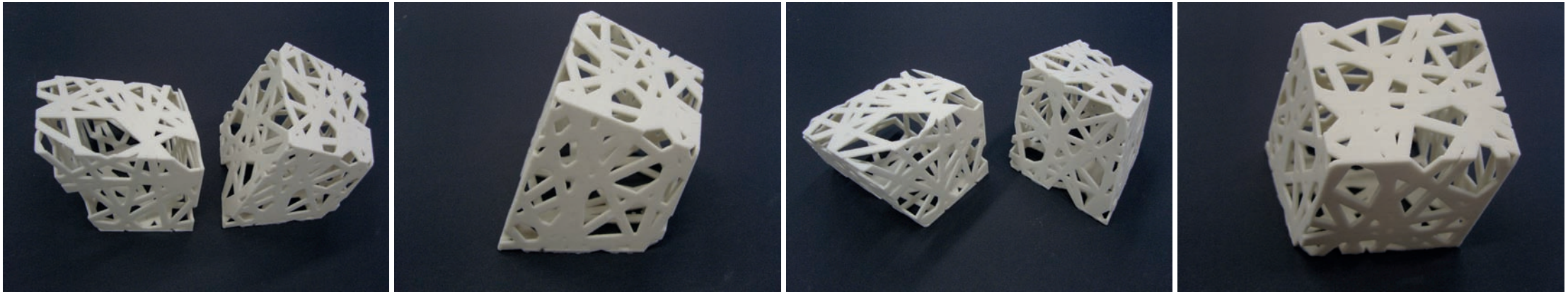
SEONGKI LEE | PORTFOLIO

romekorea@gmail.com

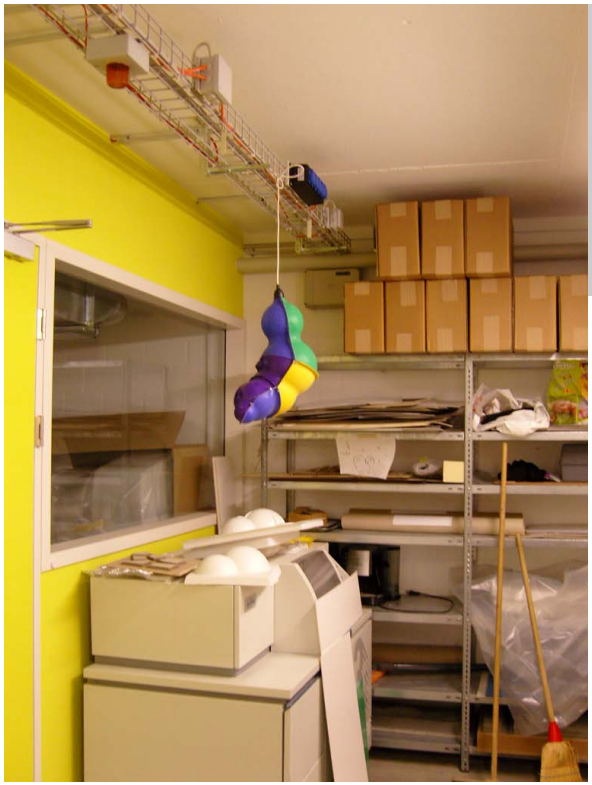
2006 - 2012



01

/SPRING 2007
/CNC FABRICATION
/VACUUM FABRICATION

Parametric Light Lamp



1.

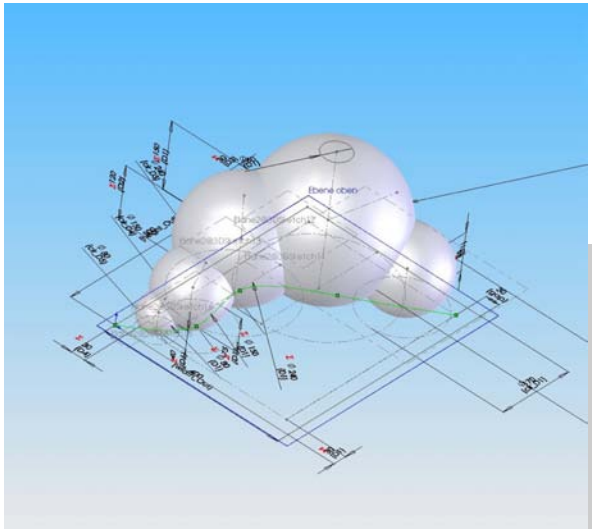
2.

3.

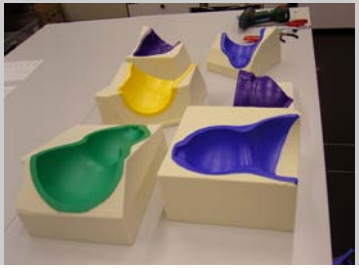
4.

Work Process

The purpose of making the parametric light lamp was to understand Computer Aided Design (CAD)-Computer Aided Manufacturing (CAM) from practical point of view. To manufacture surface parts of the lamp, the vacuum cupping tool that are popular nowadays was chosen. Based on geometric constraints, six part models of the lamp was modeled using SolidWorks and G-Code files for milling maching operation were generated respectively. The 3D model was perfectly parameterized to cope with unexpected change of constraints.
The goal of this project is to merge diverse design tools in search of tangible design alternatives.
As strong design constraints such as material and size were given, a careful consideration from the early design stage was needed. The motif of the design came from a bug. After testing productivity, five female cutting form and one male cutting form were produced from a milling machine. Maya and SolidWorks were used for 3D solid modeling and Surfcam and RhinoCAM for G-Code generation, and "Precix" CNC 3-axis milling machine and vacuum cupping for 3D production.



5.



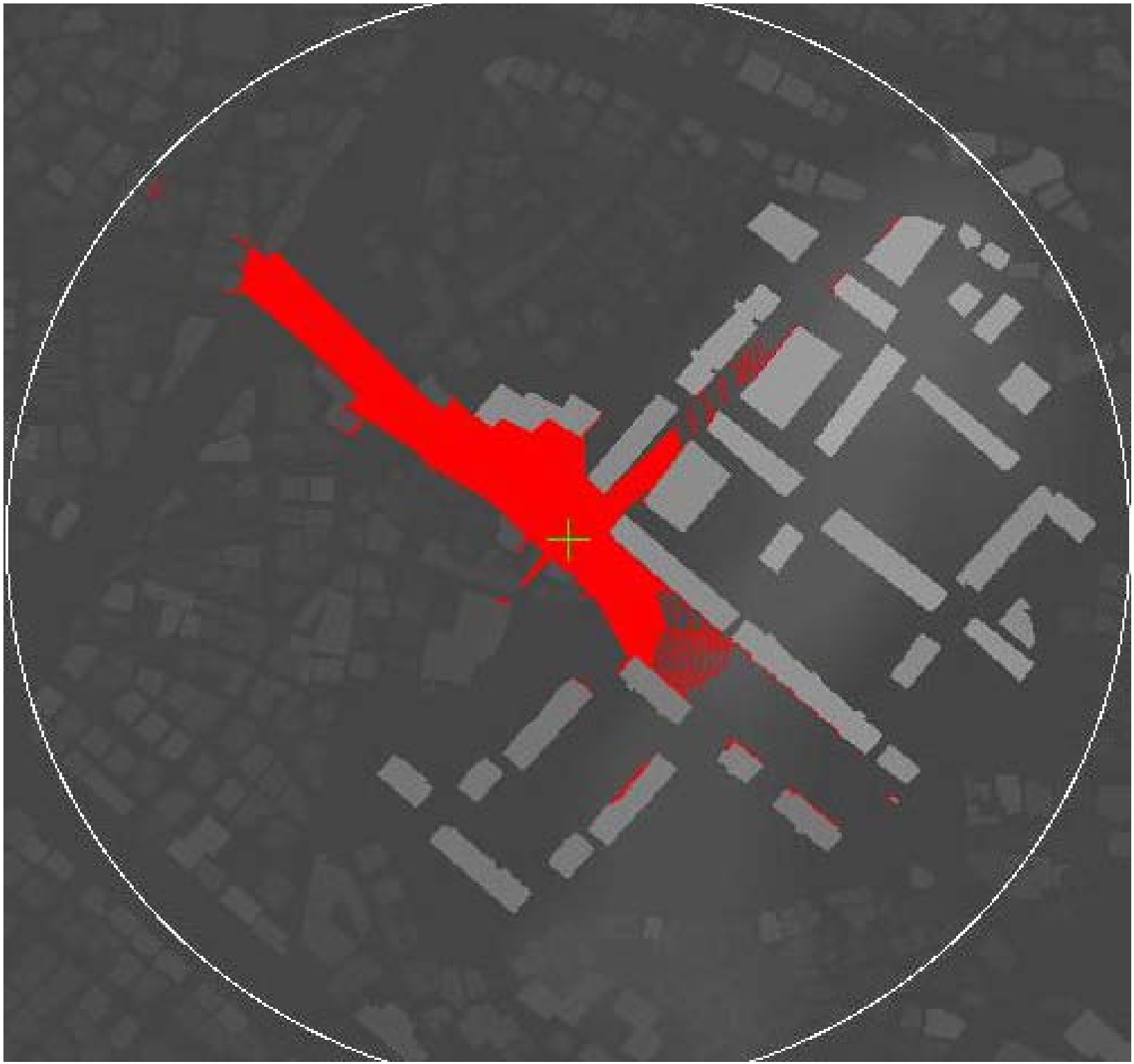
6.

7.

8.

02

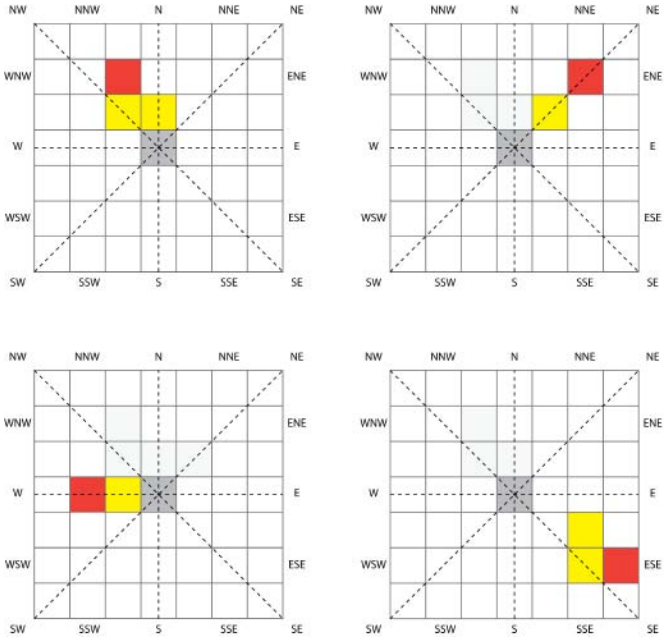
/SPRING 2011
/OUTDOOR VISIBILITY ANALYSIS
/VIEWSTACK ANALYSIS TOOL

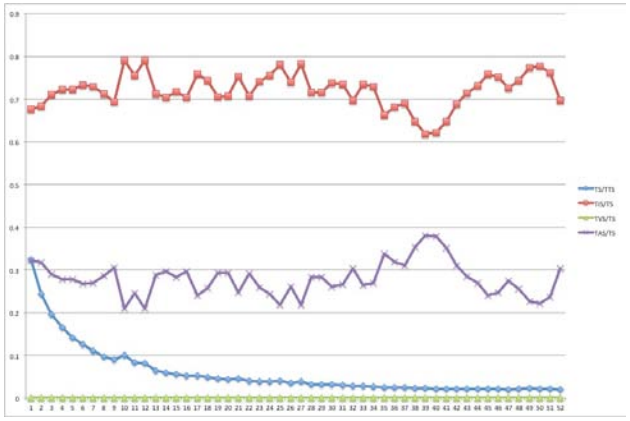


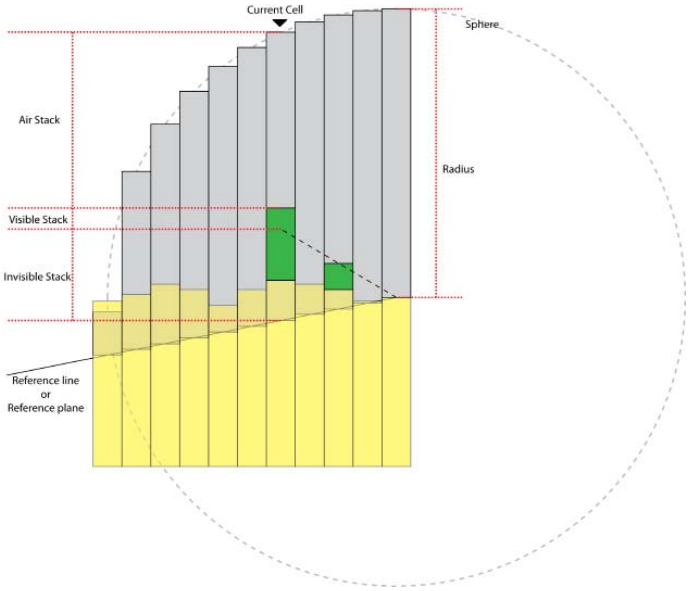
Purpose

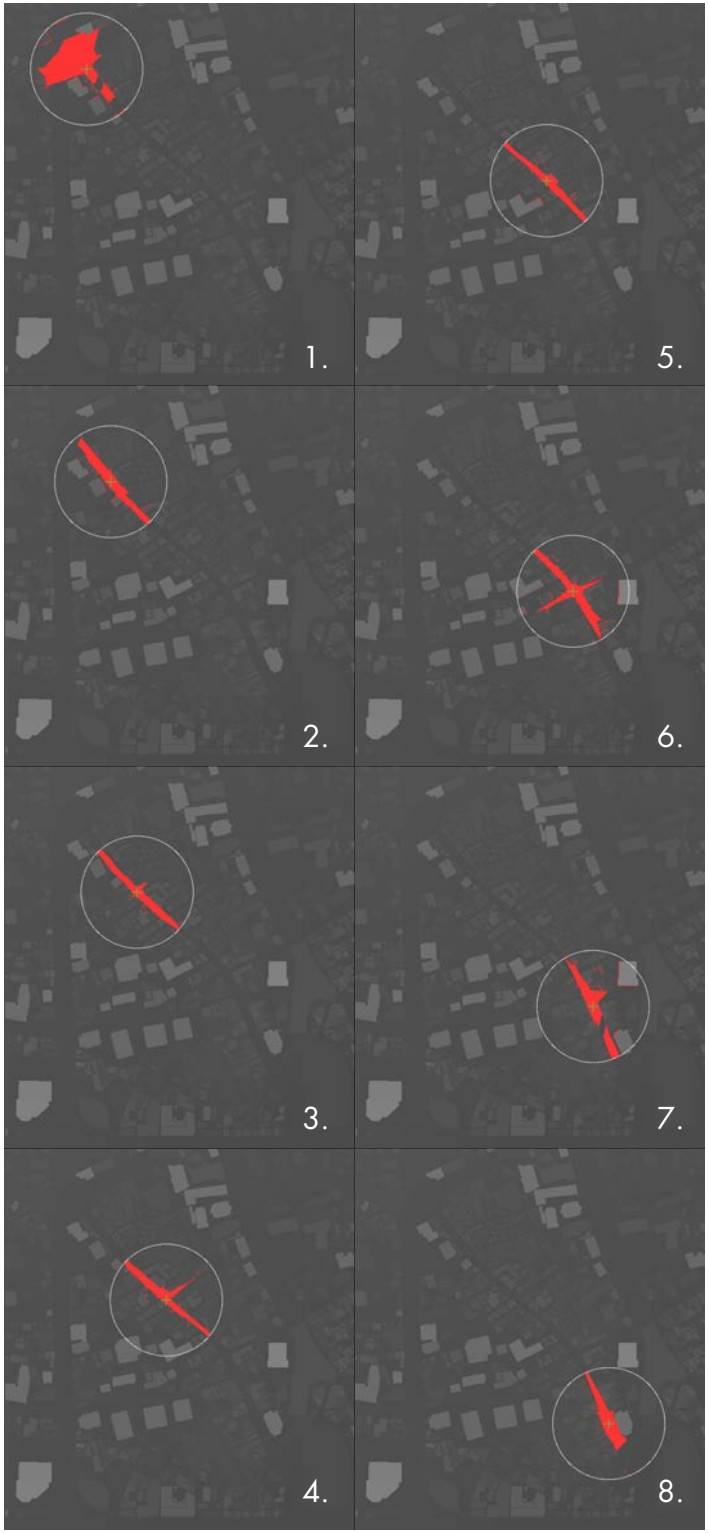
Intuitive visual queries such as “How much area near this position can be seen?” and “How far can be seen?” are one of the most important design requirements for both architect and urban planner to ask and solve in their design activities. Human’s visual perception is critical when it comes to the environmental assessment. In reality it is frequently witnessed that future dwellers of the newly built want to have a good view on the one hand, neighbors want to have less impact from it on the other hand. Especially in architecture and urban design process where many stakeholders concern, quantitative analyses are necessary to acquire a satisfactory design entry. To cope with the anticipated conflict that can be induced from visual factor, municipality is requested to set careful visibility guideline. In these context designers need to be equipped with more convincing analysis tools.

It is suggested that Viewsphere analysis method is important in that it gives users quantitative visibility analysis in urban environment (Yang et al., 2007). In this research, I propose Viewstack analysis method derived from viewshed and viewsphere analysis method for urban design evaluation based on GIS.



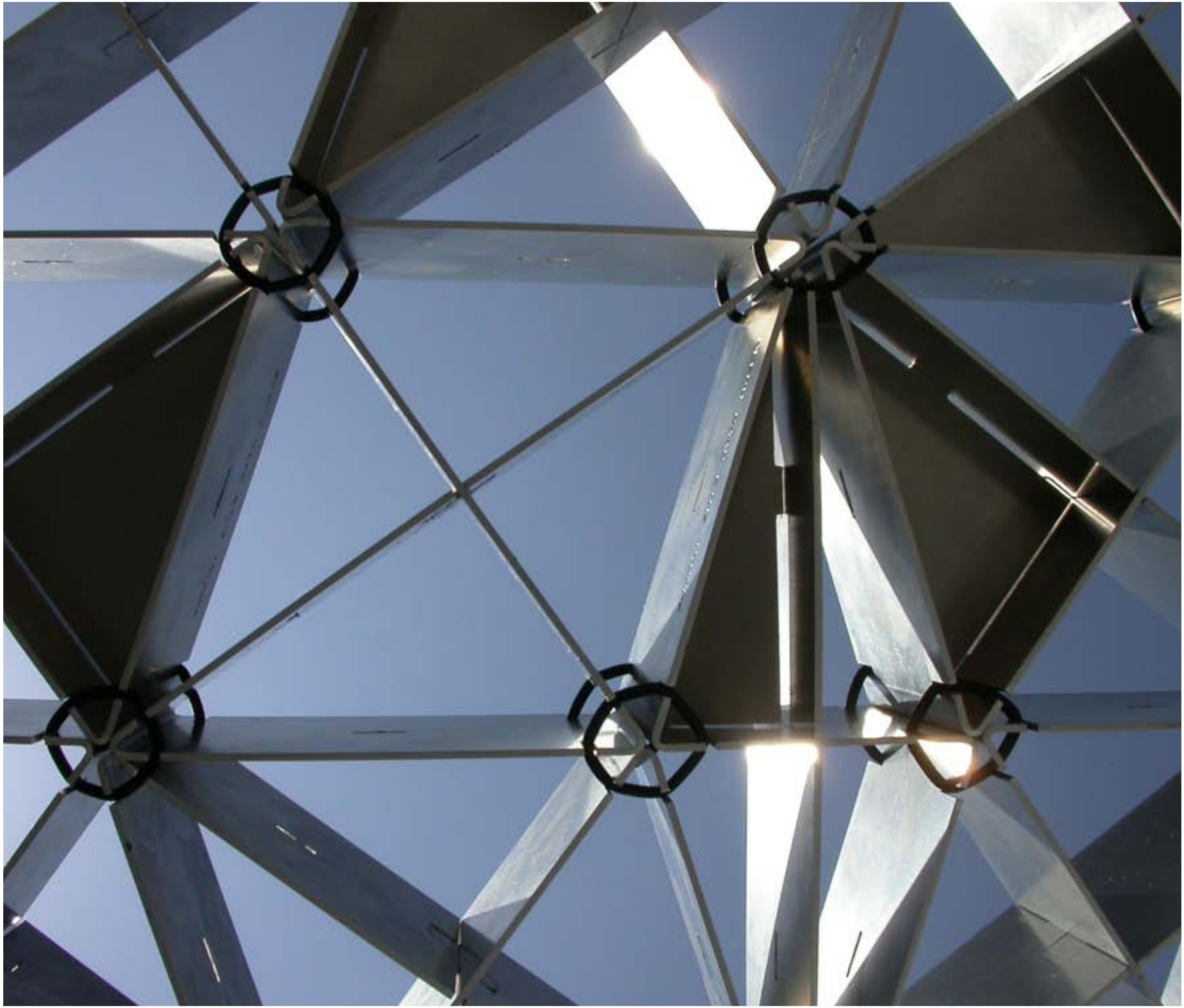


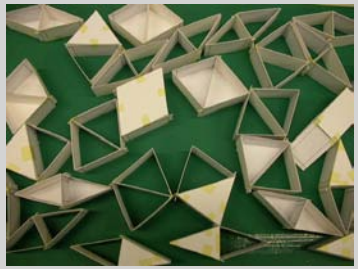
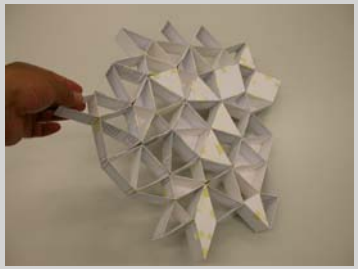


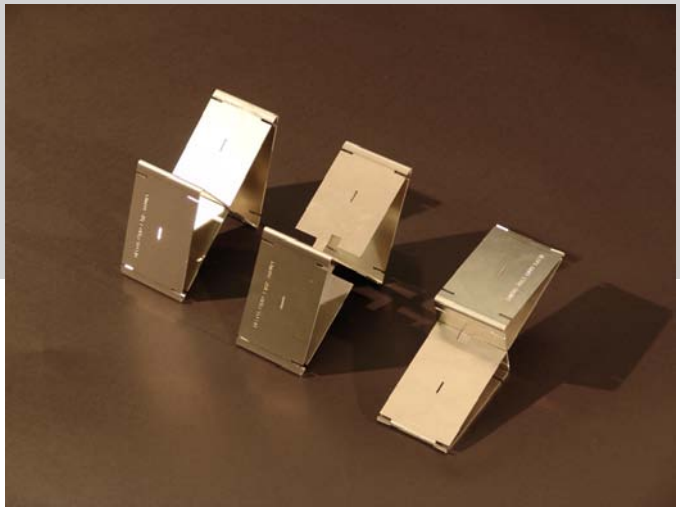


03

/SUMMER 2007
/DESIGN WITH METAL SHEET
/PARAVENT PROJECT

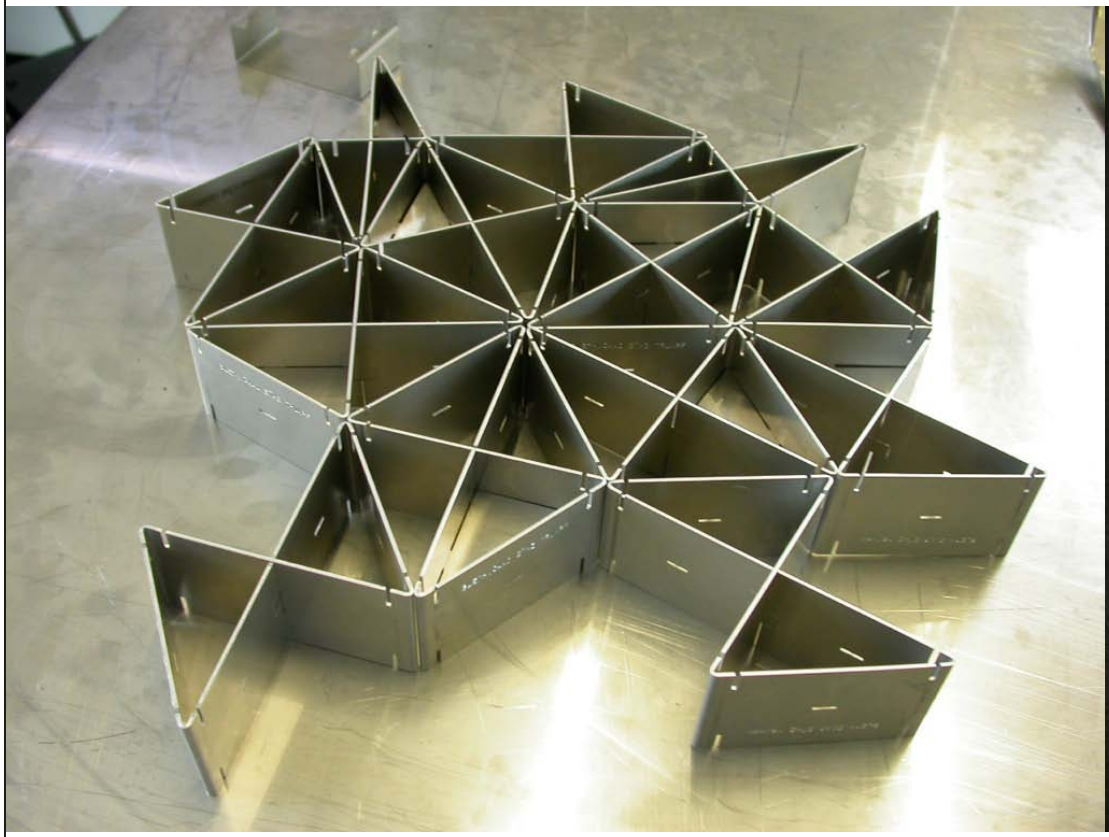
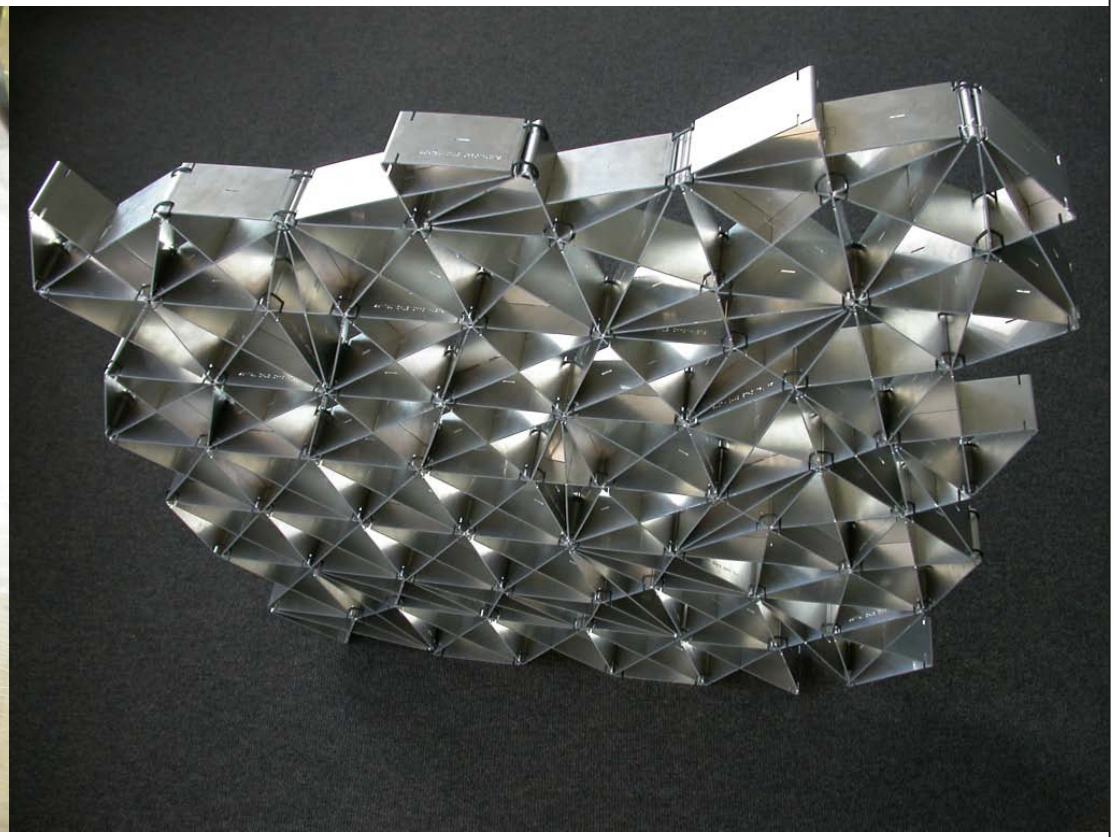




Intention

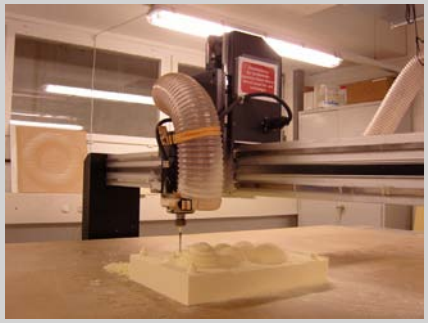
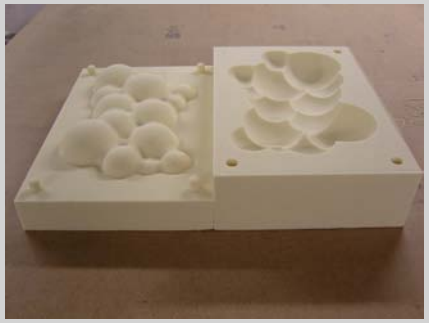
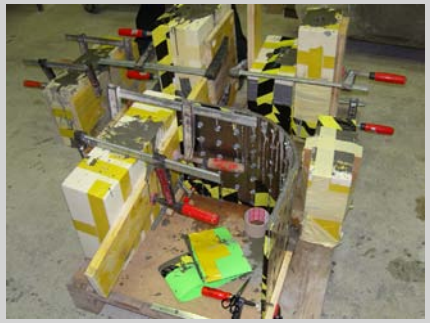
As a final project of an introduction CNC manufacturing technologies for metal sheet, a penrose pattern for a building facade was chosen in order to produce endless configuration by joining two assembly parts. Through multi-level workshops, CNC laser cutting, welding, folding and bending technologies were practiced and design, construction and CNC production of several designs were exercised.

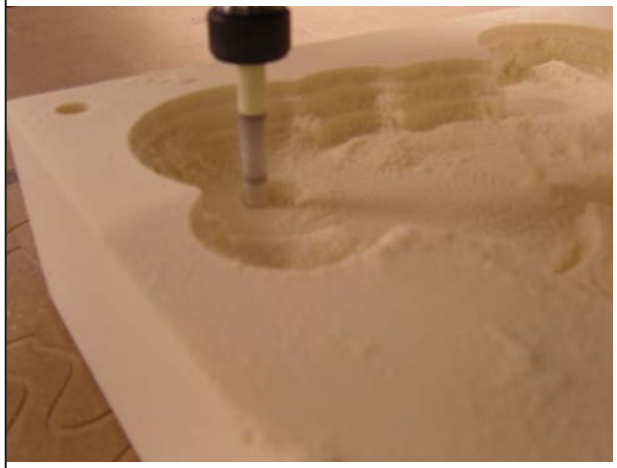



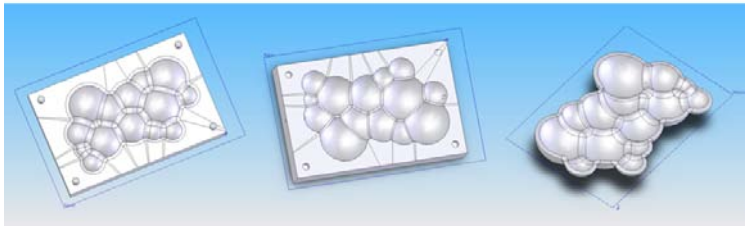
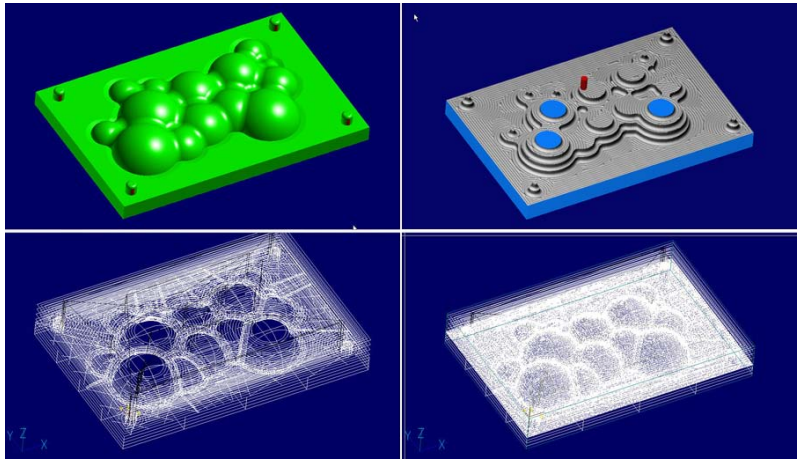
04

/SUMMER 207
/CONCRETE FRUIT BOWL
/RAPID PROTOTYPING

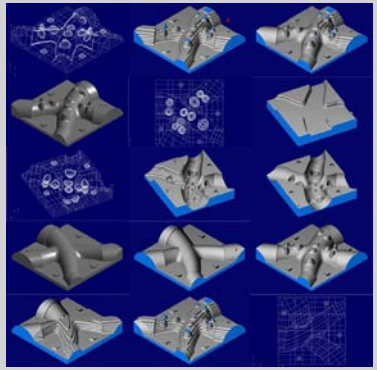
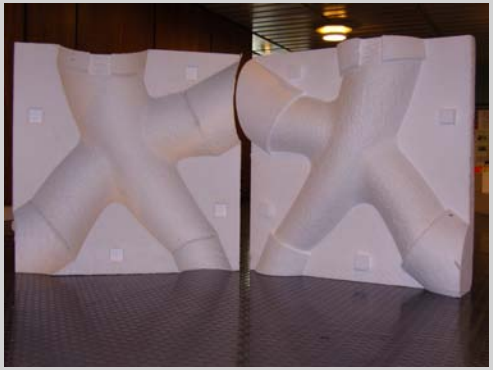
Concrete Fruit Bowl



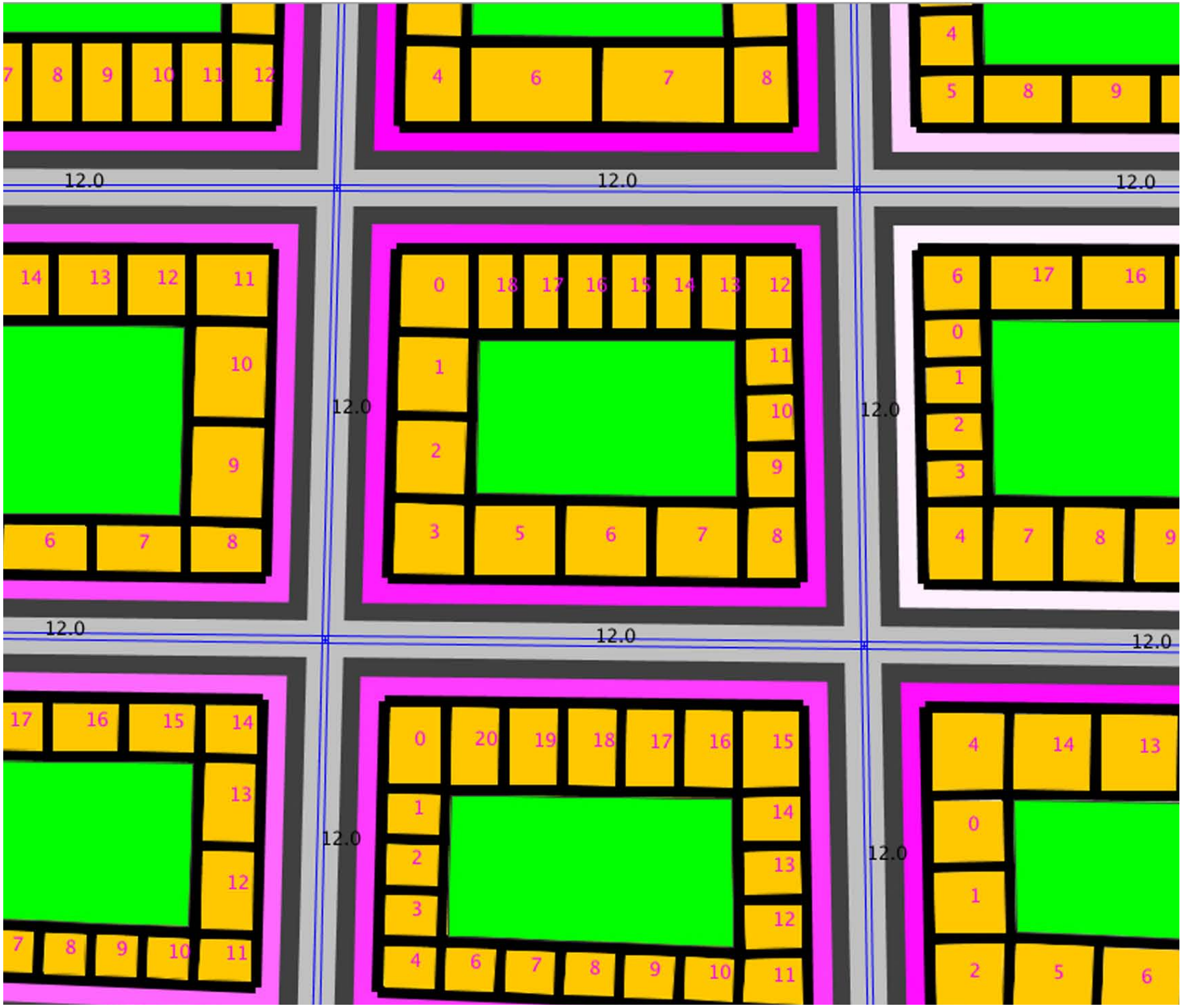



Concrete Chair

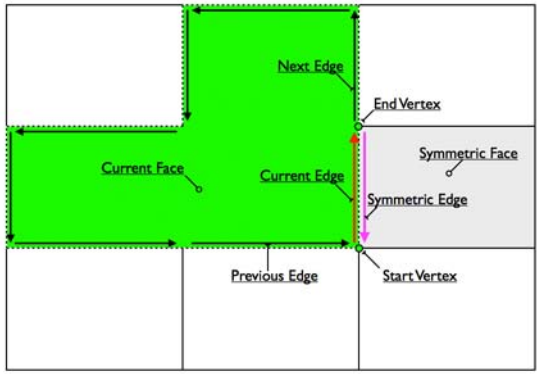

05

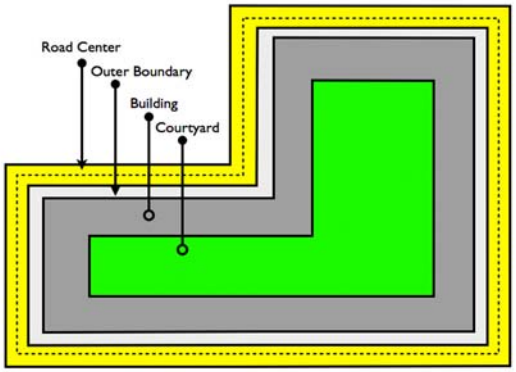
/2010-2011
/GENERATIVE TOWN DESIGN
/JAVA PROJECT

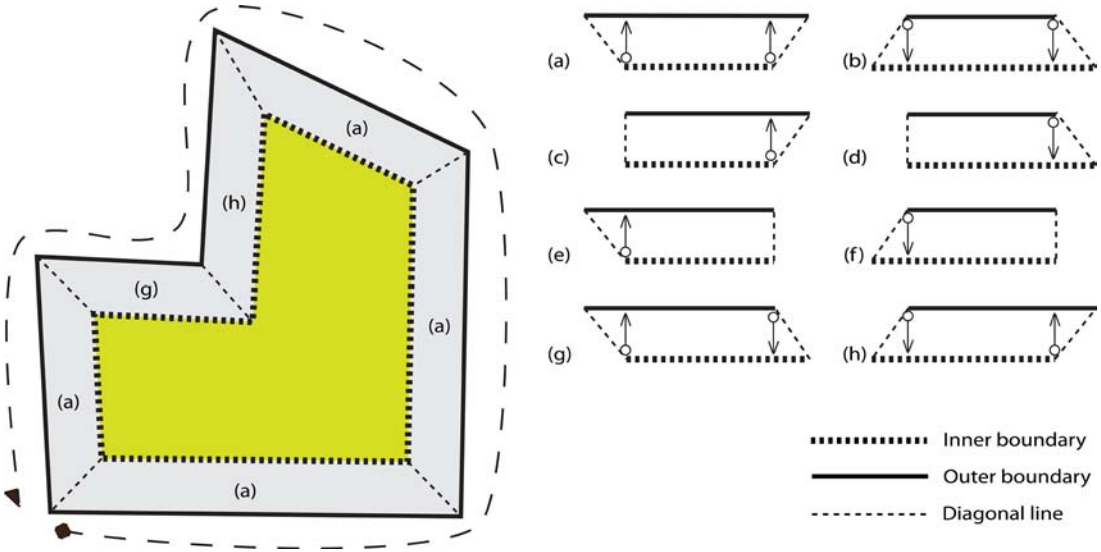


Abstract

The goal of this paper is to present the feasibility computing tool for the perimeter block housing design in early design process. Firstly, the paper describes briefly issues of perimeter block housing focusing on block design cases of Seoul. Secondly, constraints and requirements of perimeter block housing are analysed and formulated based on specific zone ordinance and regulation. Thirdly application of half-edge data structure is presented for interconnected geometric problem solving. Fourthly, multi-objective optimization algorithm developed is shortly explained as problem solving method. Finally, feasibility-computing software using Java object oriented programming is developed. This can contribute to the tool development that can generate, optimize, evaluate and visualize perimeter block housings in early phases of design process by providing reliable design solutions for stakeholders.





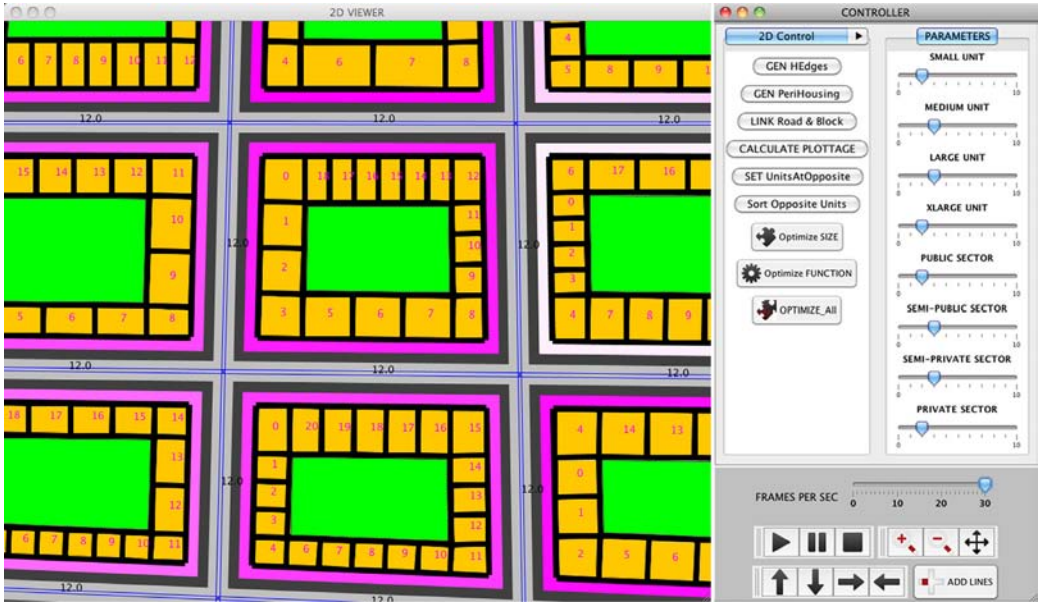


Main Purpose

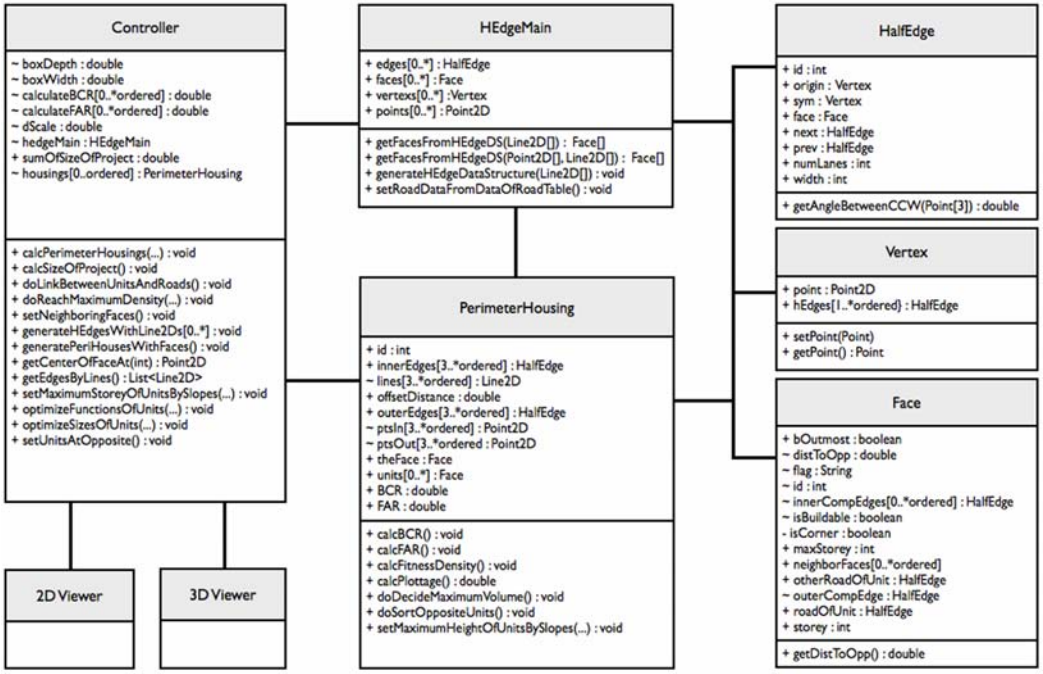
Urban town housing is a kind of perimeter block housing that is studied as an urban development model in Seoul. Urban town house in Seoul is defined as a five to seven stories perimeter block housing based on zone ordinance. One of the reasons why it received interests except for aforementioned merits is that it can escape the monotonous apartment complex development that have been considered most dominant building type while sustaining pre-existing urban structures such as block and road (Kim et al., 2010).The main purpose was to determine and quantify the factors that influence the shape and size of the final building block of an perimeter block housing to establish the relationships between the client's requirements, the building code constraints, and the architectural practice in the early design stages. In the end we developed a feasibility computing software to support the creation of optimized, parametric allocations of space during periphery block housing design. The tool was able to suggest feasible design alternatives by flexible alternations of design parameters. Moreover it was independently developed as open-end software that can be used to similar occasions more broadly. In order to accomplish the purpose, constraints and requirements that must be considered with was summarized at first. Next, halfedge data structure that can solve complex geometrical problems of perimeter block housing was introduced and several important algorithms were invented subsequently. Agent-based modeling with multiobjective optimization was proposed as well.

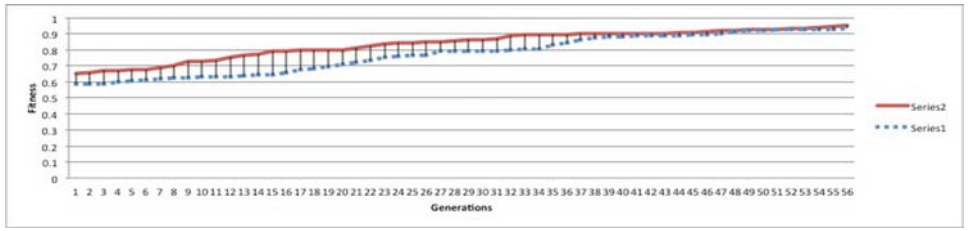
Halfedge Data Structure

Seeds to construct a basic data structure of the system are lines that representing roads and streets. Lines are manually drawn in the canvas or given from external file. Next step is to find intersections among lines. If a line has n respective intersection points (nodes) it is composed of n+1 respective line segment. If there is no line segment from a line, this line is not considered valid because it has no link to other line. As a result, nodes and links are obtained and represented as Halfedge Data Structure (HDS). To construct perimeter blocks, HDS is of importance in representing plane graphs of road (street) system (Brönnimann, 2001). As shown in figure 2, when HDS is constructed properly we can develop several functions that can be used in the system as follows: Firstly, neighboring road segments of one road segment can be pointed. Secondly, neighboring perimeter blocks of one perimeter block can be pointed. Thirdly, two confronting perimeter blocks that are divided by a road segment can be pointed by selecting the road segment. Fourthly, by choosing one corner point, corner-sharing perimeter blocks can be found. Fifthly, surrounding road segments of a perimeter block can be pointed.

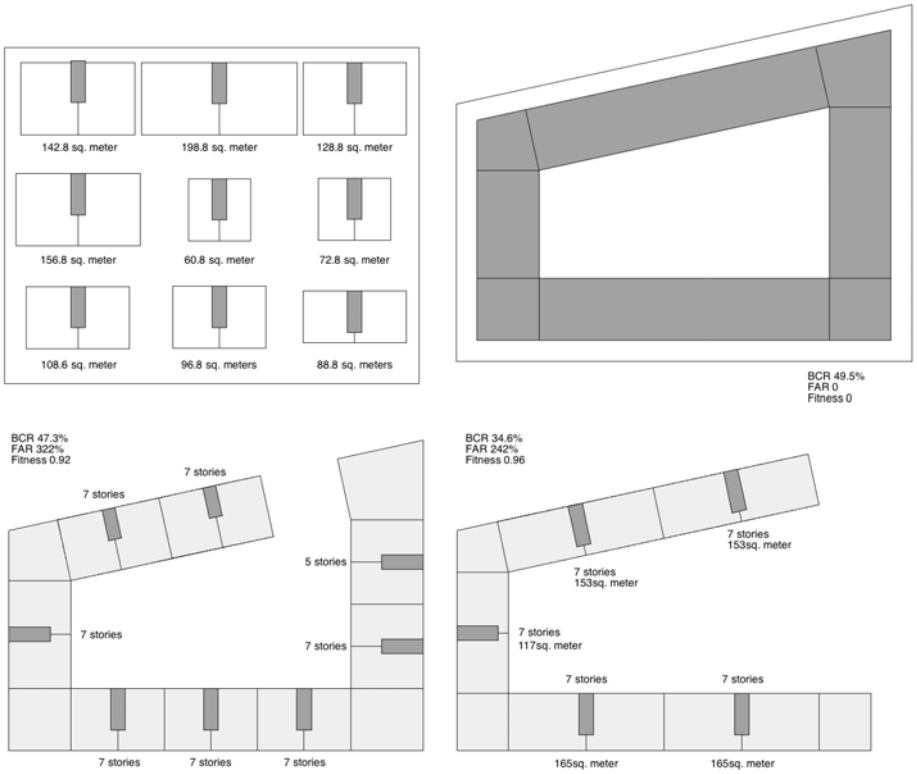


Agent Based Modeling





To approach this research from a computational point of view, an agent-based modeling (ABM) is suitable for the periphery block housing design task. The new software is programmed as an independent tool using Java language. Here ABM is composed of decision-making entities called agents (Bonabeau, 2002; Macal and North, 2007). Each agent acts independently according to the rules and execute its pre-programmed behaviors such as repetitive competitive pro-actions and re-actions between agents. As drawn in figure 4, an agent– "PerimeterHousing" – that represents attributes of a building complex in a block is designed to solve problems that are related to physical constraints. Geometric tools by Schneider and Eberly (2002) are widely applied in order to treat complex geometric issues. Moreover architectural entities of this modeling highly resemble those of IFC (Industry Foundation Classes) architectural domain, which make it possible to interoperate with other BIM software.

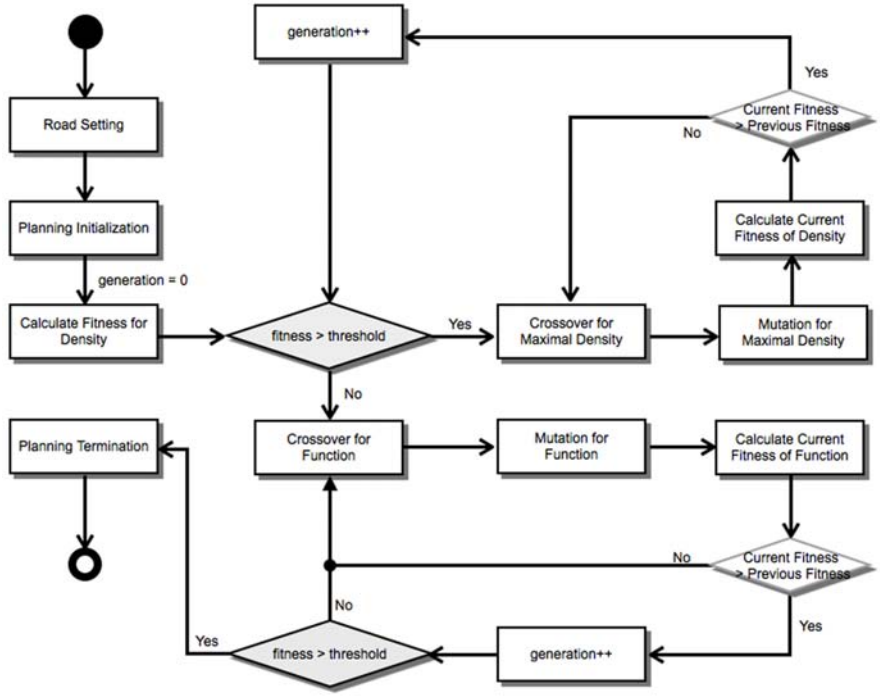


Case Study

There are differences in the size of blocks depending to completion period and local context, but mostly 40~60 meters in width and 120~160 meters in depth. Ideally to realize 7 stories urban town house [if two buildings of 21 meter high are planed which are lined with widened incident road of 12 meters width, more than 70 meters is required which is summed up of setback (10M), building depth (30M), and courtyard (21~30M)] that can maximize the feasibility following Korean building code, short edge of a block must be at least more than 60 meters in width for planning and more than 70 meters long for securing enclosure.
Images show a case design of a PBH in a test project. When the software initialize the optimization algorithm, a footprint of the PBH is calculated as figure 7 (top-right). Flats are pictorially represented on top-left image of figure 7. Figure 7 (bottom-left) is a final design (FAR: 322%, Fitness: 0.92, gFAR: 350%) of the PBH. Figure 7 (bottom-right) is other final design (FAR: 242%, Fitness: 0.96, gFAR: 250%).

Optimization

Experience of early attempts at the solving process promotes the development of new approaches. Multi-objective optimization algorithm as represented in Figure 5 is designed to manage a PBH project. Genetic Algorithm (GA) that mimics the design process of imaging, presenting, and testing (Zeisel, 1984) is utilized as a design generation and a problem solving strategy. The merit of GA is its possibility to draw feasible solutions that do not violate the constraints while maintaining their diversity (Fonseca and Fleming, 1998; Coello, 2001). It is sometimes possible to deduce unexpected ones that are distinctive. In almost all cases, it is impossible to obtain the best performance across all the objectives concurrently. Hence two series of GA are operated. The one called density fitness function (DFF) is a tool that concerns the overall density in a project. The other participates in the functional space allocations according to a space program. Although the proposed computational tool is originally designed and implemented for periphery block housing in Seoul, there are potentialities to generalize the tool for the application to other similar contexts. During early design stages, designers who use the tool to obtain the schematic volumes can minimize the risk of infringing the building codes. Moreover we can positively employ the tool to generate innovative design alternatives.



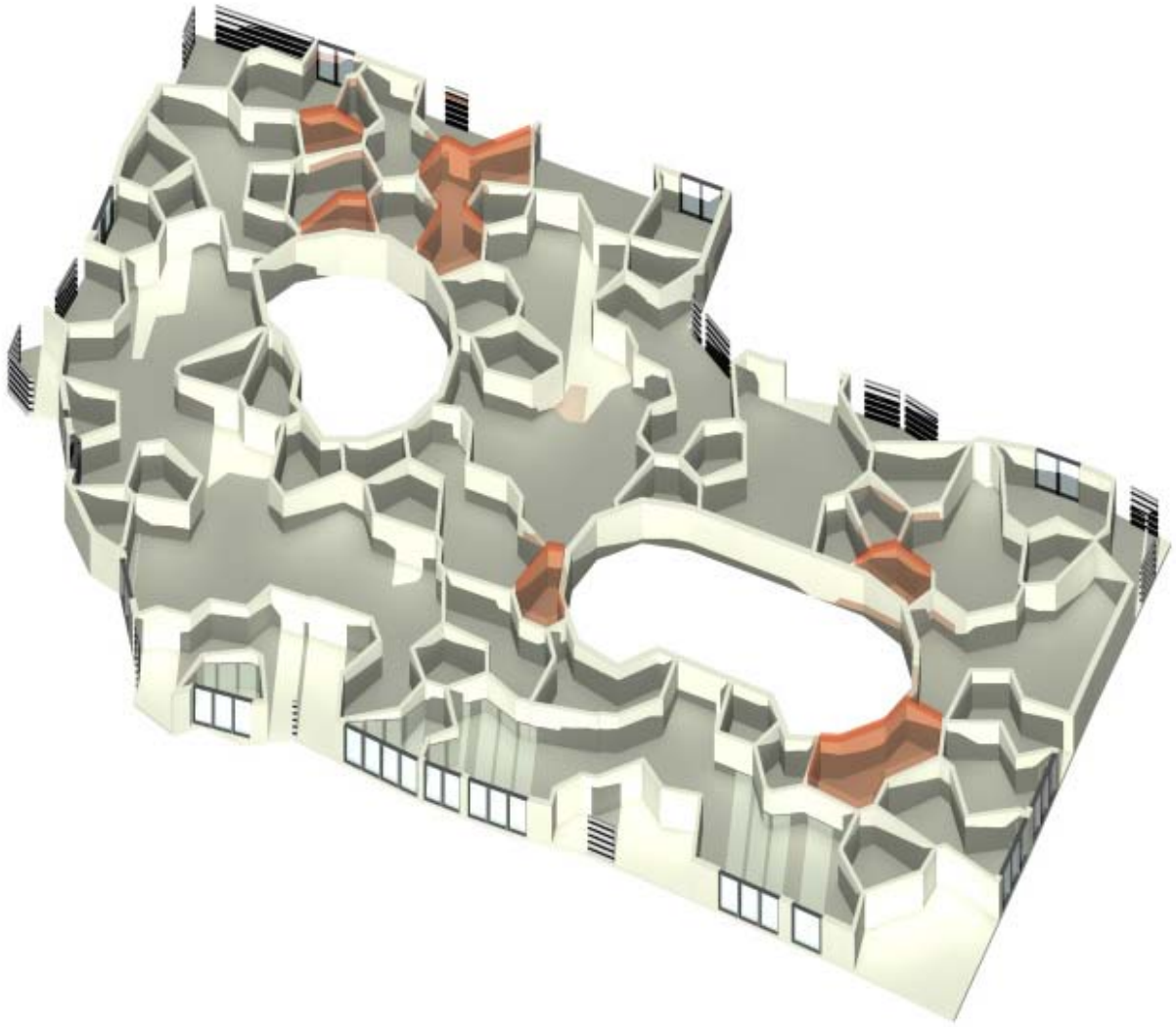
06

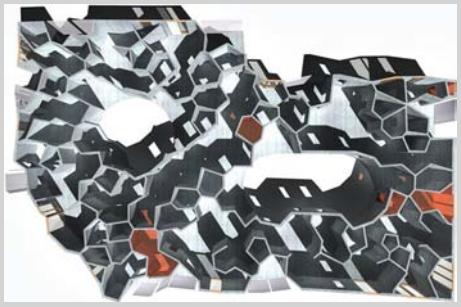
/SUMMER-WINTER 2007
/GENERATIVE DESIGN
/MAS THESIS PROJECT

Self-Organising Architectural Design using 2-d Voronoi Diagram

The motivation to inaugurate this research is to develop a computational tool using design precedents in the creation of the avant-garde architectural layout design. The hypotheses behind the study performed could be summarised by - provide the analysis framework of architectural precedents in a bottom-up way; provide a problem solving method between concept and form (geometry); compare design process

The purpose of this study is to generate architectural layout design. For my study, First, we find suitable methodology to use design precedents from literature. Second, the relationships among design concepts and forms for algorithmic architecture are sought. Finally, comparison of design process between conventional design process and a new generative design process is made as an outgrowth.



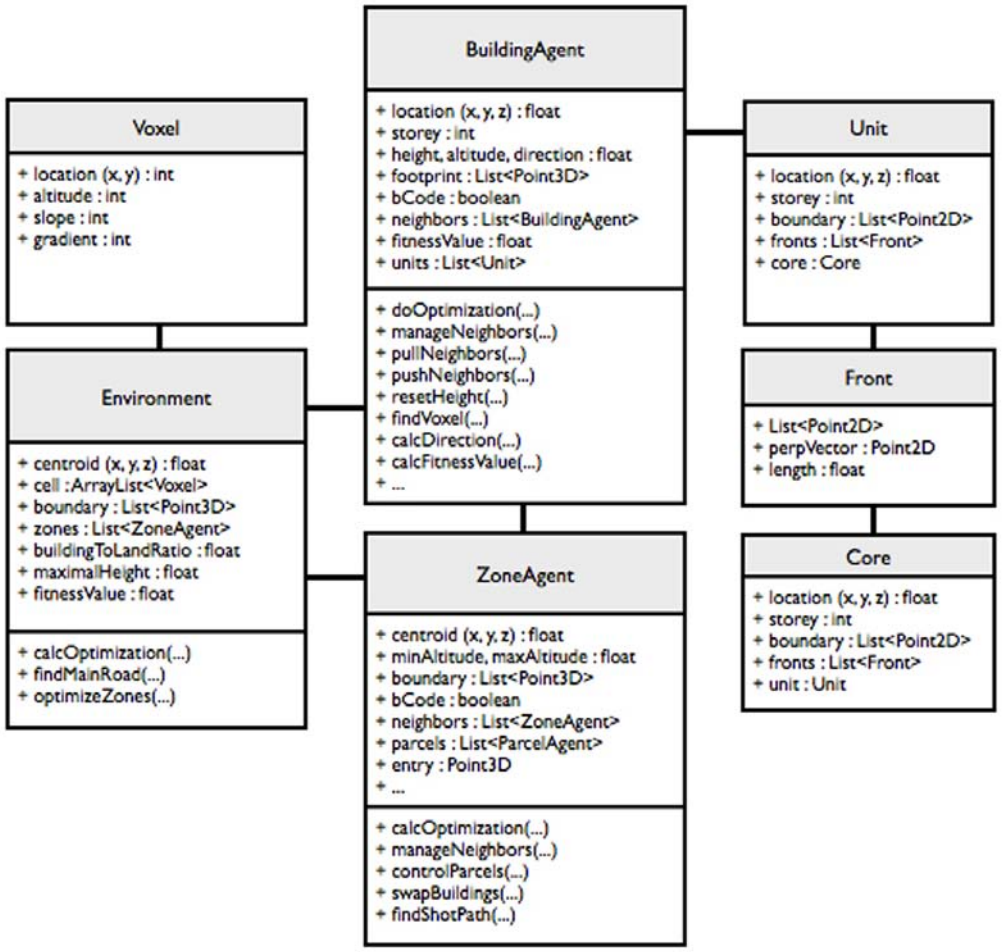


07

/2008-2010
/TOWN PLANNING
/JAVA SOFTWARE DEVELOP-
MENT PROJECT

A complex adaptive residential quarter planning software is developed using Java programming language and targeting at configuring the tower-type apartment in a dense area during early design stage. Rules are analyzed and formulated based on building code and zone ordinance. Moreover we develop an agent-based modeling with multi-objective optimization algorithm. In this modeling, each agent acts independently according to the rules that are designed to solve the complex geometric problems that are related to physical constraints.

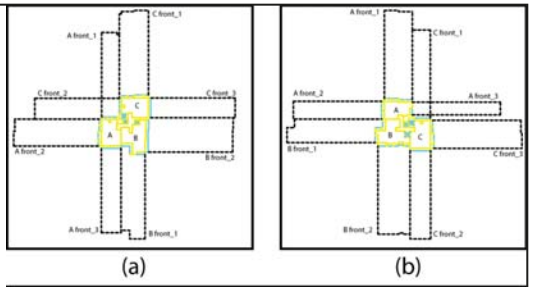
System Framework

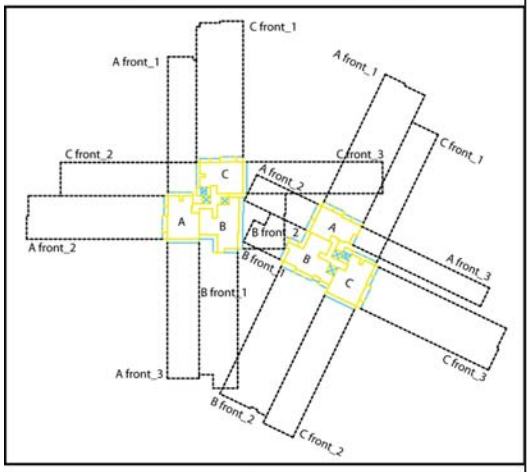


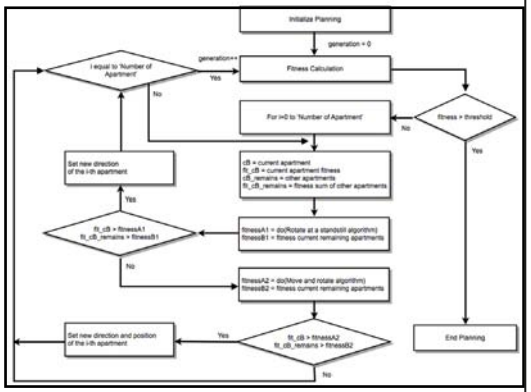
Residential quarter planning in itself is a multi-objective problem similar to many other real-world problems. Design decision-making process involves varied tradeoffs among multiple objectives, which should be optimized sim-ultaneously. Depending on the information size of the target problem, the amount of both incoming information and outgoing information that is pro-cessed with by a designer is enormous. During early design stage that is spe-cially admitted to be one of the most important steps, it is requested to ac-quire many tangible design solution candidates (Zeisel, 1984; Simon, 1996; Lawson, 1997).

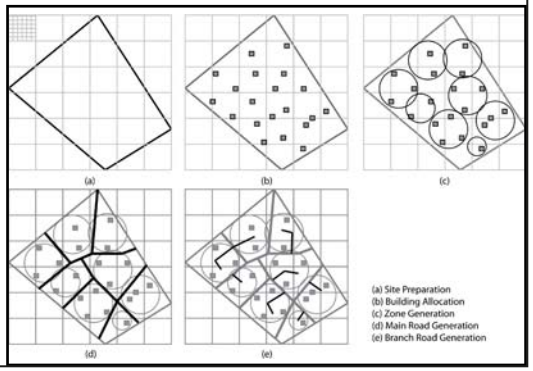
The reasons for that are to minimize the risk to select wrong idea and to obtain more feasibly creative design. However it is not easy to acquire satis-factory design spaces within given duration when human's cognitive ability becomes the major resource to navigate many interme-diate design alterna-tives. Solution evalua-tion that is designer-led with very little computational support can be largely subjective as well. Multiple design objec-tives to satisfy and various constraints to comply with are insufficient and are violated re-spectively in some circum-stances. Those reasons lead to the limitation of conceptual design decisions. In this research, we develop a residential quar-ter planning software based on multi-objective optimization procedure.

Agent-based modelling (hereafter ABM) is utilized not to provide a description of a specific system but to solve the residential quarter planning by adapting the ABM theory. ABM is composed of autonomous decision-making entities called agents (Macal and North, 2007). Each agent acts independently according to the rules and execute its pre-programmed behaviours. Repetitive competitive pro-actions and re-actions between agents are important features of the proposed ABM similar to other general ABMs. Two kinds of agent are modelled; one is building agent and the other are zone agent. Environment of the ABM is modelled based on GIS information of the target site. However instead of precisely designing learning system that is found of importance in general ABM, specially designed rules are utilized to help each agent find a suitable behaviour and to assist a program operator to obtain meaningful planning. Emergent phenomena that are captured at the local area between neighbouring building agents can explain the clear relationship between agents. Global phenomenon can be thought as design proposals that we wish to obtain at the end. Whether the outcomes are feasible or not can be decided by optimization technique. When the optimization process is terminated, planning branch roads can be generated.



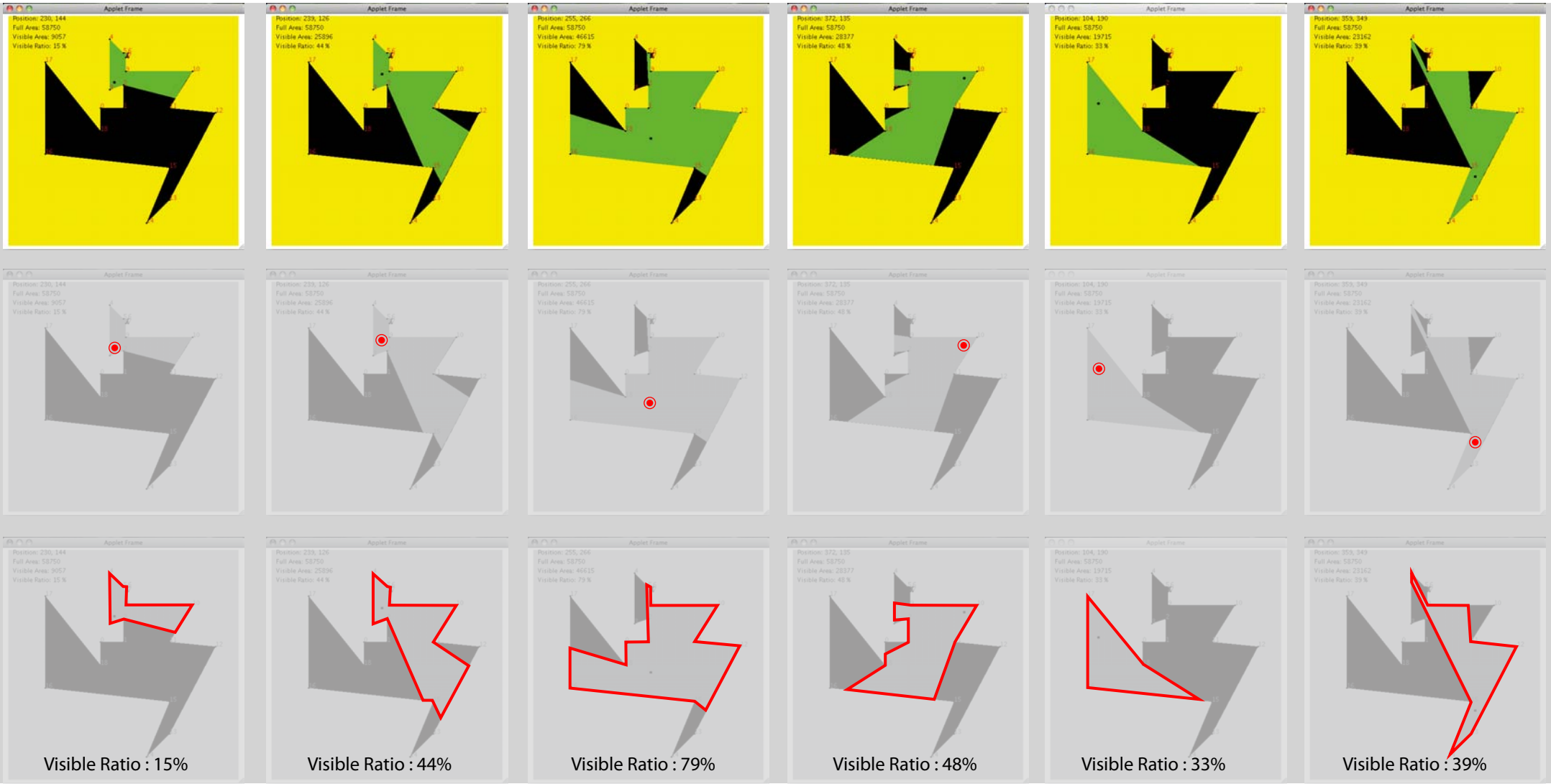


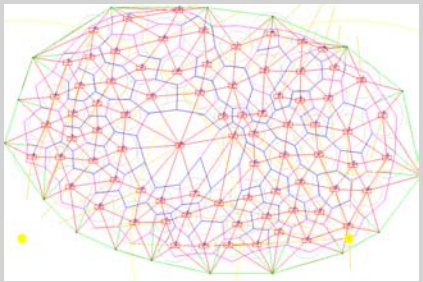




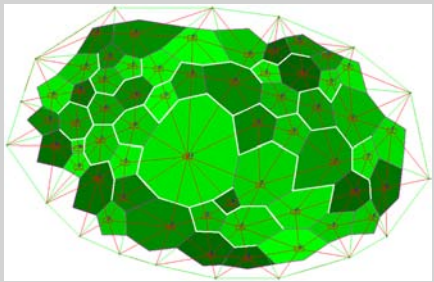
Indoor Visibility Analysis Software (Spring 2009)

This indoor visibility analysis software was developed using JAVA programming language based on the Joe and Simpson's paper (Joe, B., and Simpson, R. B., Corrections to Lee's Visibility Polygon Algorithm, BIT 27, 1987, pp. 458-473). When points of a boundary polygon and a standing point are given, this software can calculate visible area from whole area and visualize it automatically. This applet used a thread to generate visible area in realtime. This tool can be extended by combining not only an outdoor visibility analysis tool (Viewstack) but also other analysis tools. For better indoor visibility analysis, this tool can be developed to deal with diverse input information such as objects in a room and more complex geometry.

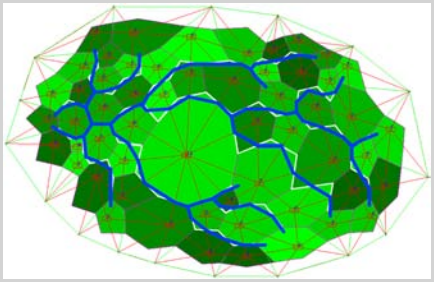




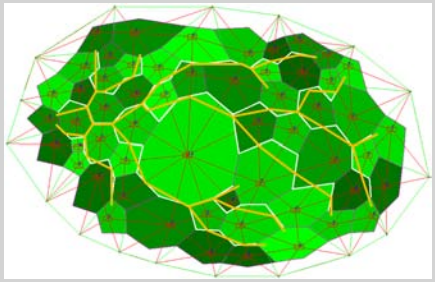
1.



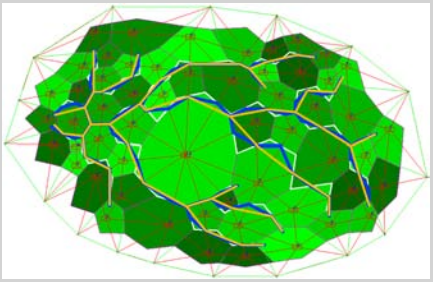
2.



3.



4.

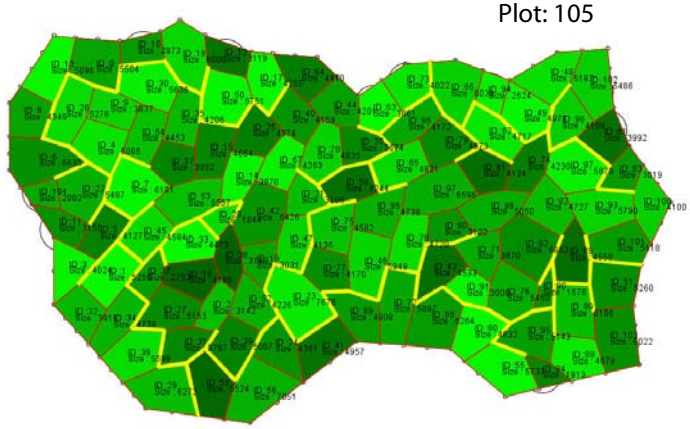


5.

Plot generation

With the centre points of circles and residential quarter boundary, Voronoi diagram (Berg et al., 2008) can be obtained and each cell boundary segment is reconstructed as an acyclic graph for path finding algorithm. The acquired convex polygons are reshaped to rectilinear shapes by intersection condition. Generally three points meet at an intersection point and one of the smallest angles can be modified to the right angle. Which results in a relatively straight line and it can contribute to the economic planning of the branch road system.
The process of building road systems among zones is dependent on both the distance from each zone to access points of the residential quarter and the gradient angles of graph edges constructed from zone boundaries.
A connecting path from a zone to one nearest access points can be efficiently computed by Breadth-First Search (BFS) algorithm (Goodrich and Tamassia, 2010). If necessary, closed loop road can be made by linking two neighbouring graph vertices. The end product of this algorithm is a series of road segments applied with breadth.

Generation: 2042
Fitness: 0.0
Plot: 105



Generation: 1730
Fitness: 0.0
Plot: 102

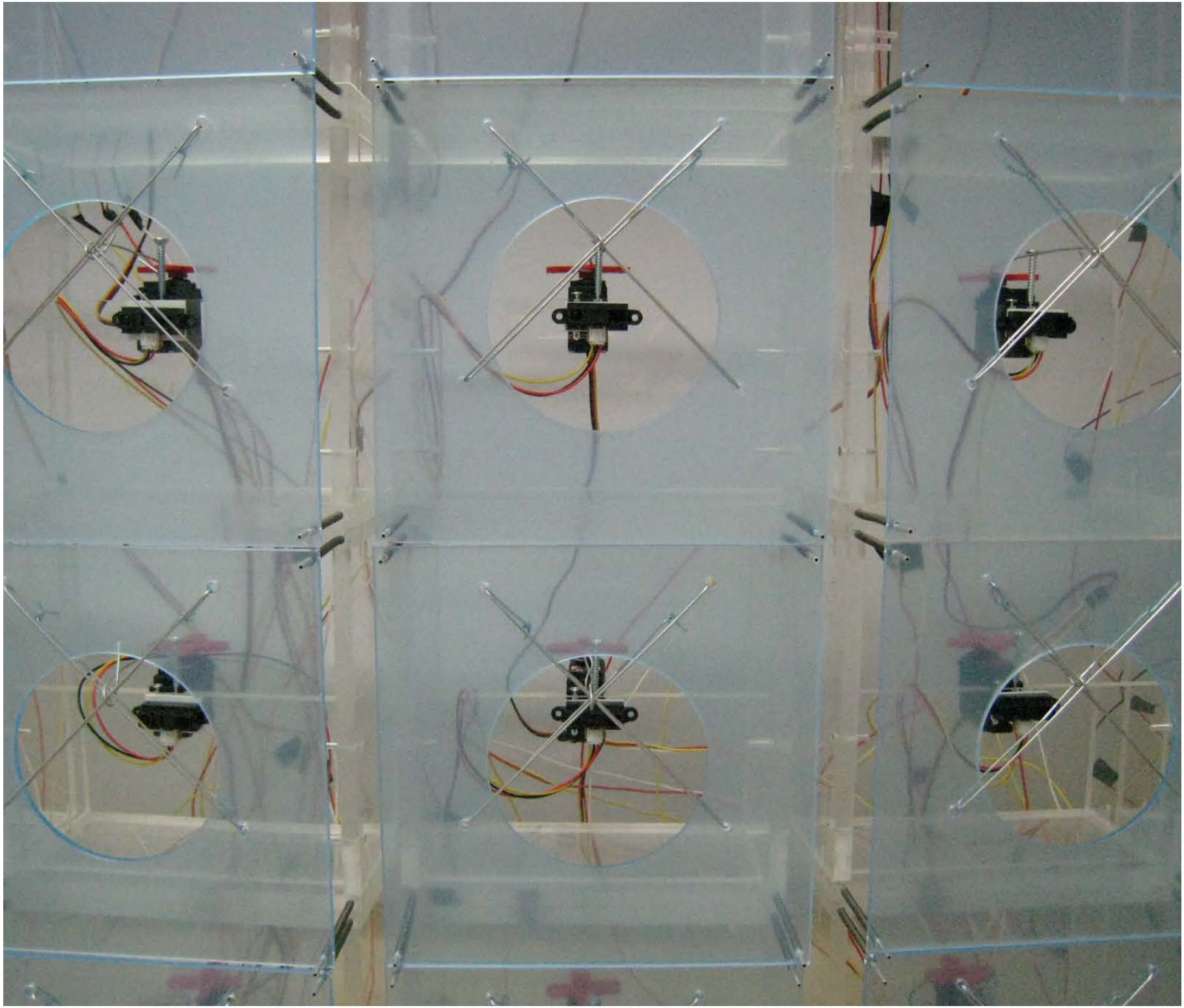
Generation: 1762
Fitness: 0.0
Plot: 102

Generation: 2470
Fitness: 0.0
Plot: 218

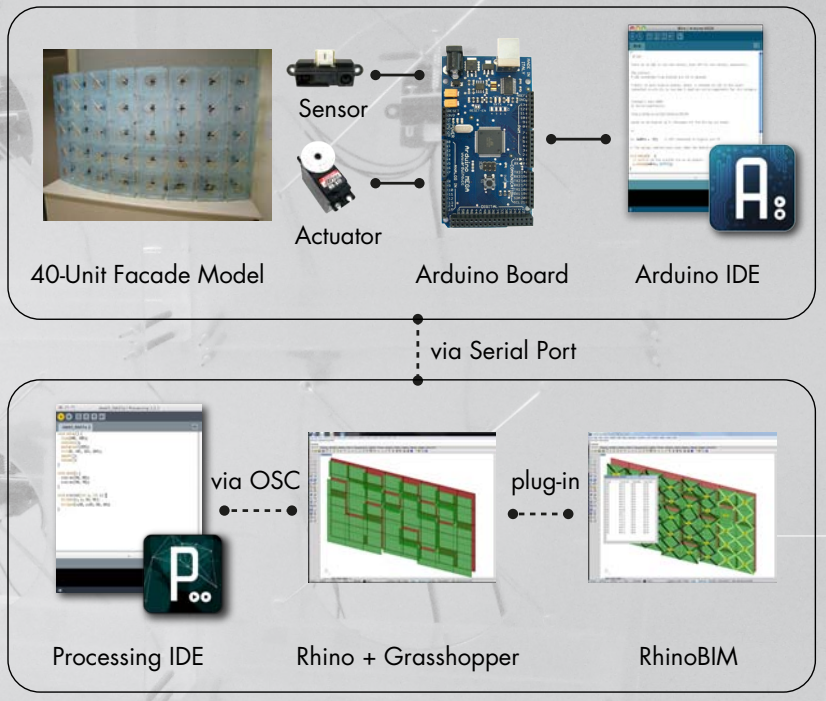
Generation: 1994
Fitness: 0.0
Plot: 220

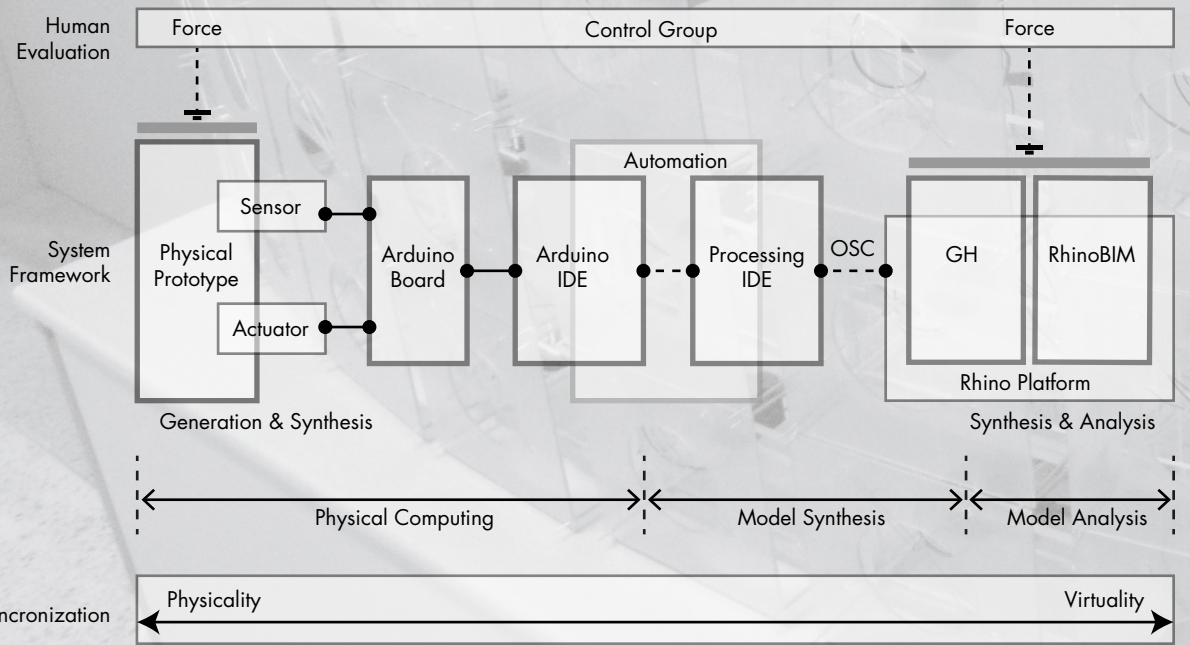
08

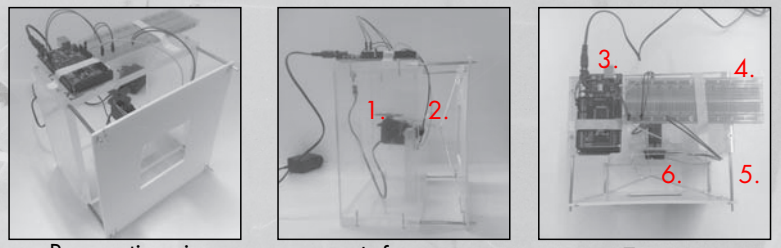
/SPRING 2012
/GENERATIVE DESIGN
/INTERACTIVE WALL PROJECT
/PROF. KIM SUNG-AH



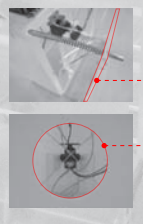
System Framework

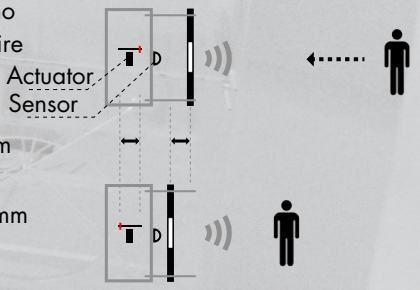






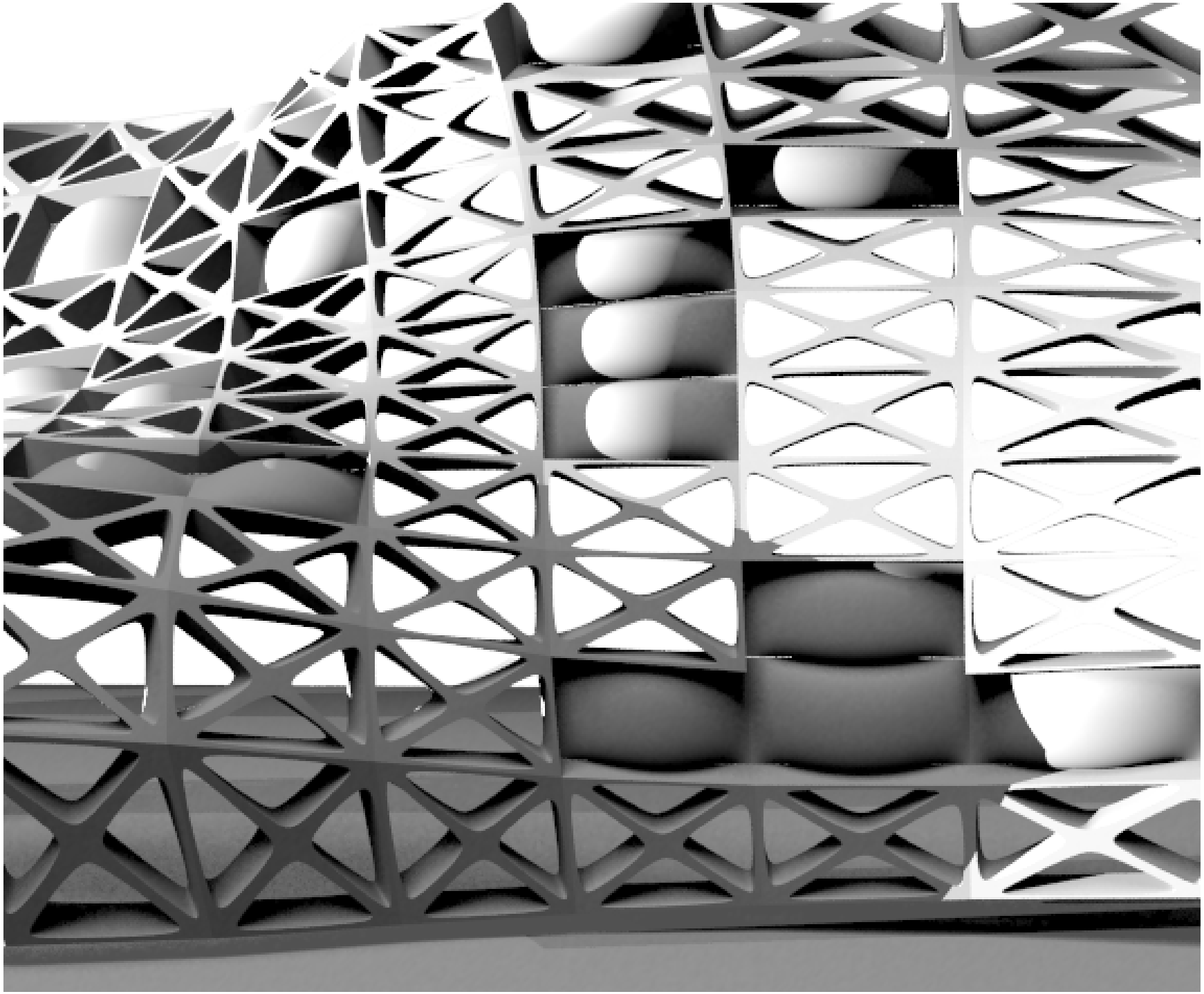
1. Actuator 2. Sensor 3. Arduino
4. Bread board 5. Spring 6. Wire



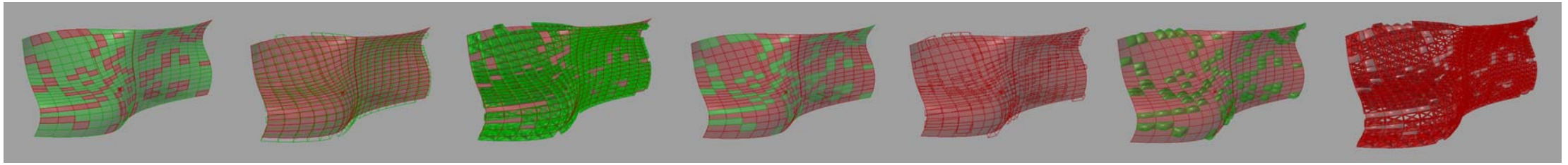


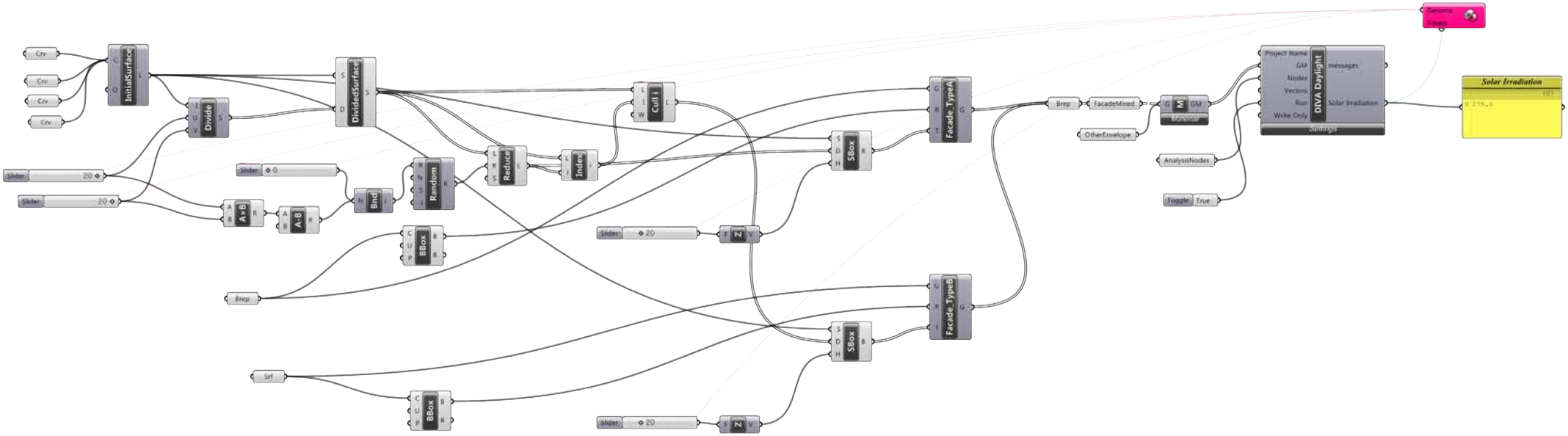
09

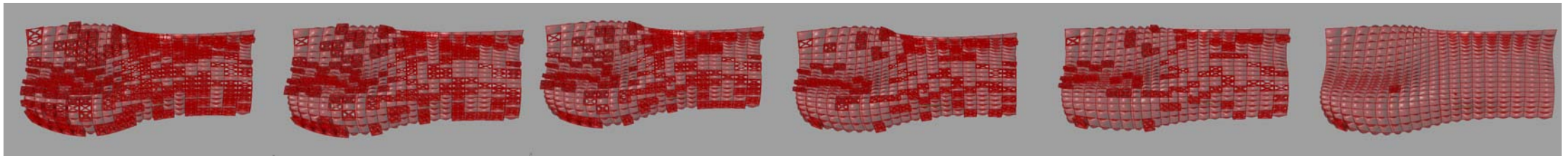
/SPRING 2012
/GENERATIVE DESIGN
/GH+GALAPAGOS



Facade design using Grasshopper

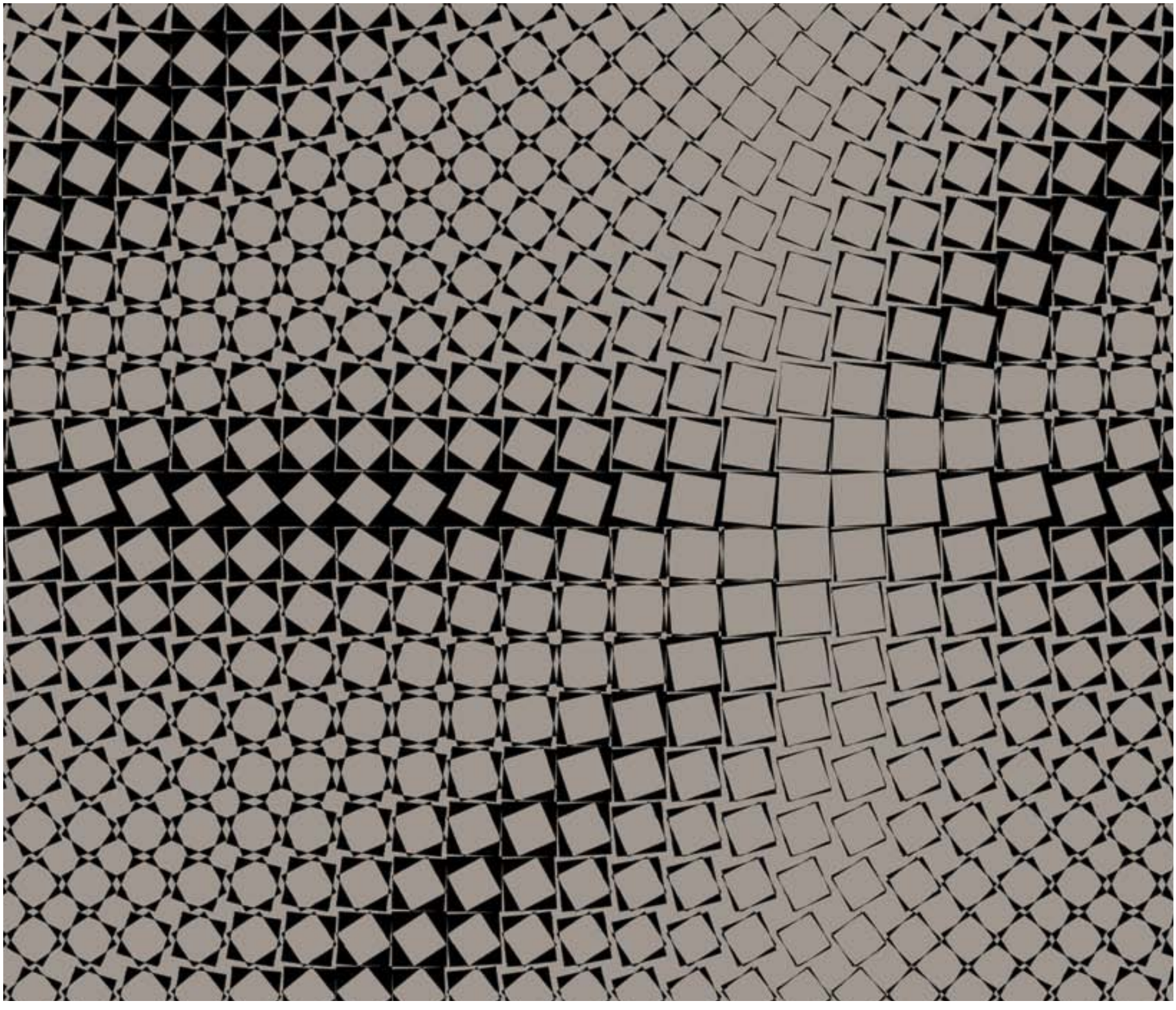




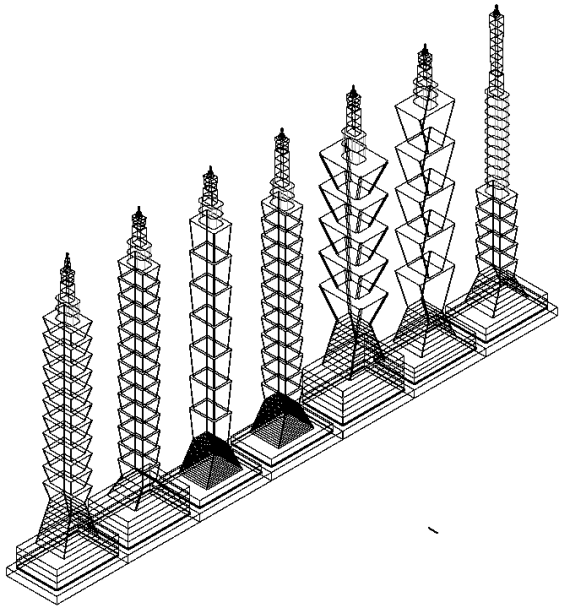


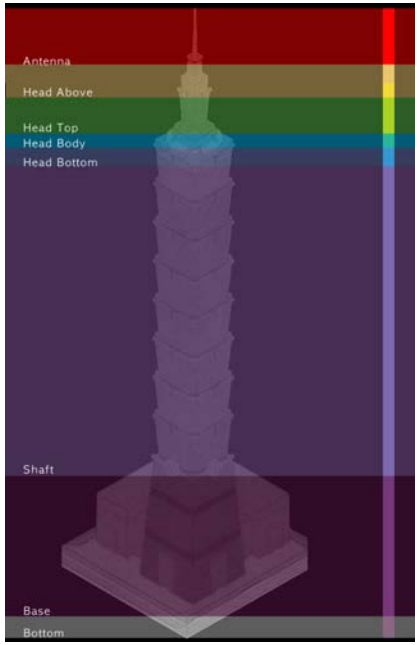
10

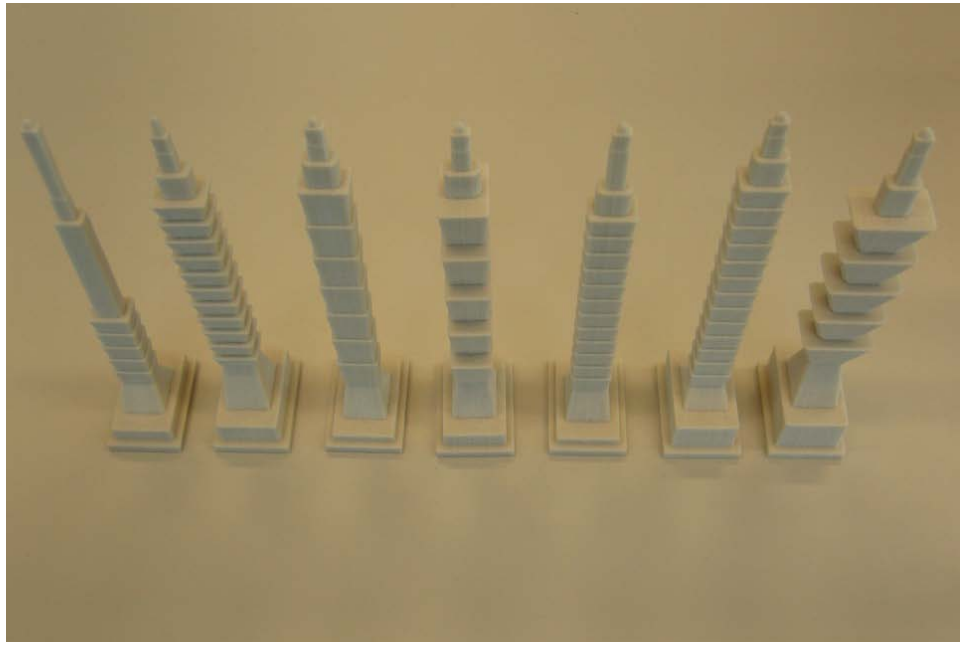
/FALL 2006
/DESIGN BY NUMBER
/VECTORWORKS SCRIPTING



Taipei 101 parametric skyscraper







Parameters

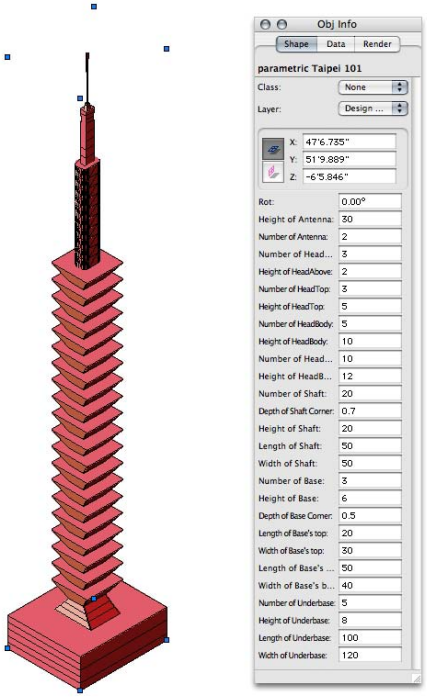
Parameters

Name	Alternate Name	Type	Default
AntennaHeight	Height of Antenna	Number	30
Numb_antenna	Number of Antenna	Integer	2
Numb_headAbove	Number of HeadAbove	Integer	3
HeadAboveHeight	Height of HeadAbove	Number	2
Numb_headTop	Number of HeadTop	Integer	3
HeadTopHeight	Height of HeadTop	Number	5
Numb_headBody	Number of HeadBody	Integer	5
HeadBodyHeight	Height of HeadBody	Number	10
Numb_headBottom	Number of HeadBottom	Integer	5
HeadBottomHeight	Height of HeadBottom	Number	12
Numb_shaft	Number of Shaft	Integer	20
ShaftCornerDepth	Depth of Shaft Corner	Number	0.7
ShaftHeight	Height of Shaft	Number	20
Shaft2MyLength	Length of Shaft	Number	25
Shaft2MyWidth	Width of Shaft	Number	35
Numb_base	Number of Base	Integer	3
BaseHeight	Height of Base	Number	6
BaseCornerDepth	Depth of Base Corner	Number	0.5
Base2MyLength	Length of Base's top	Number	20
Base2MyWidth	Width of Base's top	Number	30
Base1MyLength	Length of Base's bottom	Number	50
Base1MyWidth	Width of Base's bottom	Number	40
Numb_underbase	Number of Underbase	Integer	5
UnderbaseHeight	Height of Underbase	Number	8
Underbase1MyLength	Length of Underbase	Number	100
Underbase1MyWidth	Width of Underbase	Number	120

Add... Edit... Remove Move Up Move Down

Press the Help key or Cmd+? for help.

Cancel OK



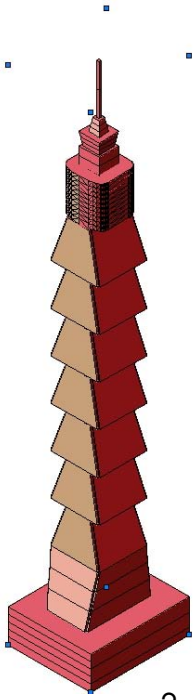
Obj Info

Shape Data Render

parametric Taipei 101

Class:	None
Layer:	Design ...
X:	47'6.735"
Y:	51'9.889"
Z:	-6'5.846"
Rot:	0.00°
Height of Antenna:	30
Number of Antenna:	2
Number of Head...:	3
Height of HeadAbove:	2
Number of HeadTop:	3
Height of HeadTop:	5
Number of HeadBody:	5
Height of HeadBody:	10
Number of Head...:	10
Height of HeadB...:	12
Number of Shaft:	20
Depth of Shaft Corner:	0.7
Height of Shaft:	20
Length of Shaft:	50
Width of Shaft:	50
Number of Base:	3
Height of Base:	6
Depth of Base Corner:	0.5
Length of Base's top:	20
Width of Base's top:	30
Length of Base's ...:	50
Width of Base's b...:	40
Number of Underbase:	5
Height of Underbase:	8
Length of Underbase:	100
Width of Underbase:	120

1.



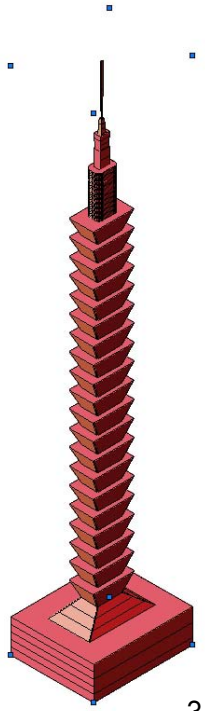
Obj Info

Shape Data Render

parametric Taipei 101

Class:	None
Layer:	Design Lay...
X:	47'6.735"
Y:	51'9.889"
Z:	-6'5.846"
Rot:	0.00°
Height of Antenna:	30
Number of Antenna:	2
Number of HeadAbove:	3
Height of HeadAbove:	5
Number of HeadTop:	3
Height of HeadTop:	5
Number of HeadBody:	2
Height of HeadBody:	10
Number of HeadBottom:	10
Height of HeadBottom:	5
Number of Shaft:	7
Depth of Shaft Corner:	2
Height of Shaft:	50
Length of Shaft:	30
Width of Shaft:	50
Number of Base:	3
Height of Base:	20
Depth of Base Corner:	2
Length of Base's top:	60
Width of Base's top:	60
Length of Base's bottom:	50
Width of Base's bottom:	80
Number of Underbase:	5
Height of Underbase:	8
Length of Underbase:	100
Width of Underbase:	120

2.



Obj Info

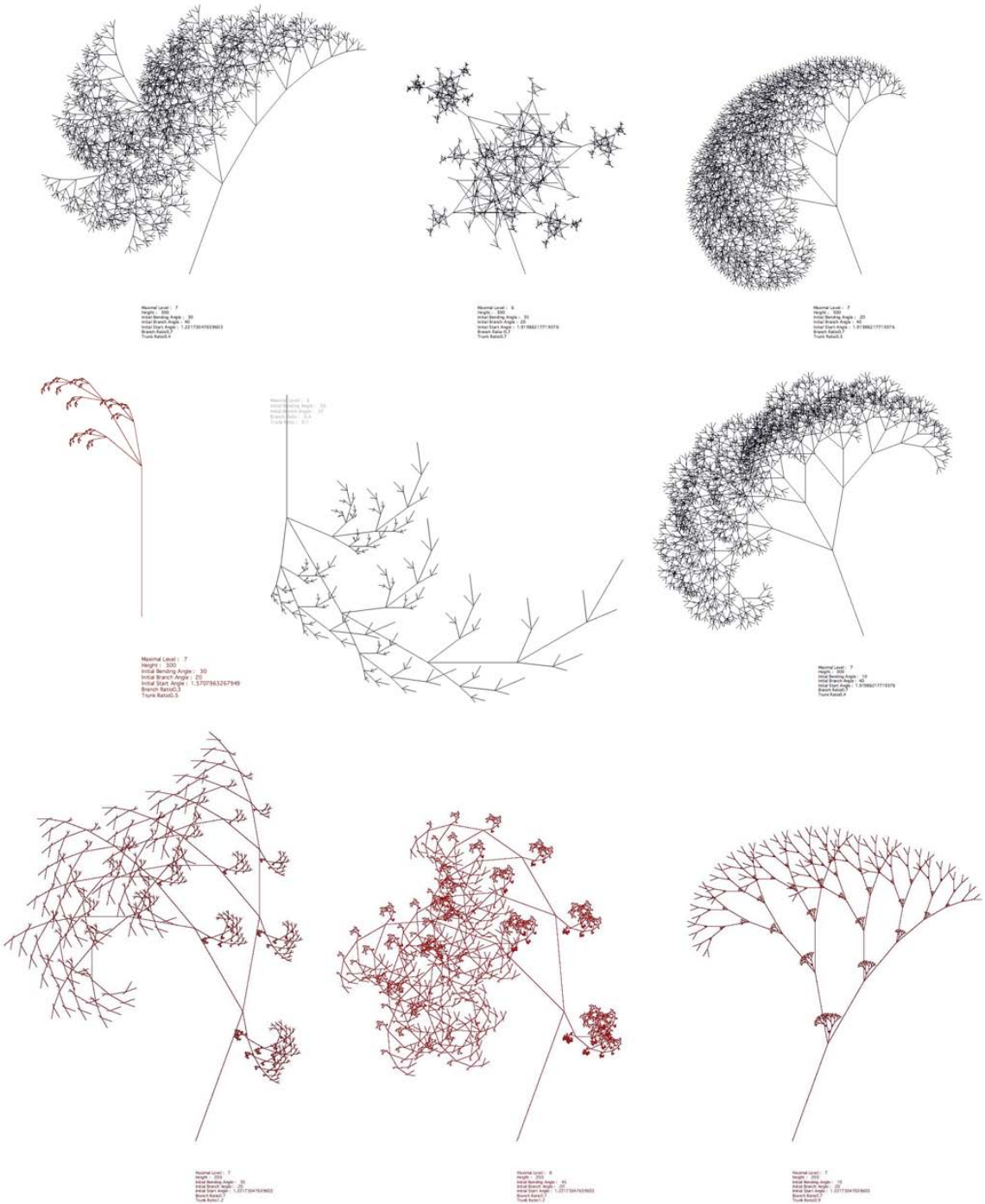
Shape Data Render

parametric Taipei 101

Class:	None
Layer:	Design Lay...
X:	47'6.735"
Y:	51'9.889"
Z:	-6'5.846"
Rot:	0.00°
Height of Antenna:	30
Number of Antenna:	2
Number of HeadAbove:	3
Height of HeadAbove:	5
Number of HeadTop:	3
Height of HeadTop:	5
Number of HeadBody:	2
Height of HeadBody:	10
Number of HeadBottom:	10
Height of HeadBottom:	5
Number of Shaft:	20
Depth of Shaft Corner:	0.7
Height of Shaft:	20
Length of Shaft:	30
Width of Shaft:	50
Number of Base:	3
Height of Base:	6
Depth of Base Corner:	0.5
Length of Base's top:	20
Width of Base's top:	30
Length of Base's bottom:	50
Width of Base's bottom:	80
Number of Underbase:	5
Height of Underbase:	8
Length of Underbase:	100
Width of Underbase:	120

3.

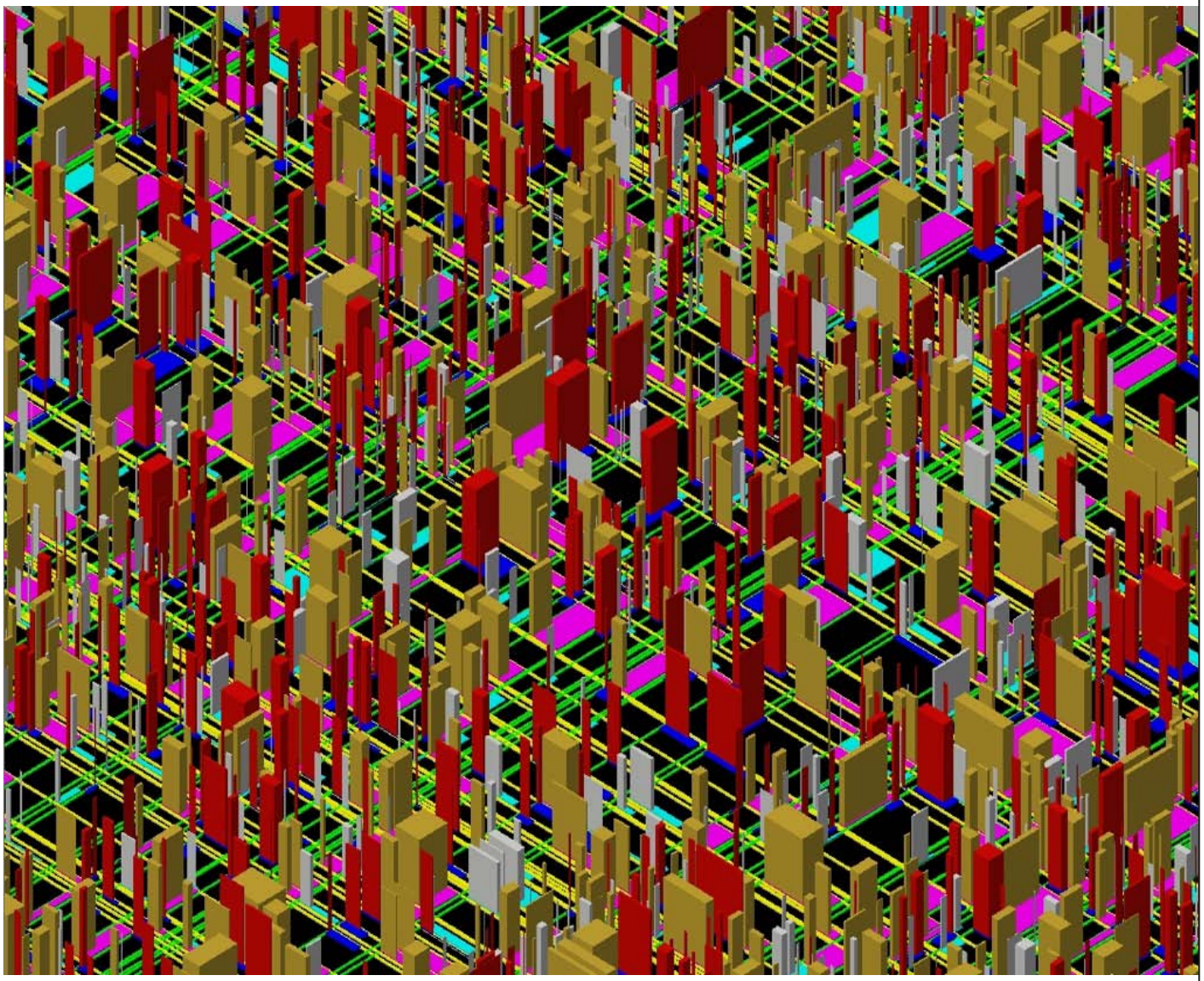
L-SYSTEM CASE STUDY



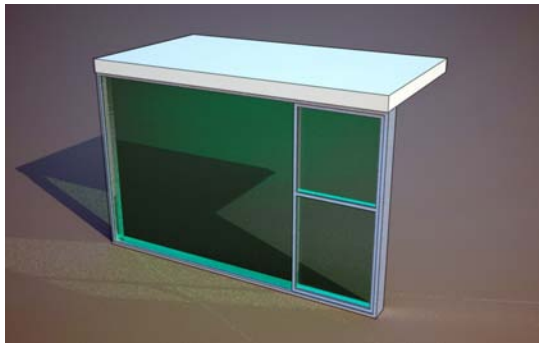
```
PROCEDURE tree32;

VAR

   N,i:INTEGER;
   x0,x,y0,y,Leng,A:REAL;

PROCEDURE subtree(N:INTEGER;x0,y0,Leng,A:REAL);

BEGIN
   IF N<>0 THEN

   BEGIN
      MoveTo(x0,y0);
      FOR i:=1 TO 4 DO
      BEGIN
         Anglevar;
         A:=A+90;
         LineTo(Leng,#A);
         PenLoc(x,y);
         x0:=x;
         y0:=y;
         noAnglevar;
      END;
      subtree(N-1,
            x0+Leng*Cos((A-45)*PI/180)/sqrt(2),
            y0+Leng*Sin((A-45)*PI/180)/sqrt(2),
            Leng/Sqrt(2),
            A-45);
      subtree(N-1,
            x0+Leng*Cos((A+45)*PI/180)/sqrt(2),
            y0+Leng*Sin((A+45)*PI/180)/sqrt(2),
            Leng/Sqrt(2),
            A-45);
   END;

END;

BEGIN
   N:=7;
   Leng:=10;
   A:=10;
   subtree(N,0,0,Leng,A);
END;

RUN(tree32);
```

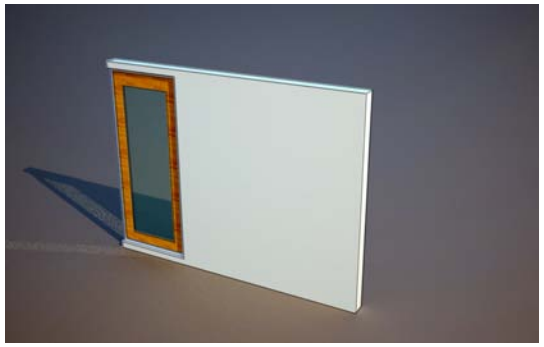
City Generation

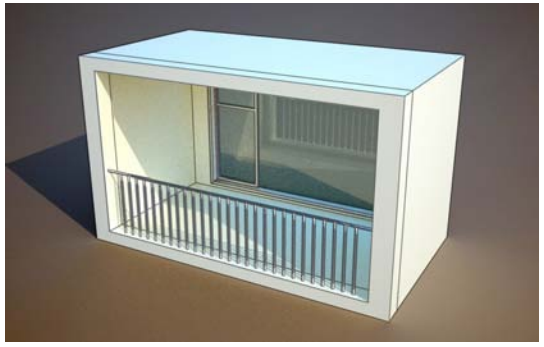
```
PROCEDURE cityplan  ;


VAR
        i, j : INTEGER ;
        roadPos : COORD ;
        sitePos : COORD ;
        buildingPos : COORD ;
        imgSizeV : INTEGER ;
        imgSizeH : INTEGER ;
        thicknessOfRoad : INTEGER ;
        numbOfVRoads : INTEGER ;
        numbOfHRoads : INTEGER ;
        posRoadV : DYNARRAY[ ] OF REAL ;
        posRoadH : DYNARRAY[ ] OF REAL ;
        x_pos : ARRAY[1..2000] OF REAL ;
        y_pos : ARRAY[1..2000] OF REAL ;
        bubbleDummy : INTEGER ;
        r, g, b : LONGINT ;
        tmpRandom1234 : INTEGER ;
        siteXpos1, siteXpos2, siteYpos1, siteYpos2 : REAL ;

        PROCEDURE setImage(setImageSizeH, setImageSizeV : REAL) ;
        VAR
                r, g, b : LONGINT ;

        BEGIN
                PenFore(r, g, b) ;
                FillBack(r, g, b) ;
                BeginXtrd(-5, -10);
                Rect(0, 0, setImageSizeH, setImageSizeV) ;
                EndXtrd;

        END ;

        PROCEDURE setRoad(restricH, restricV : INTEGER ; numbOfVertRoad,
numbOfHoriRoad : INTEGER ) ;
        VAR
                i : INTEGER ;
                dummy_i, dummy_j : INTEGER;
                dummy : REAL ;
                dummy_tX: STRING;
                dummy_tY: STRING;
        BEGIN
                {Vertical Roads}
                FOR i := 1 TO numbOfVertRoad DO BEGIN
                        x_pos[ i ] := (Random * restricH) ;
                END;
                {Horizontal Roads}
                FOR i := 1 TO numbOfHoriRoad DO BEGIN
                        y_pos[ i ] := (Random * restricV) ;
                END;

                FOR dummy_i := 1 TO numbOfVertRoad DO BEGIN
                        FOR dummy_j := 2 TO numbOfVertRoad
DO BEGIN
                                IF (x_pos[dummy_j - 1] >
x_pos[dummy_j]) THEN BEGIN
                                dummy :=
```

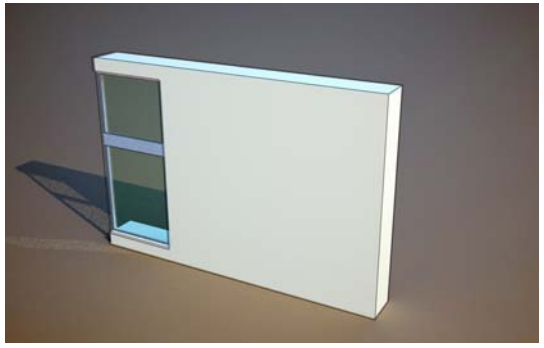


Parts









Parametric Facade

Using a Powercopy function and Catia script files in Digital Project software, a unique building envelope can be generated. After modeling four basic facade units, CATIA Script files that contain dimension values of units can be generated from independent Java software. Final 3D model of a building envelop can be remodeled rapidly.

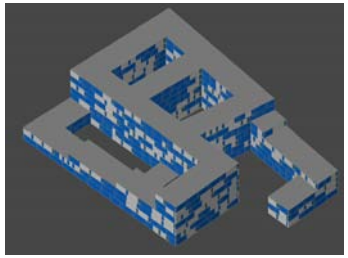
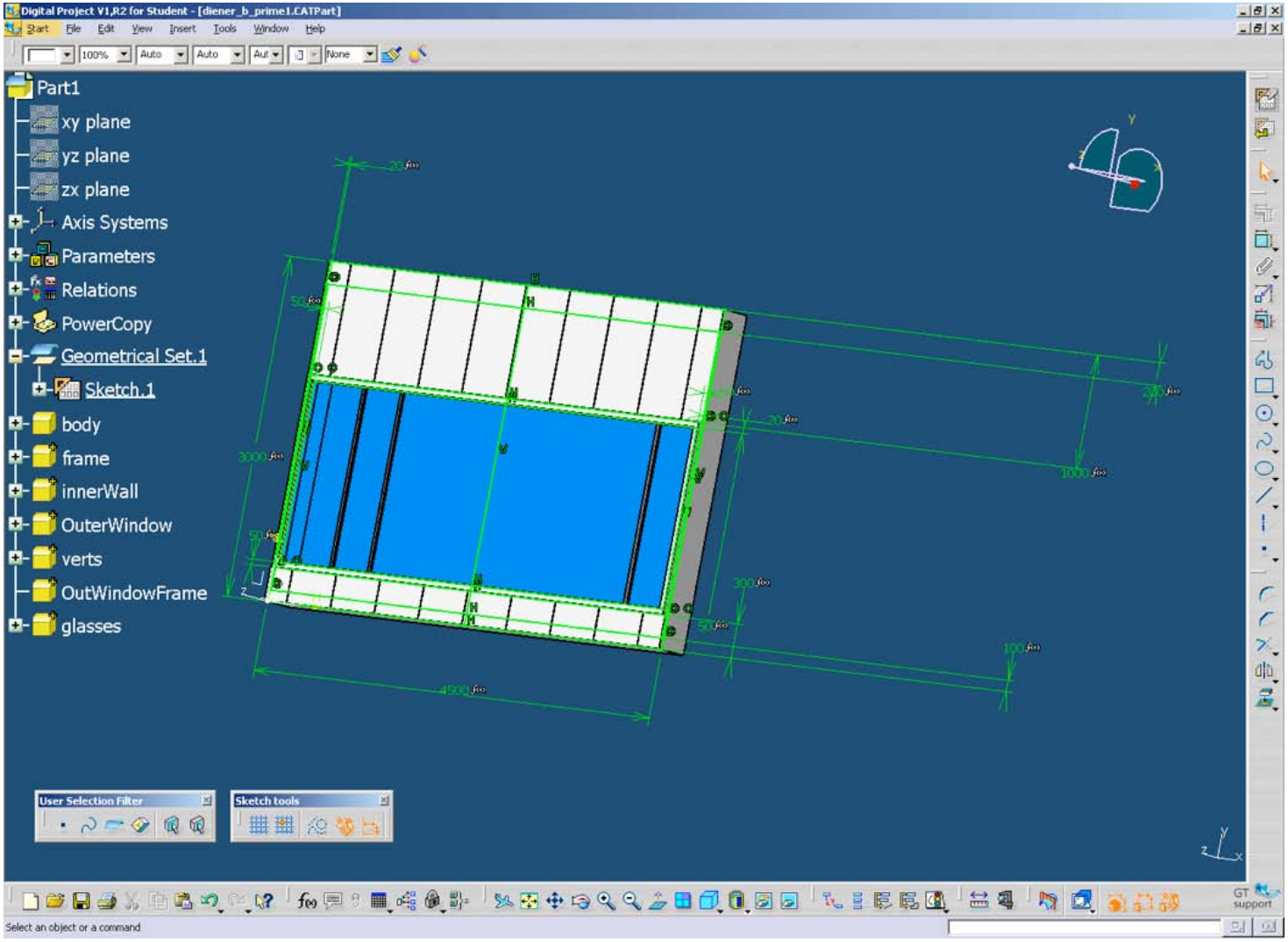
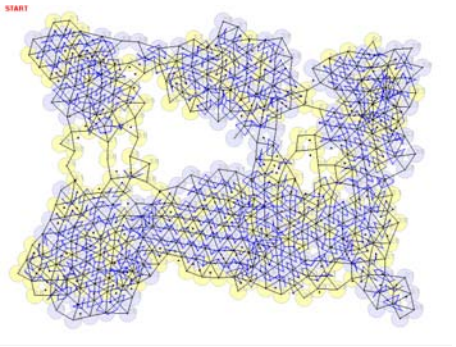


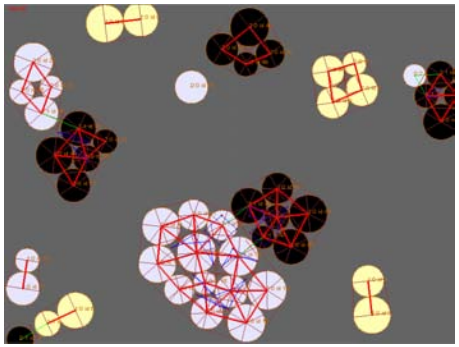
Image Source: Steffen Lemmerzahl

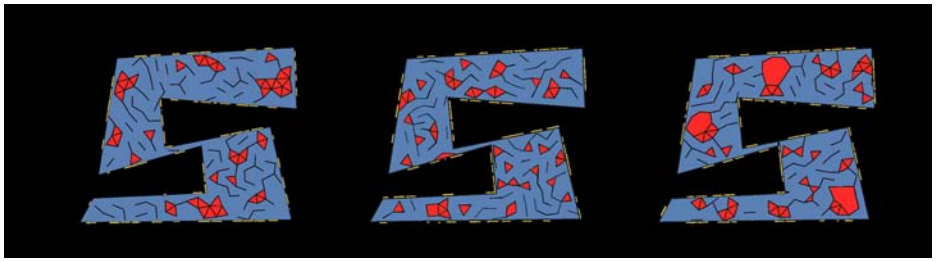


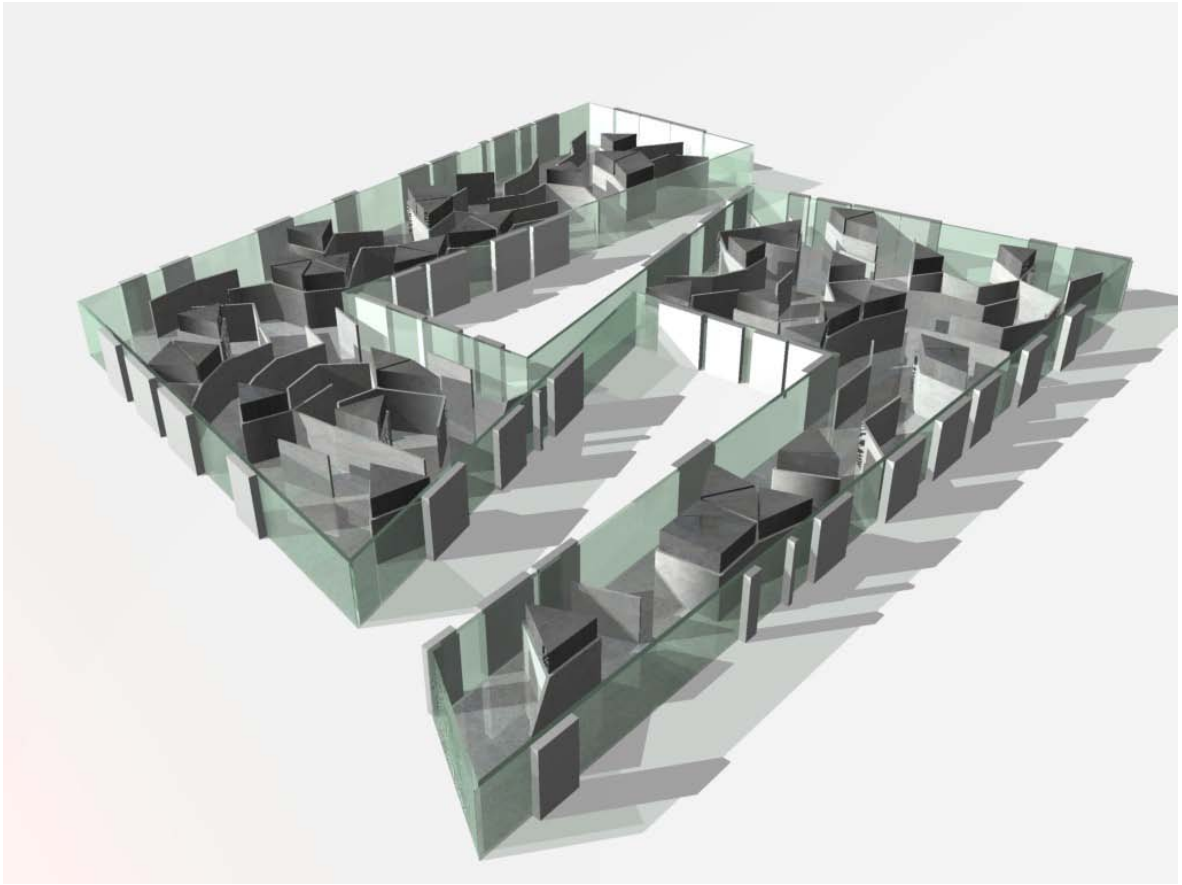
Object Oriented Programming in Processing (2007)

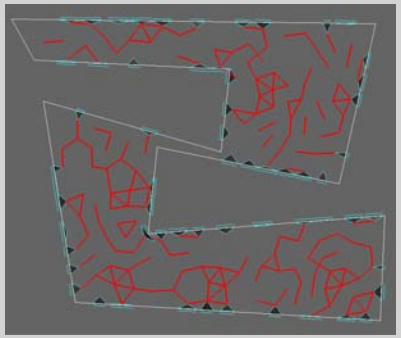
A modified circle packing algorithm and other geometric algorithms are dynamically programmed in Processing in order to understand self-organizing principle. Based on initinal conditions and emergent properties, each circle behaves like a living creature. Data gathered from Processing is saved into XML file and modelled in Vectorworks software.

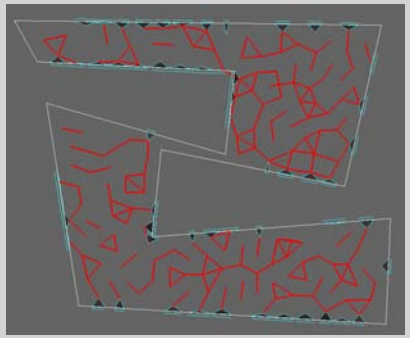


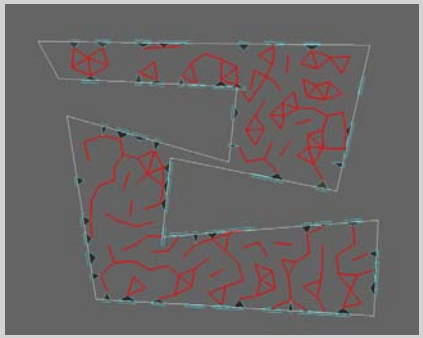


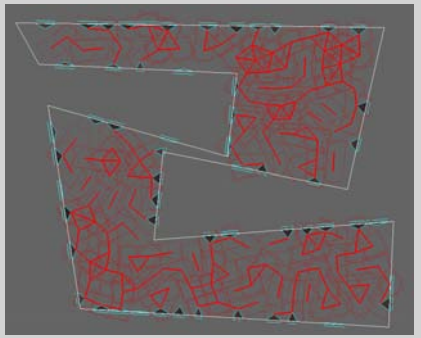


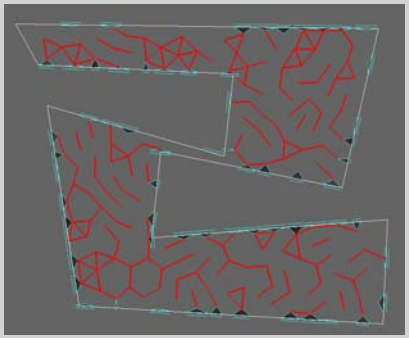


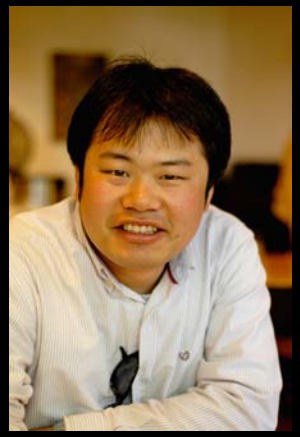












3D Tools - 3dsMax, Maya, FormZ, AutoCAD, ArchiCAD, SketchUP, Revit, CATIA, Solidworks, Rhinoceros, VectorWorks, CityEngine, ArcGIS (ESRI)

2D Tools- Adobe Photoshop, Adobe Illustrator, Macromedia Flash

Video Editing tools - Adobe Premiere Pro, Pinnacle Studio

Language - C#.NET (Good), FORTRAN(Basic), C(Basic), C++(Basic), JAVA(Professional), Processing(Good), Visual Basic(Good)

Script Language - X3D, GDL, MEL, HTML, Vectorworks Script, CATIA Script, Python, Rhino Script

CAM Software - Surfcam, RhinoCAM

Miscellaneous - Solibri Model Checker