the ceiling, which determine the number of tiles in the ceiling.
2- TILES:
   In a second step, the user chooses the *Type of Tiles* for the design of the ceiling. The two proposed types are the angular tile that creates a texture in a ceiling, and the straight tile that emphasizes the inclination of a ceiling.
3- MOVEMENT:
   the movement is an expression of two main grammars for the ceiling: the *Pattern* grammar allows for different motifs to appear in the ceiling, and thus different textures of lines and curves. The *Inclination*

grammar inclines the ceiling either with a fold in the middle of the ceiling, and thus it creates a 'double' inclination, or at once along the entire ceiling, which becomes a 'single' inclination.



fold type of *Inclination*



straight type of *Inclination*

4- PARAMETERS:

The parameters first assign the type of translation in the tiles of the ceiling and by doing so; they give the user the choice of the shape rules that control the specified type of ceiling.

The *Growing Depth* translation increments the depth of the tiles as it repeats them in a row. The *Constant Depth* translation repeats the tiles with the same depth. The *Reflection 1* translation allows the tiles to grow deeper in the center of the ceiling, while the *Reflection 2* translation makes the tiles shallower in the center of the ceiling, and the edges higher in depth.

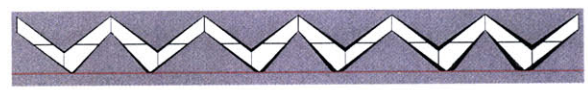
Once the choice of the rules is done, the density of the patterns in the ceiling can be input for the *Lines* type and the *Curves* type. This means that a pattern can be repeated in each tile in a modular composition, or can occur on the totality of the ceiling. Variations happen between these two extremes and a multitude of designs motifs can result out of the manipulation of this parameter.

The *Depth* parameter is also input in the interface.

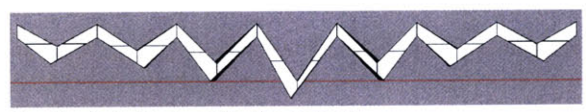
The *Location* of the pattern can also be determined either on the left side or the middle or the right side of the tile.

The *Inclination* input determines the angle of the inclined ceiling.

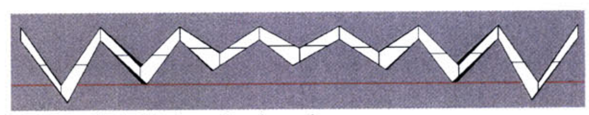
the depth of the tiles grows progressively along the ceiling



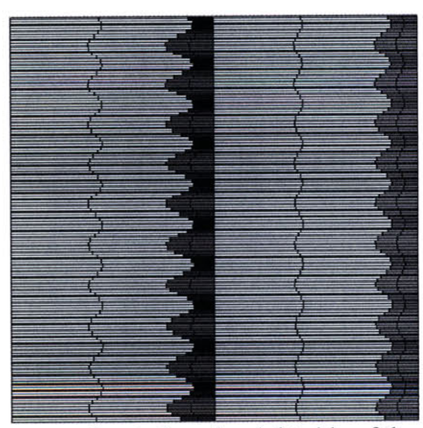
the depth is constant



the depth is highest in the center



the depth is highest in the edges



pattern located on the right side of the tiles

The changing of one of these input values results in a variation of the design.
(Examples of these variations are shown in appendix 1)

3.2.2 Control of Parameters

The process of building the input interface starts with the extraction of control parameters in the RhinoScripts. Since the geometry of the ceilings is numerically specified in the scripts, the numbers that determine all parameters expressed in the Input Interface are defined on top of each script as numeric input to be controlled through the interface. The code then becomes a drafting tool following the rules that have been defined and a structure to interface with the user.
The extracted parameters operate within each grammar of ceiling and give the user the freedom to vary the proportions of a ceiling and its design configuration. They are carried in the rules that define each type of ceiling, and thus the combination of different rules gives a control over the parameters of different types of ceilings and increases the design possibilities.

3.2.3 Control of Rules

The control of rules is another mechanism in the interface that gives the possibility for the user to generate new designs through a recursive computation of shapes and with a variable sequence. This control generates designs using shape operations and transformations. The interface offers the choice of the initial shape, the type of the ceiling tile, and the selected type can then be applied in different rules that are selected in the types of movement or the typology of the ceiling; and in the translations which result in different design transformations. The types of tiles (initial shapes) and the types of movement (shape rules) can be

cross-selected and give a large number of design alternatives.
The use of shape grammar as a control mechanism brings contribution not only to the fact that new designs can be conceived, but also to the objective of the design investigation to maximize the performance of luminous ceilings. Indeed, by defining the shape grammars for different designs of ceilings and testing the designs, it becomes possible to identify the rules that carry the design conditions for a better performance, and thus the user can work only with rules that lead to well performing designs.

3.3 Evaluation

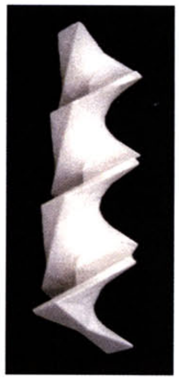
The input interface exemplifies the role of shape grammars to extract the rational of a design and to convey a method for designers to operate with rules and to make possible a series of design solutions.
The control of parameters and rules in the codes and shape grammars offer a series of design possibilities and considerations.

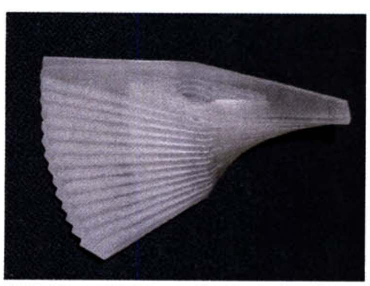
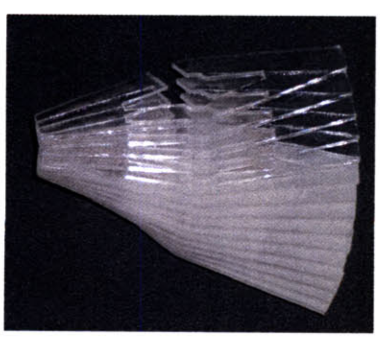
Four types of translations are possible in ceiling tiles. The translations depend on the *depth* parameter, which is observed as the height of the ceiling. Thus, instead of having one depth in all the tiles of a ceiling, the tiles can have a linear growth of depth in one direction. A gradual increase of the individual tiles depths determines an inclination of the ceiling.
In the other two types of translation, the depth of the tiles grows symmetrically from the edges to the center of the ceiling, or vice versa.
The *depth* parameter varies between two and five inches, which gives four levels of variations in a ceiling height. These bounds were fixed after the models were tested. During evaluation the models showed that an inappropriate depth results in a reduction of the brightness.

Other parameters, such as the *Lines* and the *Curves* have control over the density of a pattern in a ceiling, which in turn changes the whole configuration of the ceiling. Hence, the *Lines* parameter varies between 10 and 70 from denser to larger patterns. The largest value (70) covers the entire ceiling with one module of the pattern and the tiles are different and are composed together to make this module along the ceiling. The smallest value of this parameter (10) builds four modules of the pattern in one tile and thus makes it look very dense along the ceiling and creates a richer texture in the ceiling. Between these two extreme values, sixty-one different configurations are possible for the ceiling. The design results of some intermediate values for this parameter are unpredictable and interestingly different from the original design and pattern.

The *Curves* parameter varies between 1 and 10 from one curve module per tile to ten modules per tile, in which case the density of the pattern grows with the value of the parameter. Ten different design possibilities can be generated out of the manipulation of this parameter.

These numerous derivations from the pattern parameters can vary and multiply through their combination with the types of translation, and with the *Location* parameter that can move the pattern to the left or to the right of the tile.

CHAPTER 4: CONCLUSION

The use of parametric tools and digital fabrication technology demonstrates a strong potential for their applications in lighting design, which greatly benefits from the testing of the design and the technical performance before implementation.
Parametric methods of design produce variations in the geometry of the luminous architectural tiles and that of the luminous ceilings. The variations are the results of the manipulation of geometric parameters and physical variables evaluated in a Visual Basic code, which is used as a scripting language for modeling in Rhinoceros.
These variations allow for the testing of the design in relation to the brightness of the tile, or the system of tiles through the complementary use of rapid prototyping techniques. Thus, generated designs are prototyped in physical models and are equipped with linear light sources (LED strips) placed along the side edges of the tile, resulting in a luminous architectural element.

The work of this research represents an unprecedented example of the use of cutting edge computational tools for the design of lighting equipment, and proposes an exploration path for designing luminous architectural elements with embedded LEDs. It has completed an extensive series of demonstrations for computationally-based methods of design for luminous ceilings, yet many possibilities remain unimplemented. Future projections for the development of this work are anticipated in:

(1) The evaluation of the light performance of a luminous ceiling by observing it in an architectural space to see how it affects the space in its vertical and horizontal surfaces.

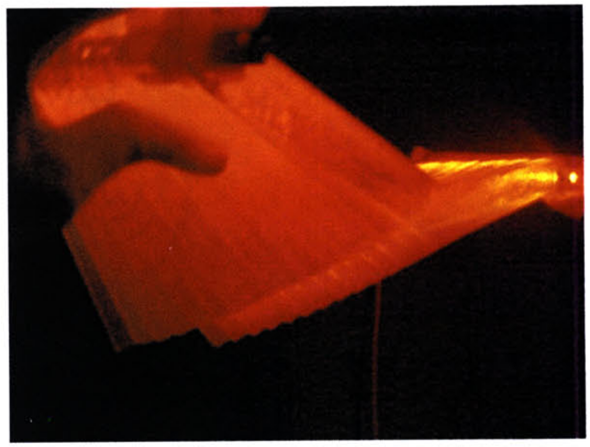
(2) The creation of design rules for the distribution of direct lighting sources following the geometry of the luminous ceilings.

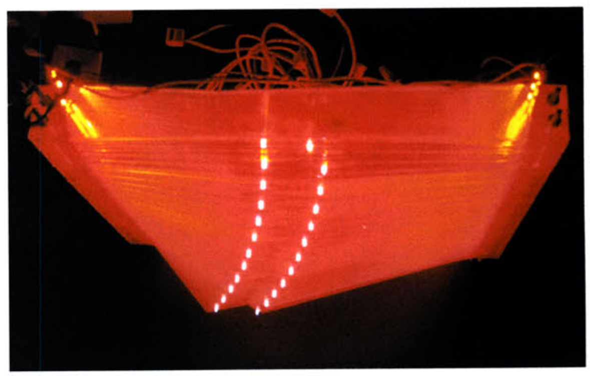
(3) The definition of a relationship between the diffuse and the direct light sources to maximize the performance of the ceiling.

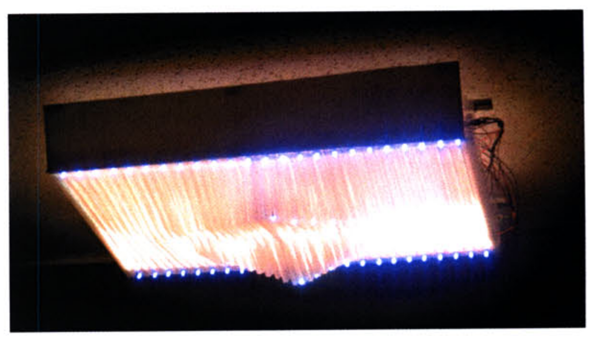
(4) The improvement of the performance through a scientific approach toward the optical properties of acrylic and the behavior of light in the different shapes of the pieces in a tile. This plan encourages the use of different materials in the investigation of reflectance and transmittance such as glass and silicon.

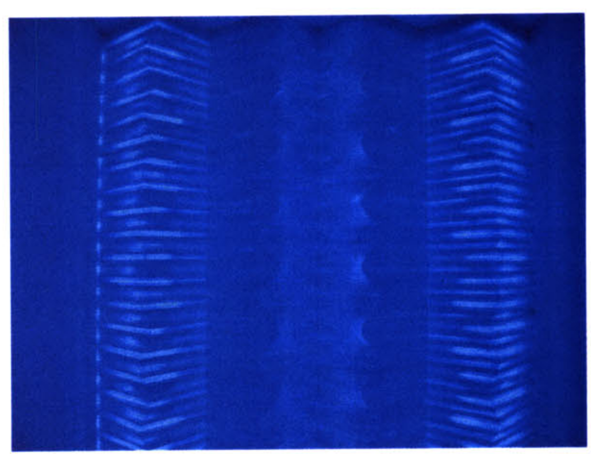
(5) The exploration of different applications in real architectural spaces.

(6) The development of the input interface in a software for the design of luminous ceilings.

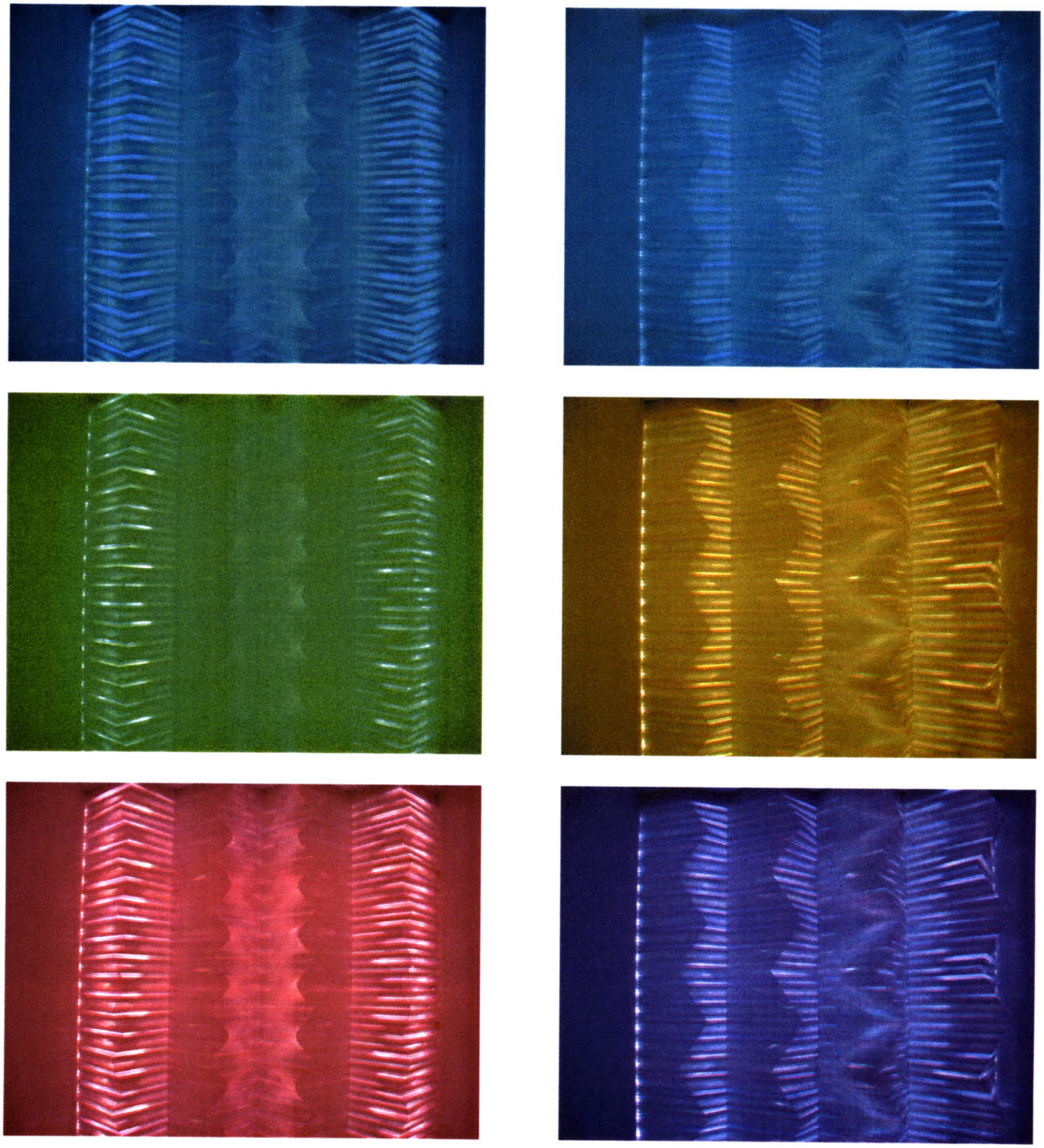








Synopsis of the design investigation of this work, showing the development of the design from a 3D object to an architectural luminous tile and finally to a entire luminous ceiling



physical prototypes of luminous ceilings equipped with embedded color-mix LEDs
demonstrations of two designs generated from the interface in Rhinoceros, one with the *Curves* pattern and another with the *Lines* pattern (4 x 4 tiles, Scale 1:4)

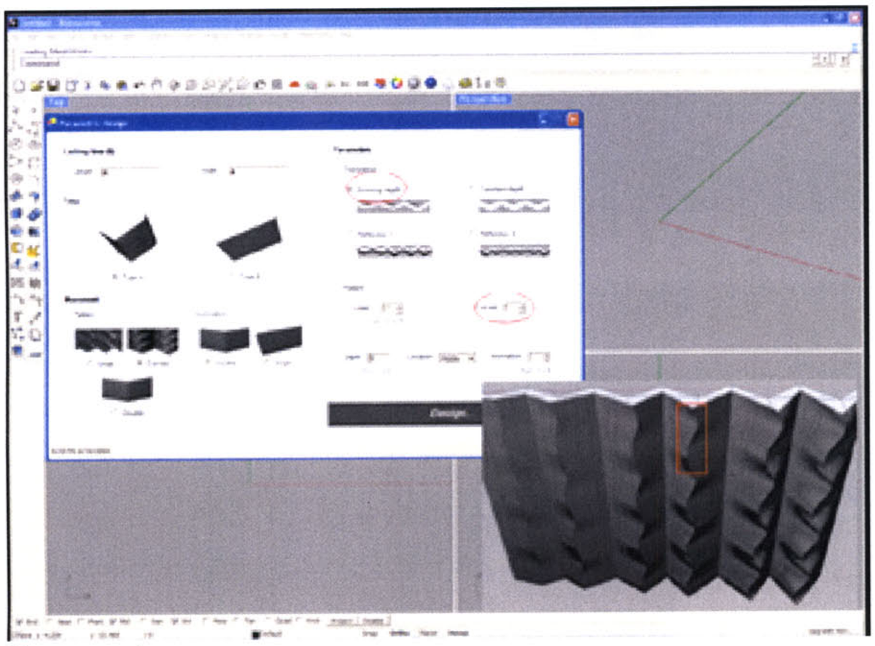
APPENDICES

Appendix 1: Derivations from the Input Interface

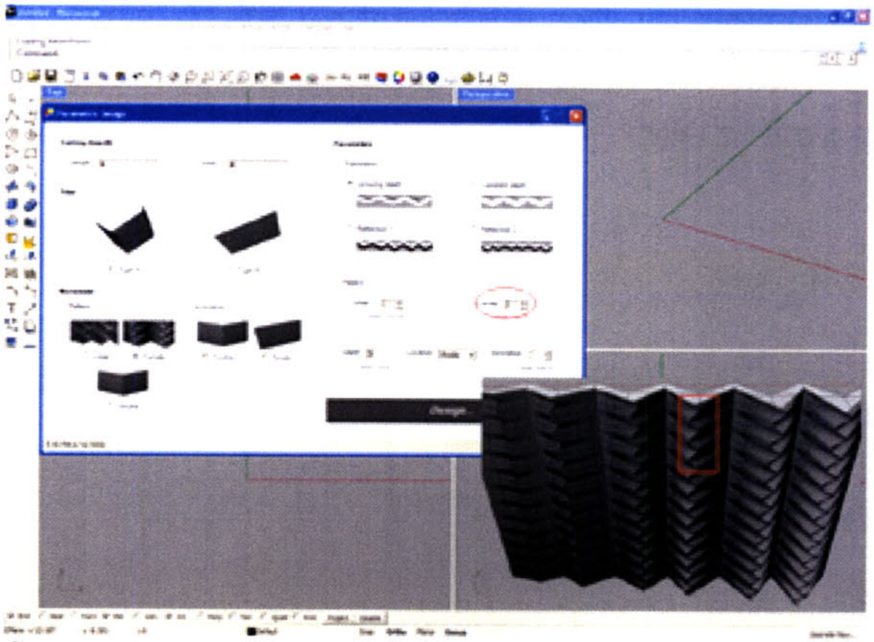
This section shows a series of design output from the manipulation of rules and parameters through the input interface used in Rhinoceros

1.a Manipulation of the *Curves* parameter value

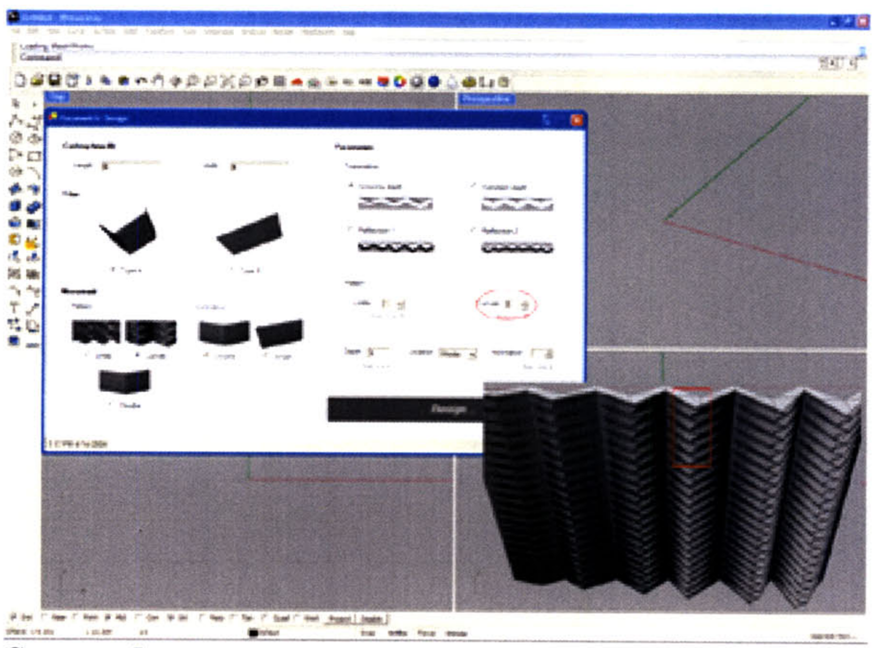
Tiles Type A
Movement Pattern: Curves
Parameters Translation: Growing Depth
Depth = 3
Location: Middle



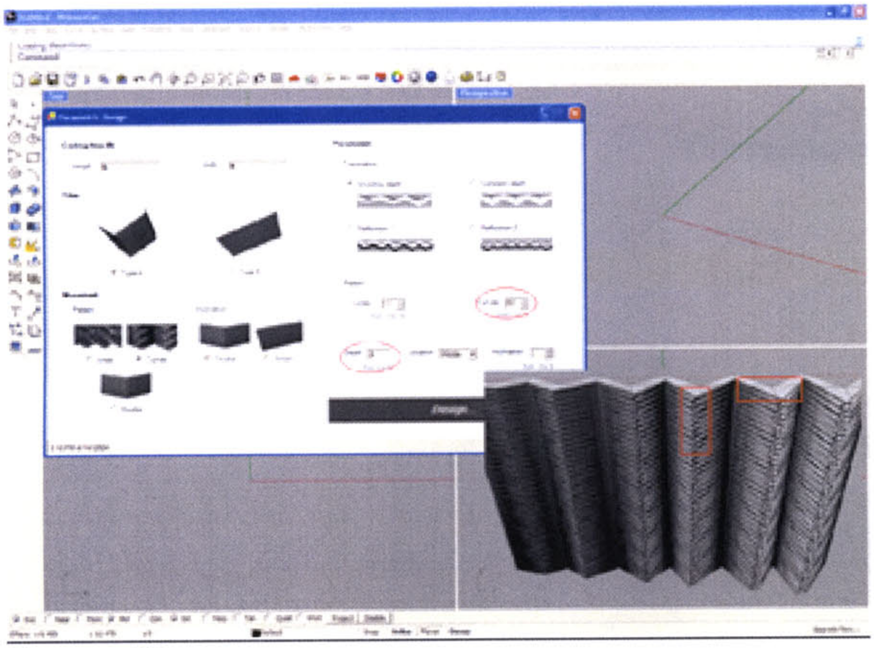
Curves = 1



Curves = 3



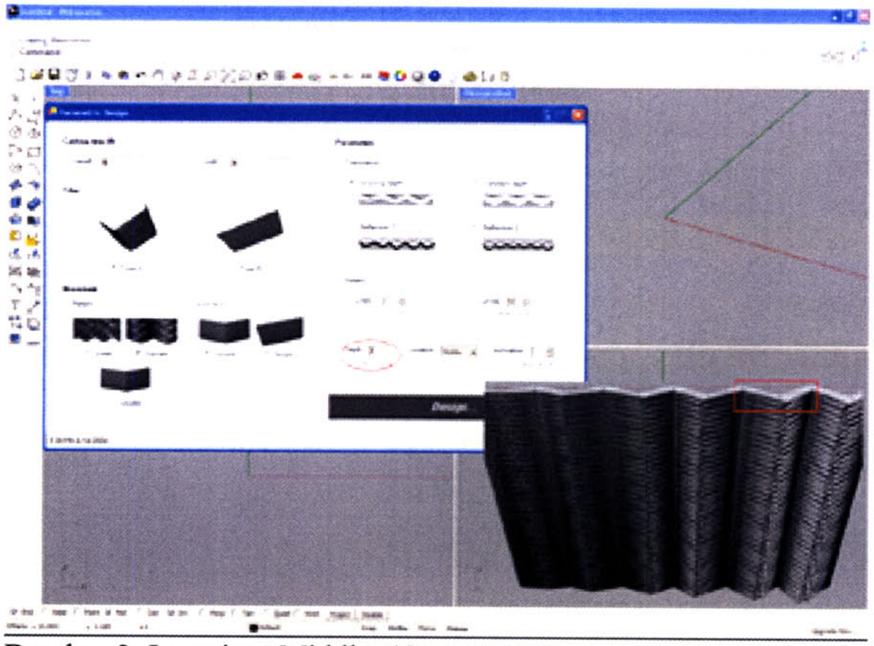
Curves = 5



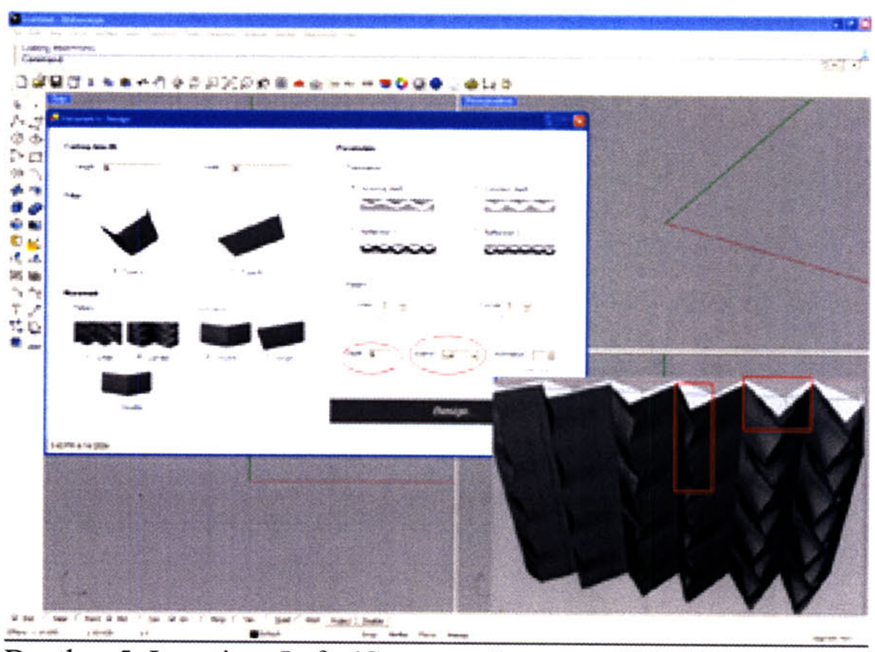
Curves = 10

1.b Manipulation of the *Depth* parameter value and the *Location* parameter

Tiles Type A
Movement Pattern: Curves
Parameters Translation: Growing Depth



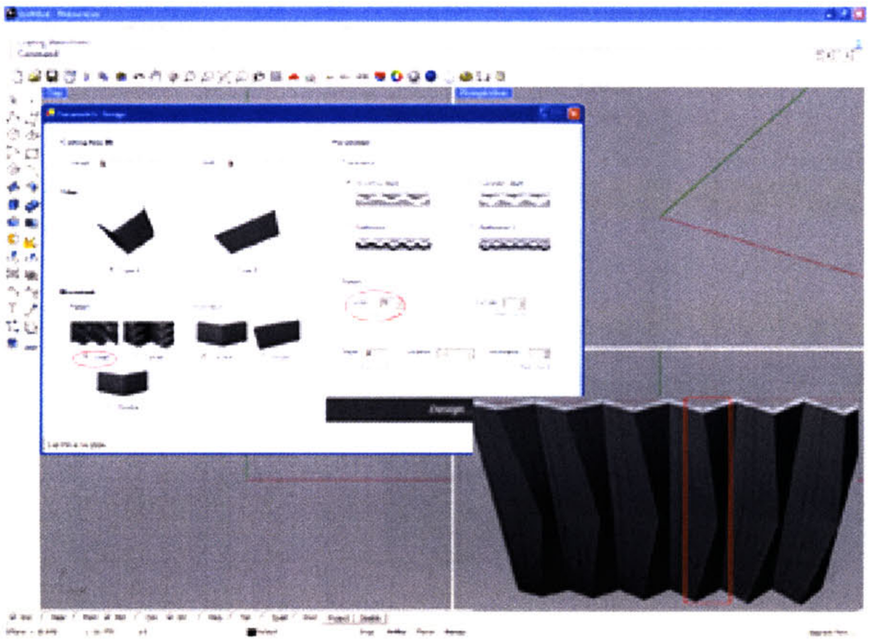
Depth = 2, Location: Middle (Curves = 10)



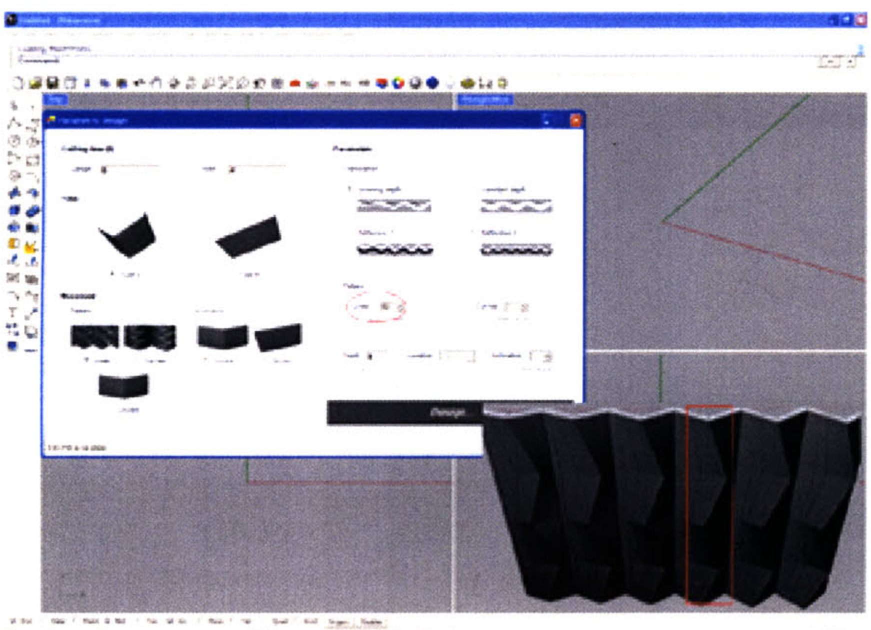
Depth = 5, Location: Left (Curves = 1)

1.c Manipulation of the *Lines* parameter value

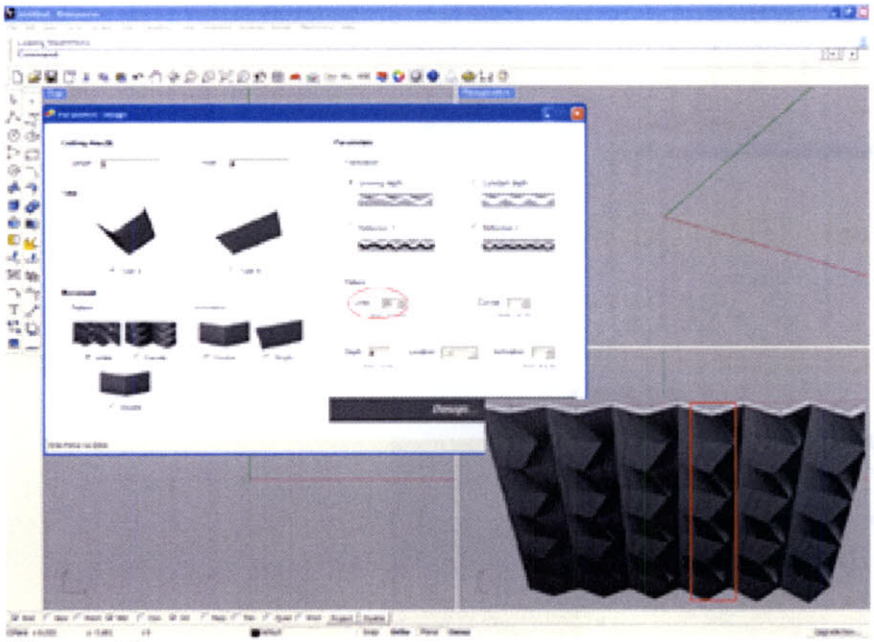
Tiles Type A
Movement Pattern: Lines
Parameters Translation: Growing Depth
Depth = 4



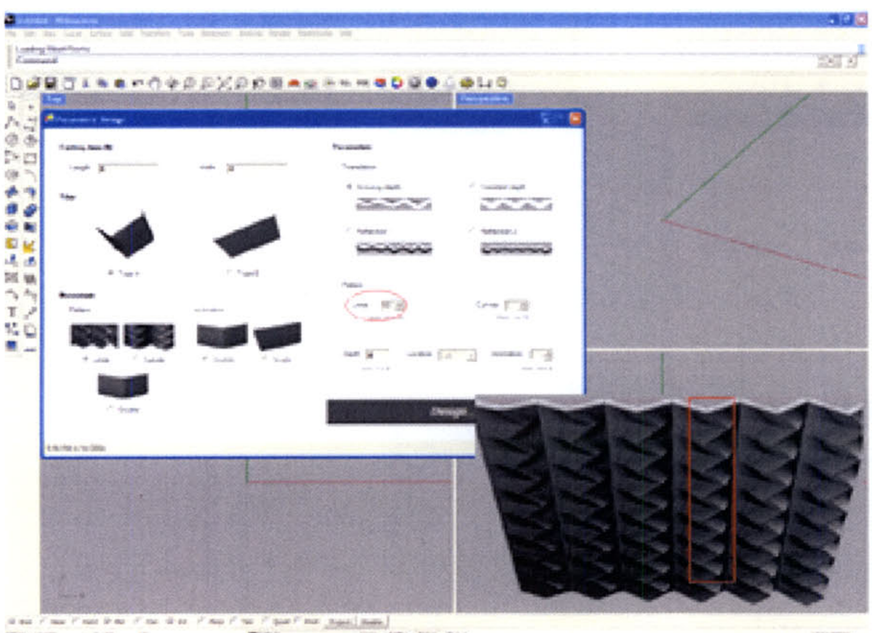
Lines = 70



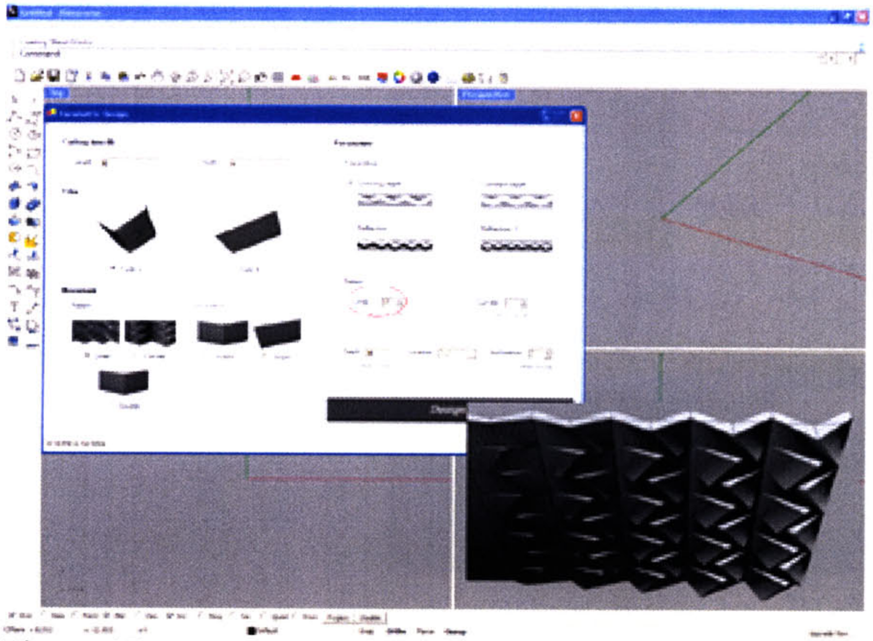
Lines = 50



Lines = 30



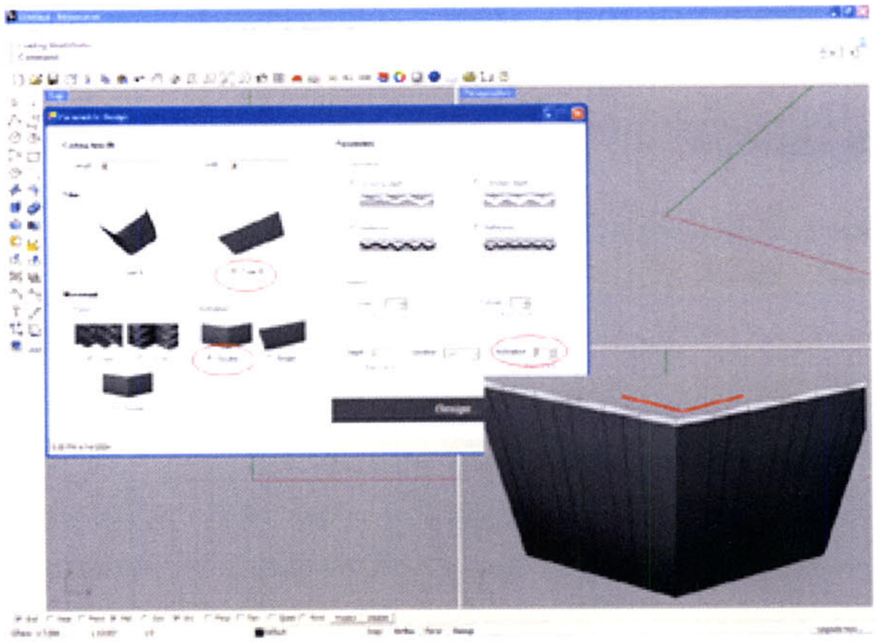
Lines = 10



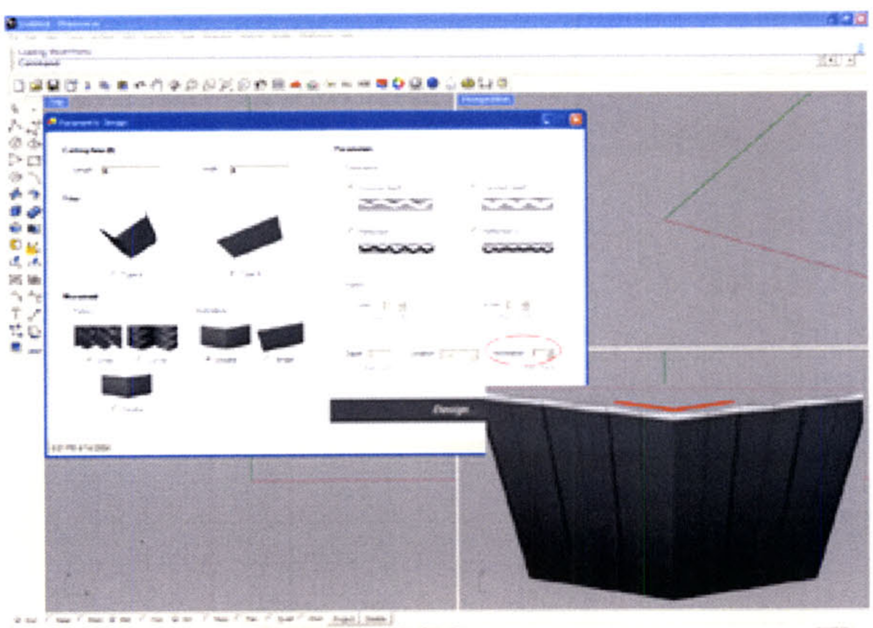
Lines = 37

1.d Manipulation of the *Inclination* parameter value

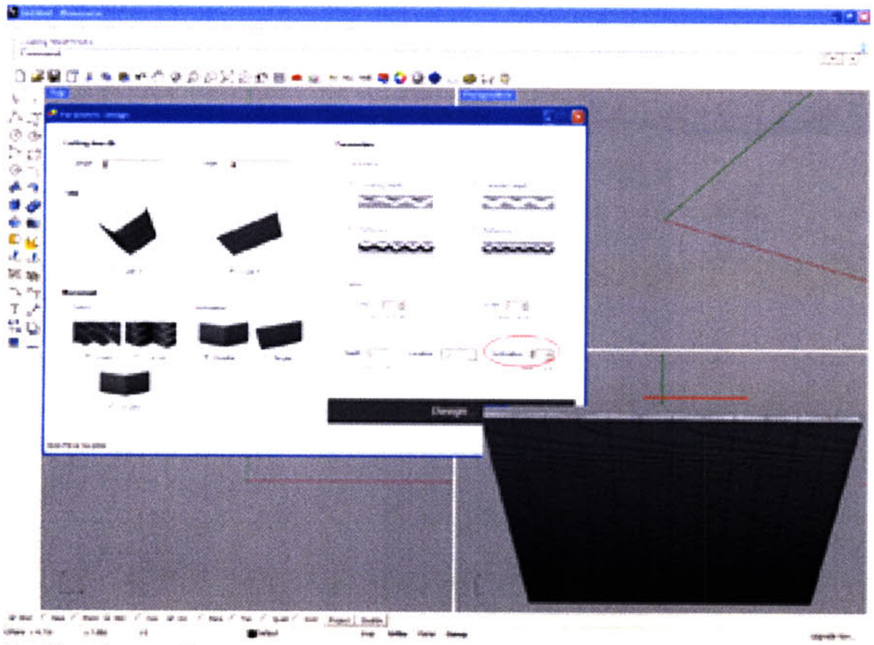
Tiles Type B
Movement Inclination: Double



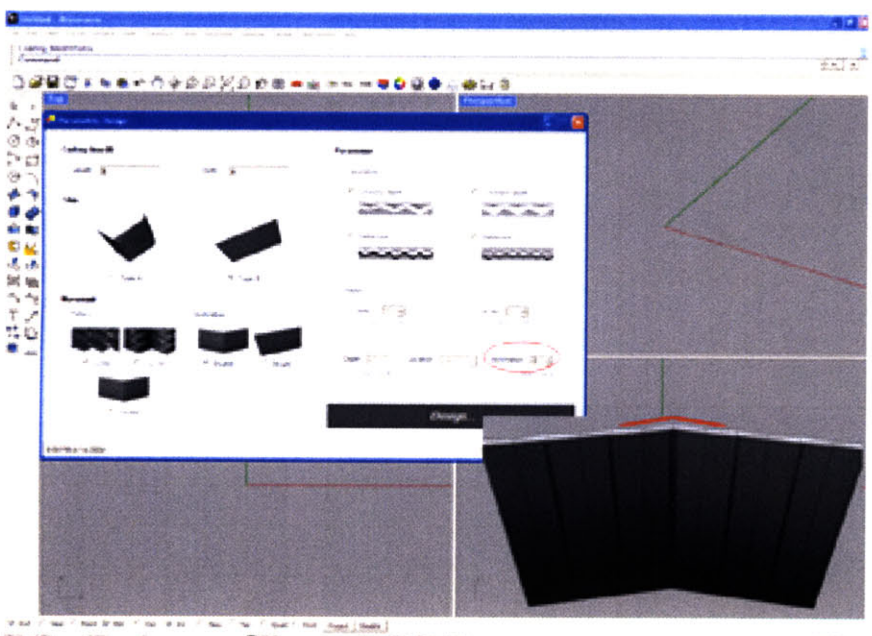
Inclination = 3



Inclination = 1

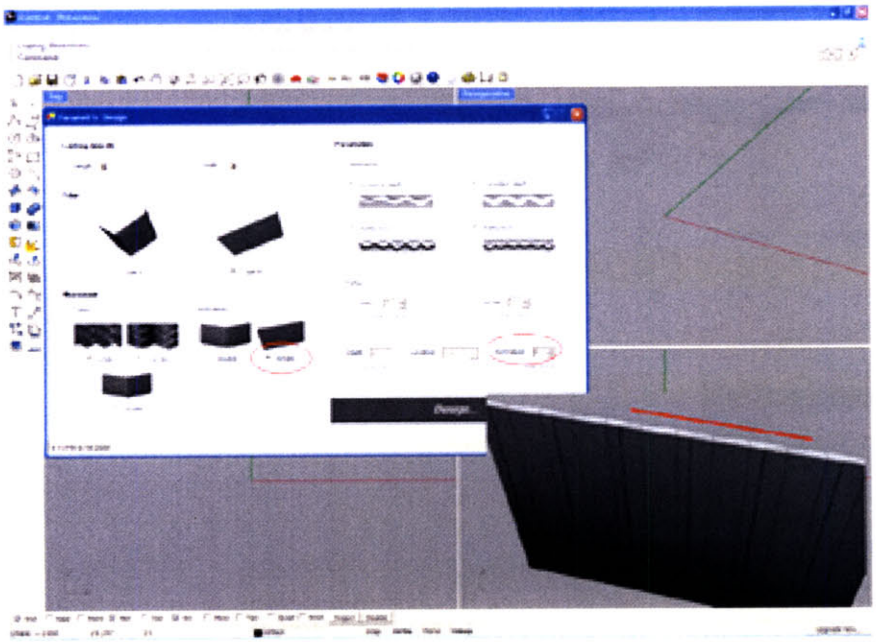


Inclination = 0



Inclination = -3

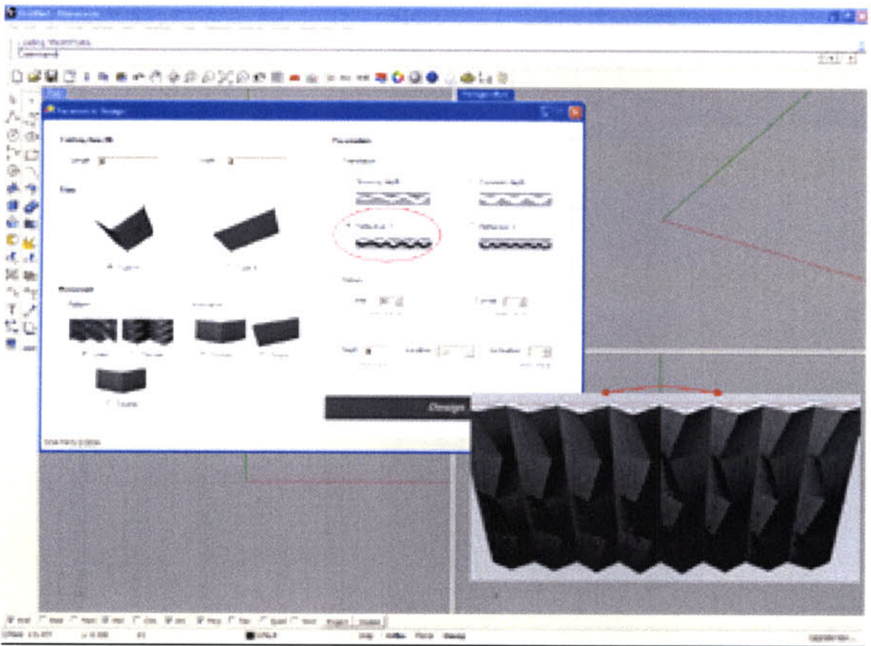
Movement Inclination: Single



Inclination = 2

1.e Manipulation of the *Translation* parameter

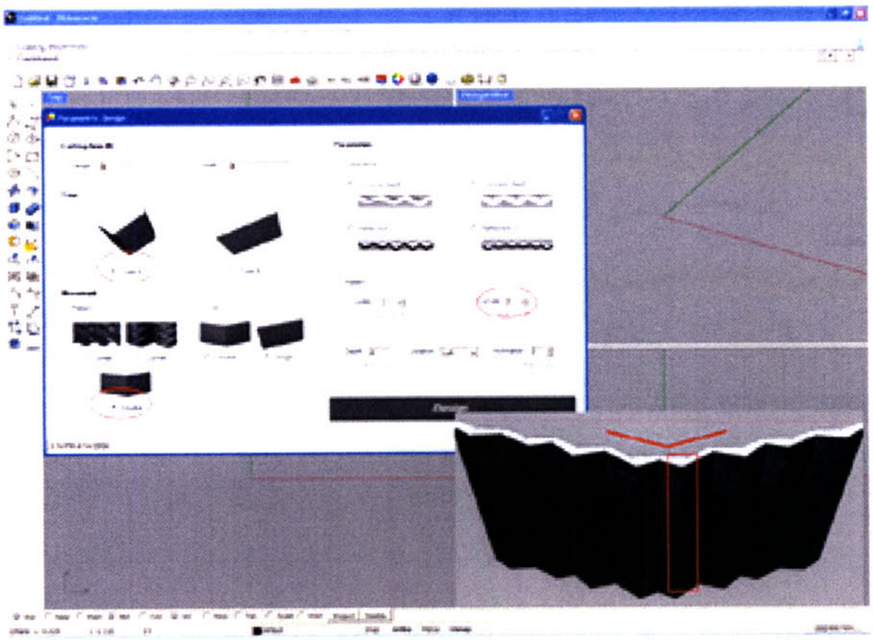
Tiles Type A
Movement Pattern: Double
Parameters Pattern: Curves = 1
Depth = 4



Reflection 1

1.f Manipulation of Rules

Tiles Type A
Movement Pattern: Double
Parameters Pattern: Curves = 1
Depth = 4

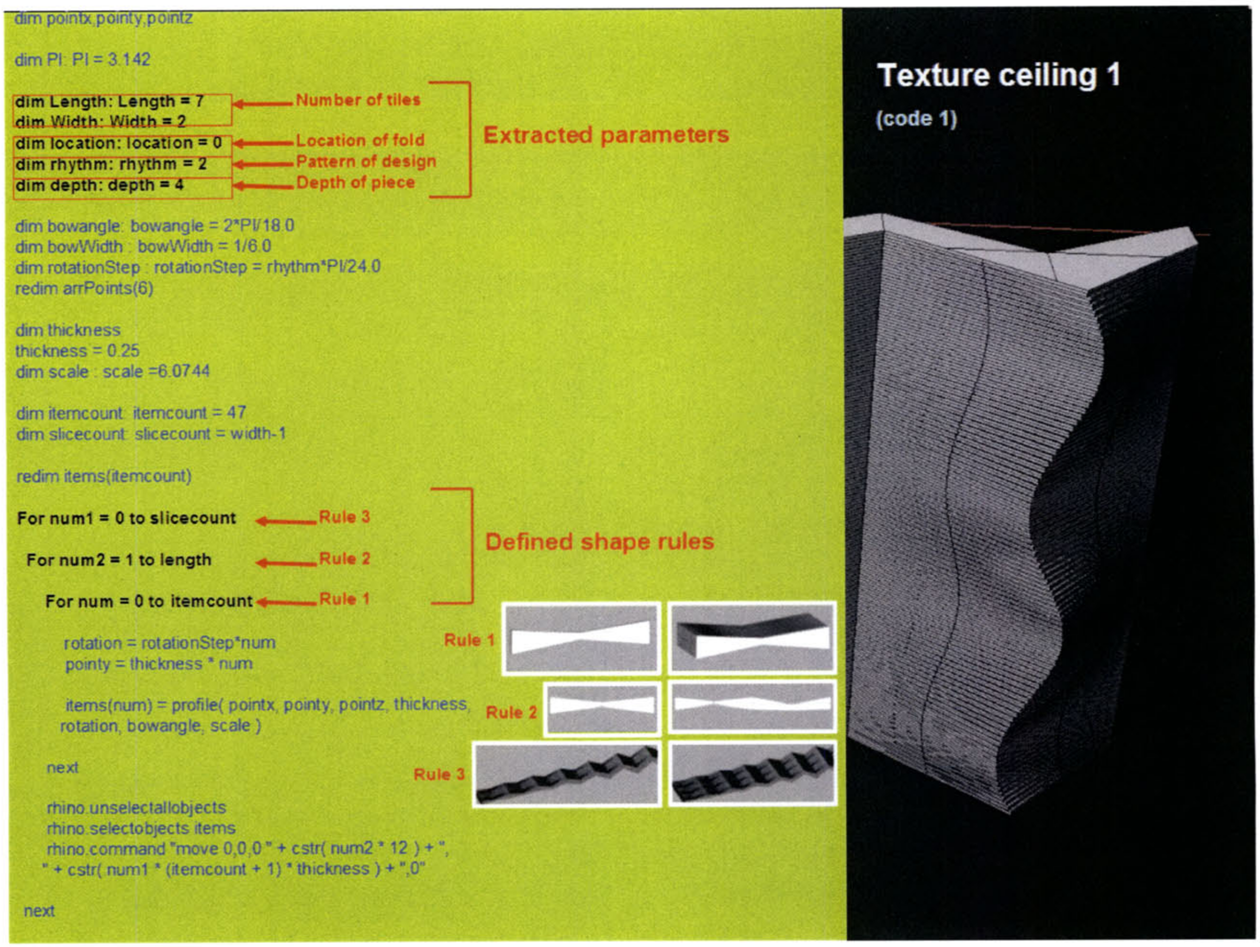


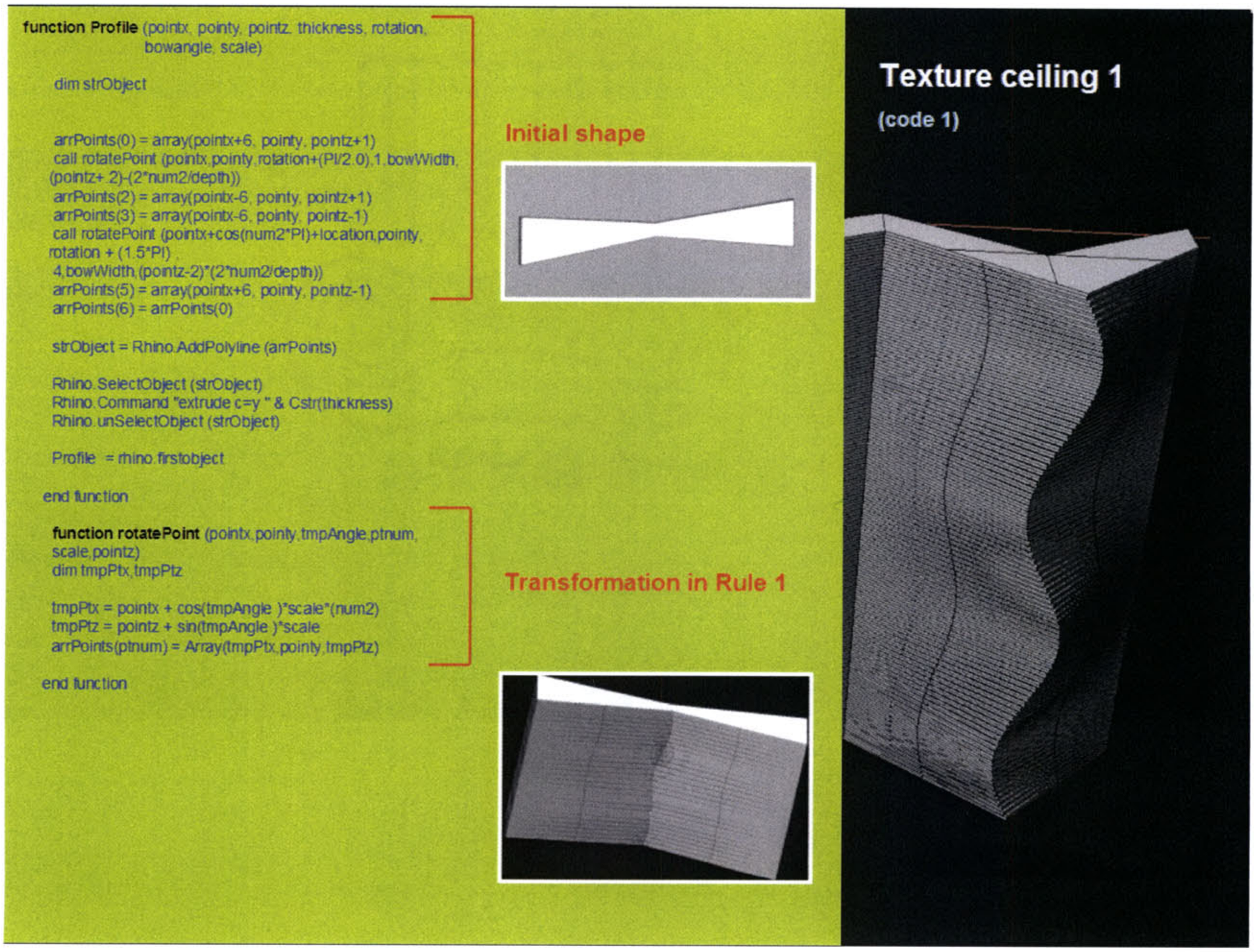
combination of texture and inclination (type 2 and type 1 ceiling)

Appendix 2: Examples in Rhinoscripting

The following scripts are written in Visual Basic programming language and are structured by the shape rules of different designs of ceilings. These rules are expressed in loops and the initial shapes in the rules are expressed in functions that call the rules through the loops.

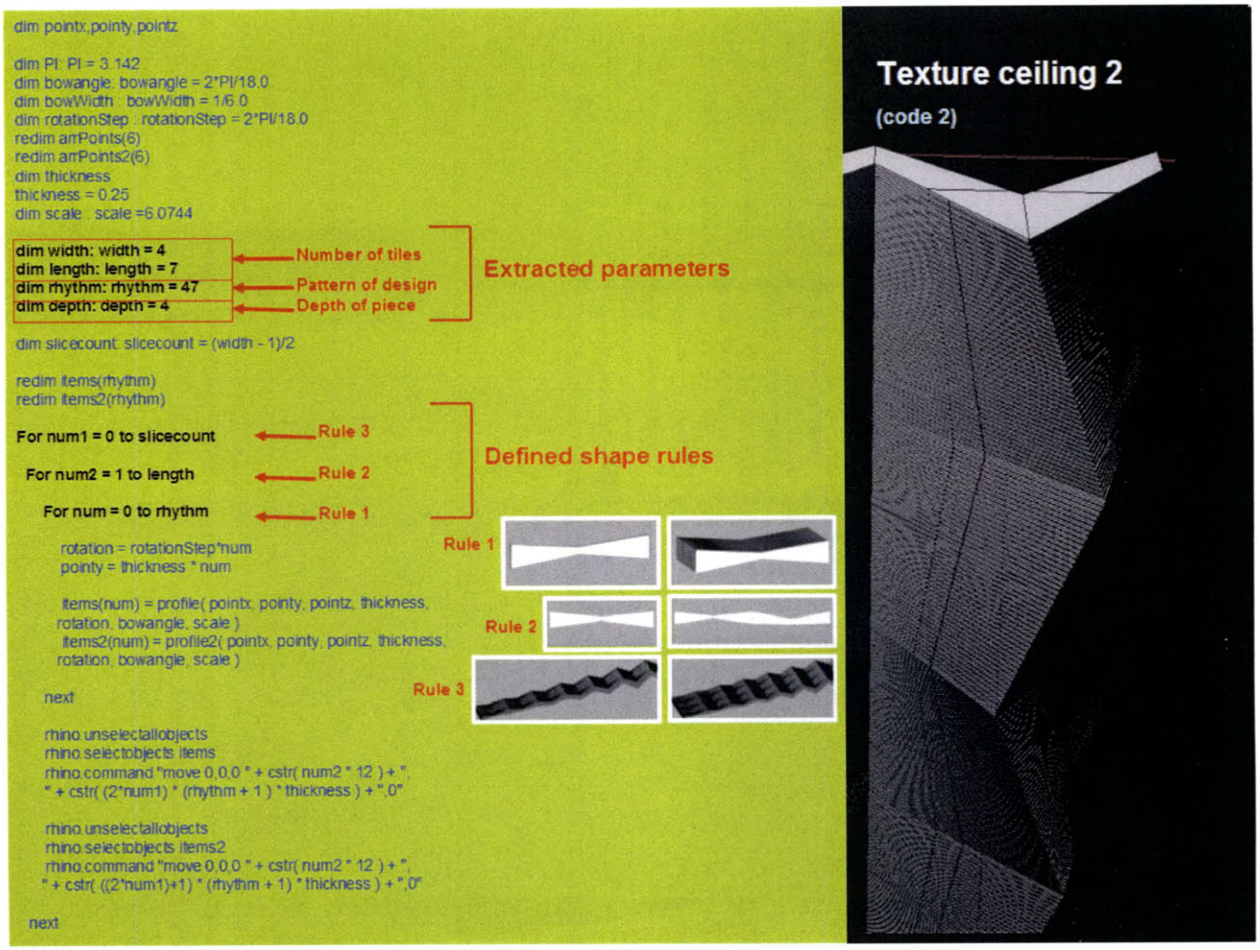
Code 1





function Profile (pointx, pointy, pointz, thickness, rotation,
bowangle, scale)
dim strObject
arrPoints(0) = array(pointx+6, pointy, pointz+1)
call rotatePoint (pointx,pointy,rotation+(PI/2.0),1,bowWidth,
(pointz+.2)-(2*num2/depth))
arrPoints(2) = array(pointx-6, pointy, pointz+1)
arrPoints(3) = array(pointx-6, pointy, pointz-1)
call rotatePoint (pointx+cos(num2*PI)+location,pointy,
rotation + (1.5*PI) ,
4,bowWidth,(pointz-2)*(2*num2/depth))
arrPoints(5) = array(pointx+6, pointy, pointz-1)
arrPoints(6) = arrPoints(0)
strObject = Rhino.AddPolyline (arrPoints)
Rhino.SelectObject (strObject)
Rhino.Command "extrude c=y " & Cstr(thickness)
Rhino.unSelectObject (strObject)
Profile = rhino.firstobject
end function
function rotatePoint (pointx,pointy,tmpAngle,ptnum,
scale,pointz)
dim tmpPtx,tmpPtz
tmpPtx = pointx + cos(tmpAngle)*scale*(num2)
tmpPtz = pointz + sin(tmpAngle)*scale
arrPoints(ptnum) = Array(tmpPtx,pointy,tmpPtz)
end function
Initial shape
Transformation in Rule 1
Texture ceiling 1
(code 1)

Code 2



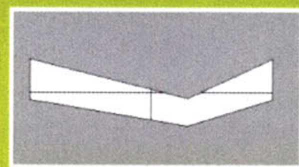
dim pointx,pointy,pointz
dim PI: PI = 3.142
dim bowangle: bowangle = 2*PI/18.0
dim bowWidth : bowWidth = 1/6.0
dim rotationStep : rotationStep = 2*PI/18.0
redim arrPoints(6)
redim arrPoints2(6)
dim thickness
thickness = 0.25
dim scale : scale =6.0744
dim width: width = 4
dim length: length = 7
dim rhythm: rhythm = 47
dim depth: depth = 4
Number of tiles
Pattern of design
Depth of piece
Extracted parameters
dim slicecount: slicecount = (width - 1)/2
redim items(rhythm)
redim items2(rhythm)
For num1 = 0 to slicecount
For num2 = 1 to length
For num = 0 to rhythm
Rule 3
Rule 2
Rule 1
Defined shape rules
rotation = rotationStep*num
pointy = thickness * num
items(num) = profile(pointx, pointy, pointz, thickness, rotation, bowangle, scale)
items2(num) = profile2(pointx, pointy, pointz, thickness, rotation, bowangle, scale)
next
rhino.unselectallobjects
rhino.selectobjects items
rhino.command "move 0,0,0 " + cstr(num2 * 12) + ", " + cstr((2*num1) * (rhythm + 1) * thickness) + ",0"
rhino.unselectallobjects
rhino.selectobjects items2
rhino.command "move 0,0,0 " + cstr(num2 * 12) + ", " + cstr(((2*num1)+1) * (rhythm + 1) * thickness) + ",0"
next
Rule 1
Rule 2
Rule 3
Texture ceiling 2
(code 2)

Texture ceiling 2

(code 2)

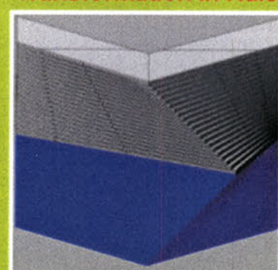
```
function Profile (pointx, pointy, pointz, thickness, rotation,
                  bowangle, scale)

    dim strObject

    arrPoints(0) = array(pointx+6, pointy, pointz+1)
    arrPoints(1) = array(pointx+((num*4)/(rhythm+1)),
    pointy, (pointz+.2)-(depth*num2/(length)))
    arrPoints(2) = array(pointx-6, pointy, pointz+1)
    arrPoints(3) = array(pointx-6, pointy, pointz-1)
    arrPoints(4) = array(pointx+((num*4)/(rhythm+1)),
    pointy, (pointz-2)-(depth*num2/(length)))
    arrPoints(5) = array(pointx+6, pointy, pointz-1)
    arrPoints(6) = arrPoints(0)

    strObject = Rhino.AddPolyline (arrPoints)

    Rhino.SelectObject (strObject)
    Rhino.Command "extrude c=y " & Cstr(thickness)
    Rhino.unSelectObject (strObject)

    Profile  = rhino.firstobject

  end function
```

Initial shape

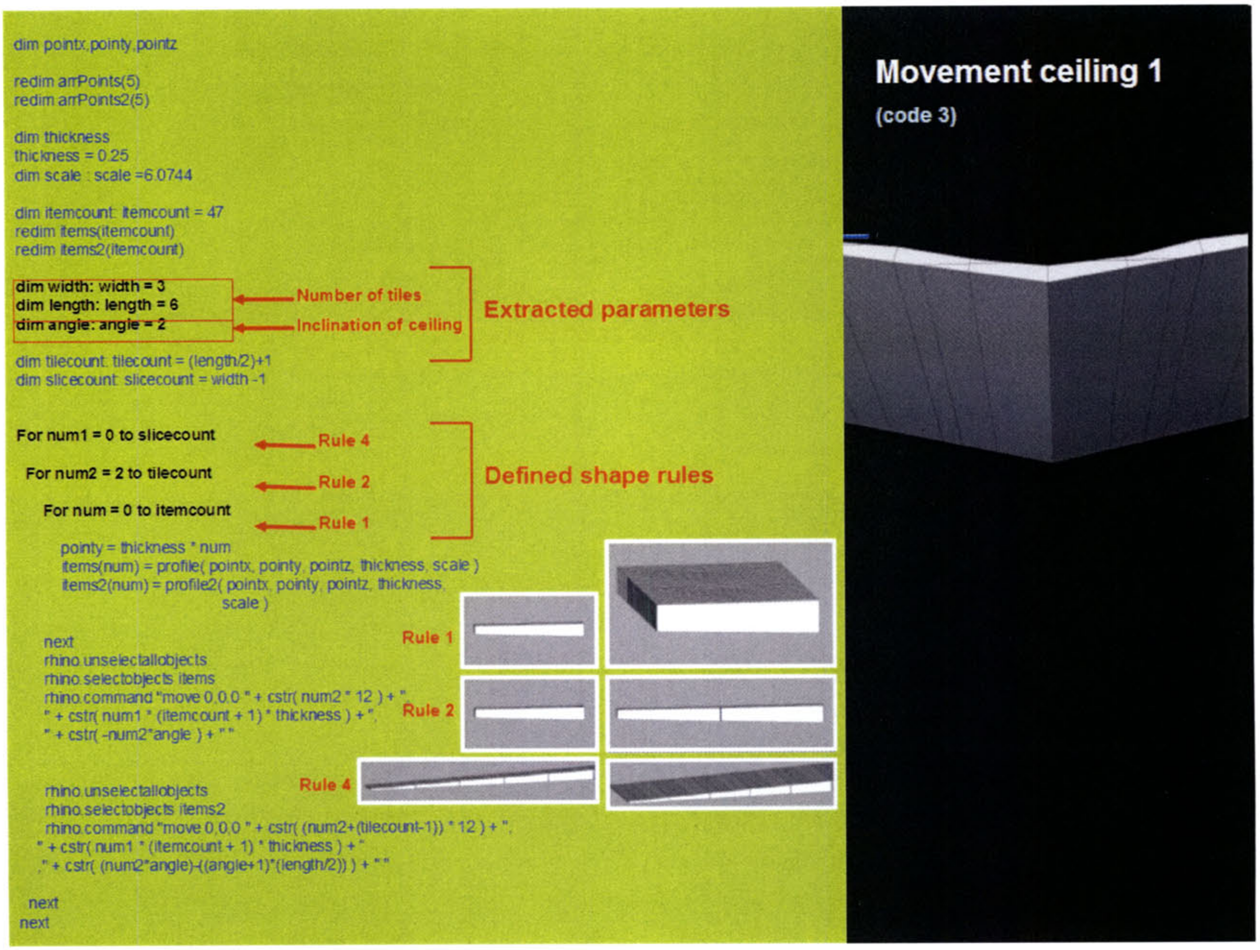


```
  function Reflection (pointx, pointy, pointz, thickness, rotation
                       bowangle, scale)

    dim strObject2

    arrPoints2(0) = array(pointx+6, pointy, pointz+1)
    arrPoints2(1) = array((pointx+4)-((num*4)/(rhythm+1)),
    pointy,(pointz+.2)-(depth*num2/(length)))
    arrPoints2(2) = array(pointx-6, pointy, pointz+1)
    arrPoints2(3) = array(pointx-6, pointy, pointz-1)
    arrPoints2(4) = array((pointx+4)-((num*4)/(rhythm+1)),
     pointy, (pointz-2)-(depth*num2/(length)))
    arrPoints2(5) = array(pointx+6, pointy, pointz-1)
    arrPoints2(6) = arrPoints2(0)

    strObject2 = Rhino.AddPolyline (arrPoints2)

    Rhino.SelectObject (strObject2)
    Rhino.Command "extrude c=y " & Cstr(thickness)
    Rhino.unSelectObject (strObject2)

    Profile2  = rhino.firstobject

  end function
```

Transformation in Rule 1



Code 3

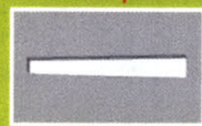

dim pointx,pointy,pointz
redim arrPoints(5)
redim arrPoints2(5)
dim thickness
thickness = 0.25
dim scale : scale =6.0744
dim itemcount: itemcount = 47
redim items(itemcount)
redim items2(itemcount)
dim width: width = 3
dim length: length = 6
dim angle: angle = 2
Number of tiles
Inclination of ceiling
Extracted parameters
dim tilecount: tilecount = (length/2)+1
dim slicecount: slicecount = width -1
For num1 = 0 to slicecount
Rule 4
For num2 = 2 to tilecount
Rule 2
For num = 0 to itemcount
Rule 1
Defined shape rules
pointy = thickness * num
items(num) = profile(pointx, pointy, pointz, thickness, scale)
items2(num) = profile2(pointx, pointy, pointz, thickness, scale)
next
rhino.unselectallobjects
rhino.selectobjects items
rhino.command "move 0,0,0 " + cstr(num2 * 12) + ", " + cstr(num1 * (itemcount + 1) * thickness) + ", " + cstr(-num2*angle) + " "
Rule 1
Rule 2
Rule 4
rhino.unselectallobjects
rhino.selectobjects items2
rhino.command "move 0,0,0 " + cstr((num2+(tilecount-1)) * 12) + ", " + cstr(num1 * (itemcount + 1) * thickness) + " ," + cstr((num2*angle)-((angle+1)*(length/2))) + " "
next
next
Movement ceiling 1
(code 3)

Movement ceiling 1

(code 3)

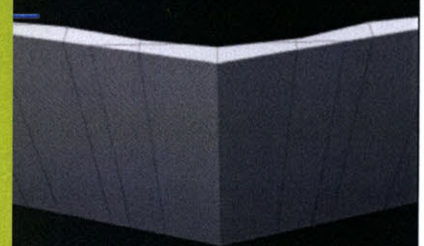
```
function Profile (pointx, pointy, pointz, thickness, scale)

  dim strObject

  arrPoints(0) = array(pointx+6, pointy,
  (pointz+2-angle)-num2)
  arrPoints(1) = array(pointx+.5,pointy,
  pointz+2-(angle/2))-num2)
  arrPoints(2) = array(pointx-6, pointy, (pointz+3)-num2)
  arrPoints(3) = array(pointx-6, pointy, (pointz+1)-num2)
  arrPoints(4) = array(pointx+6, pointy, (pointz-angle)-num2)
  arrPoints(5) = arrPoints(0)

  strObject = Rhino.AddPolyline (arrPoints)

  Rhino.SelectObject (strObject)
  Rhino.Command "extrude c=y " & Cstr(thickness)
  Rhino.unSelectObject (strObject)

  Profile  = rhino.firstobject

end function
```

Initial shape

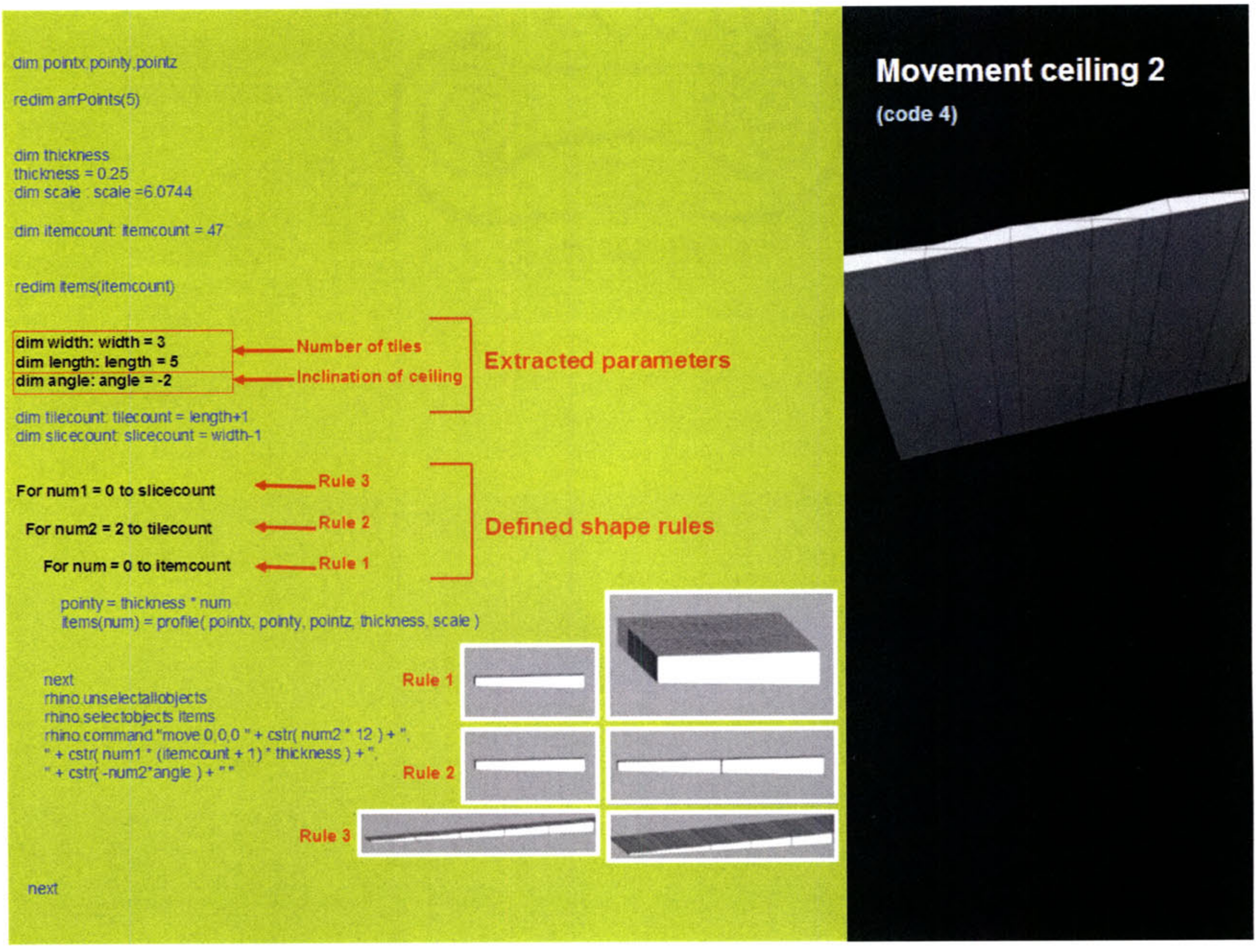


```
function Reflection (pointx, pointy, pointz, thickness, scale)

  dim strObject2

  arrPoints2(0) = array(pointx-6, pointy,
  (pointz+2-angle)+num2)
  arrPoints2(1) = array(pointx+.5,pointy,
  (pointz+2-(angle/2))+num2)
  arrPoints2(2) = array(pointx+6, pointy, (pointz+3)+num2)
  arrPoints2(3) = array(pointx+6, pointy, (pointz+1)+num2)
  arrPoints2(4) = array(pointx-6, pointy, (pointz-angle)+num2)
  arrPoints2(5) = arrPoints2(0)

  strObject2 = Rhino.AddPolyline (arrPoints2)

  Rhino.SelectObject (strObject2)
  Rhino.Command "extrude c=y " & Cstr(thickness)
  Rhino.unSelectObject (strObject2)

  Profile2  = rhino.firstobject

end function
```

Rule 3



Code 4

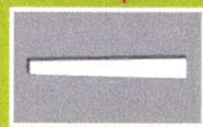

Movement ceiling 2
(code 4)
dim pointx.pointy.pointz
redim arrPoints(5)
dim thickness
thickness = 0.25
dim scale : scale =6.0744
dim itemcount: itemcount = 47
redim items(itemcount)
dim width: width = 3
dim length: length = 5
dim angle: angle = -2
Number of tiles
Inclination of ceiling
Extracted parameters
dim tilecount: tilecount = length+1
dim slicecount: slicecount = width-1
For num1 = 0 to slicecount
For num2 = 2 to tilecount
For num = 0 to itemcount
Rule 3
Rule 2
Rule 1
Defined shape rules
pointy = thickness * num
items(num) = profile(pointx, pointy, pointz, thickness, scale)
next
rhino.unselectallobjects
rhino.selectobjects items
rhino.command "move 0,0,0 " + cstr(num2 * 12) + ",
" + cstr(num1 * (itemcount + 1) * thickness) + ",
" + cstr(-num2*angle) + " "
Rule 1
Rule 2
Rule 3
next

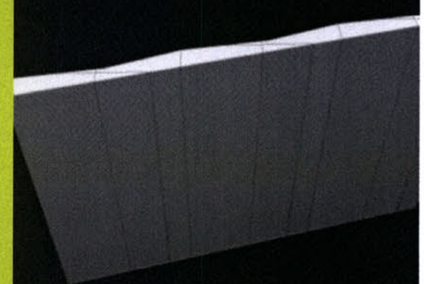
Movement ceiling 2

(code 4)

```
function Profile (pointx, pointy, pointz, thickness, scale)

    dim strObject

    arrPoints(0) = array(pointx+6, pointy,
    (pointz+2-angle)-num2)
    arrPoints(1) = array(pointx+.5,pointy,
    (pointz+2-(angle/2))-num2)
    arrPoints(2) = array(pointx-6, pointy, (pointz+3)-num2)
    arrPoints(3) = array(pointx-6, pointy, (pointz+1)-num2)
    arrPoints(4) = array(pointx+6, pointy,
    (pointz-angle)-num2)
    arrPoints(5) = arrPoints(0)

    strObject = Rhino.AddPolyline (arrPoints)

    Rhino.SelectObject (strObject)
    Rhino.Command "extrude c=y " & Cstr(thickness)
    Rhino.unSelectObject (strObject)

    Profile  = rhino.firstobject

  end function
```

Initial shape





Appendix 3: Sample Code for the creation of text boxes in the Input Interface

The following code is written in C# programming language for the creation of text boxes for the input of the length and the width of the ceilings to be generated by the interface. It is a sample extracted from the code written to build the input interface used in Rhinoceros to generate designs for luminous ceilings.

```
                        this.grpArea.Font = new System.Drawing.Font("Microsoft
Sans Serif", 8.25F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                this.grpArea.Location = new System.Drawing.Point(16, 24);
                this.grpArea.Name = "grpArea";
                this.grpArea.Size = new System.Drawing.Size(424, 72);
                this.grpArea.TabIndex = 0;
                this.grpArea.TabStop = false;
                this.grpArea.Text = "Ceiling Area (ft)";
                //
                // label1
                //
                this.label1.Font = new System.Drawing.Font("Microsoft Sans
Serif", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                this.label1.Location = new System.Drawing.Point(24, 32);
                this.label1.Name = "label1";
                this.label1.Size = new System.Drawing.Size(48, 23);
                this.label1.TabIndex = 2;
                this.label1.Text = "Length";
                //
                // txtWidth
                //
                this.txtWidth.BackColor = System.Drawing.SystemColors.Control;
                this.txtWidth.Location = new System.Drawing.Point(280, 32);
                this.txtWidth.Name = "txtWidth";
                this.txtWidth.TabIndex = 1;
                this.txtWidth.Text = "";
                this.txtWidth.TextChanged += new
System.EventHandler(this.txtWidth_TextChanged);
                //
                // txtLength
                //
                this.txtLength.BackColor =
System.Drawing.SystemColors.Control;
                this.txtLength.Location = new System.Drawing.Point(72, 32);
                this.txtLength.Name = "txtLength";
                this.txtLength.TabIndex = 0;
                this.txtLength.Text = "";
                this.txtLength.TextChanged += new
System.EventHandler(this.txtLength_TextChanged);
                //
                // label2
```

```
                //
                this.label2.Font = new System.Drawing.Font("Microsoft Sans
Serif", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                this.label2.Location = new System.Drawing.Point(248, 56);
                this.label2.Name = "label2";
                this.label2.Size = new System.Drawing.Size(48, 23);
                this.label2.TabIndex = 3;
                this.label2.Text = "Width";
```

(code written by Loai Naamani)

REFERENCES

Arik M., Petroski J., and Weaver S.
"Thermal Challenges in the Future Generation Solid State Lighting Applications: Light Emitting Diodes", *ITherm 2002- International Conference on Thermal, Mechanics and Thermomechanical Phenomena in Electronic Systems*, April 2002, 2002GRC078

Burry, Mark.
"Rapid Prototyping CAD/CAM and Human Factors", *Automation in Construction* 11 (2002): 313-333

Cagan, M. Agarwal.
"A Blend of Different Tastes: The Language of Coffeemakers", vol. 25 of *Environment and Planning B* (Great Britain: Pion Publication, 1998), 205-226

Craford, M. George.
"LEDs for Solid State Lighting: Technology, Applications, and the Remaining Challenges", *2004 International Technology Conference* (2004)

Egan, M. David.
Concepts in Architectural Lighting. USA: Mc Graw-Hill, Inc., 1983

Gardner, Carl.
Light. Switzerland: RotoVision, 2001

Gips, James.
"Computer Implementation of Shape Grammars", *Workshop on Shape Computation*. MIT, 1999

Gips J., and George Stiny.
"Production Systems and Grammars: a Uniform Characterization", vol. 7 of *Environment and Planning B* (Great Britain: Pion Publication, 1980), 399-408

Kaoru, Mende and Lighting Planners Associates.
Designing with Light and Shadow. Mulgrave: Images Publications, 2000

Kohler, Walter.
Lighting in Architecture. New York: Reinhold Publishing Corporation, 1959

Kress and Adams.
Light Spaces. Basel, Boston, Berlin: Birkhauser-Publishers for Architecture, 2003

Loukissas, Yanni.
"Rulebuilding: Exploring Design Worlds Through End-User Programming". Master of Science in Architecture Studies, MIT, 2003

Millet, Marietta.
Light Revealing Architecture. USA: International Thomson Publishing, Inc., 1996

Mitchell, William, and Malcolm McCullough.
Digital Design Media. New York: Van Nostrand Reinhold, 1995

Sass, Larry.
"Reconstructing Palladio's Villas: A Computational Analysis of Palladio's Villa Design and Construction Process". *21st Annual Conference of the Association for Computer-Aided Design in Architecture.* Buffalo: New York, 11-14 October 2001

Sass, Larry.
"Rule Based Rapid Prototyping of Palladio's Villa Details", *Digital Design, 22nd International eCAADe Conference.* Graz: Austria, 17-20 September 2003

Shelden, Dennis.
"Digital Surface Representation and the Constructibility of Gehry's Architecture". Doctor of Philosophy in the Field of Architecture: Design and Computation, MIT 2002, 263-331

Steffy, Gary, et al.
Time-Saver Standards for Architectural Lighting. USA: McGraw-Hill Companies, Inc., 2000

Steigerwald Daniel, and Jerome Bhat.
"Illumination With Solid State Lighting Technology", *IEEE Journal on Selected Topics in Quantum Electronics* 8, no. 2 (2002)

Stiny, George.
"Ice-ray: A Note on the Generation of Chinese Lattice Designs", vol. 3 of *Environment and Planning B* (Great Britain: Pion Publication, 1976), 187-210

Sutherland, Ivan Edward.
"Sketchpad, A Man-Machine Graphical Communication System", Doctor of Philosophy, MIT 1963

Special Note:
Unless specified all images are the property of the author