Evaluating the Digital Design Process:
Bottom-up vs. Top-down

A Thesis

Submitted to the

Faculty of Miami University

In partial fulfillment of

The requirements for the degree of

Master of Architecture

Department of Architecture

By

ALEX HOGREFE

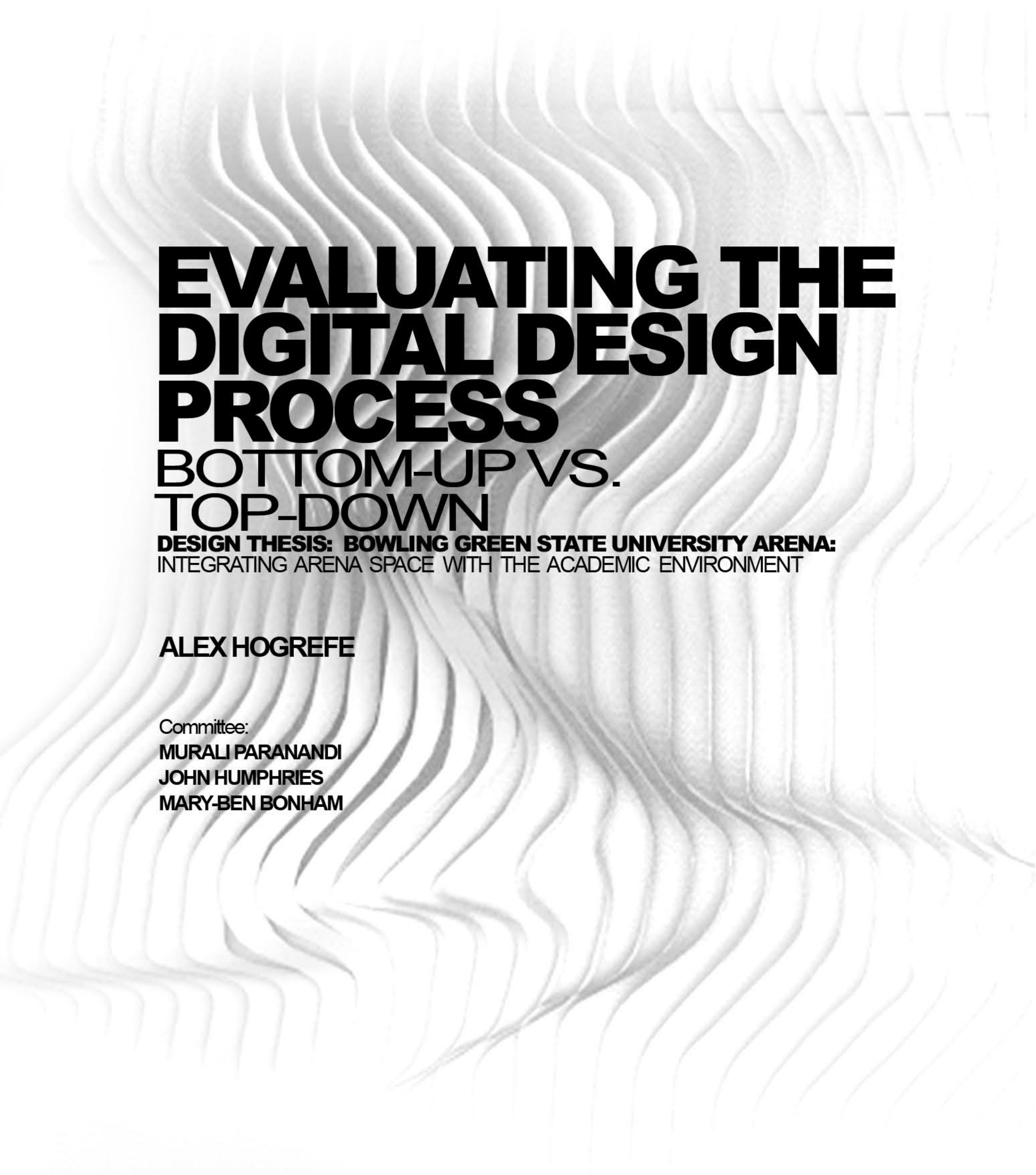
Miami University

Oxford, Ohio

2010

Advisor______________________________
John Humphries

Reader_______________________________
Mary-Ben Bonham



EVALUATING THE DIGITAL DESIGN PROCESS

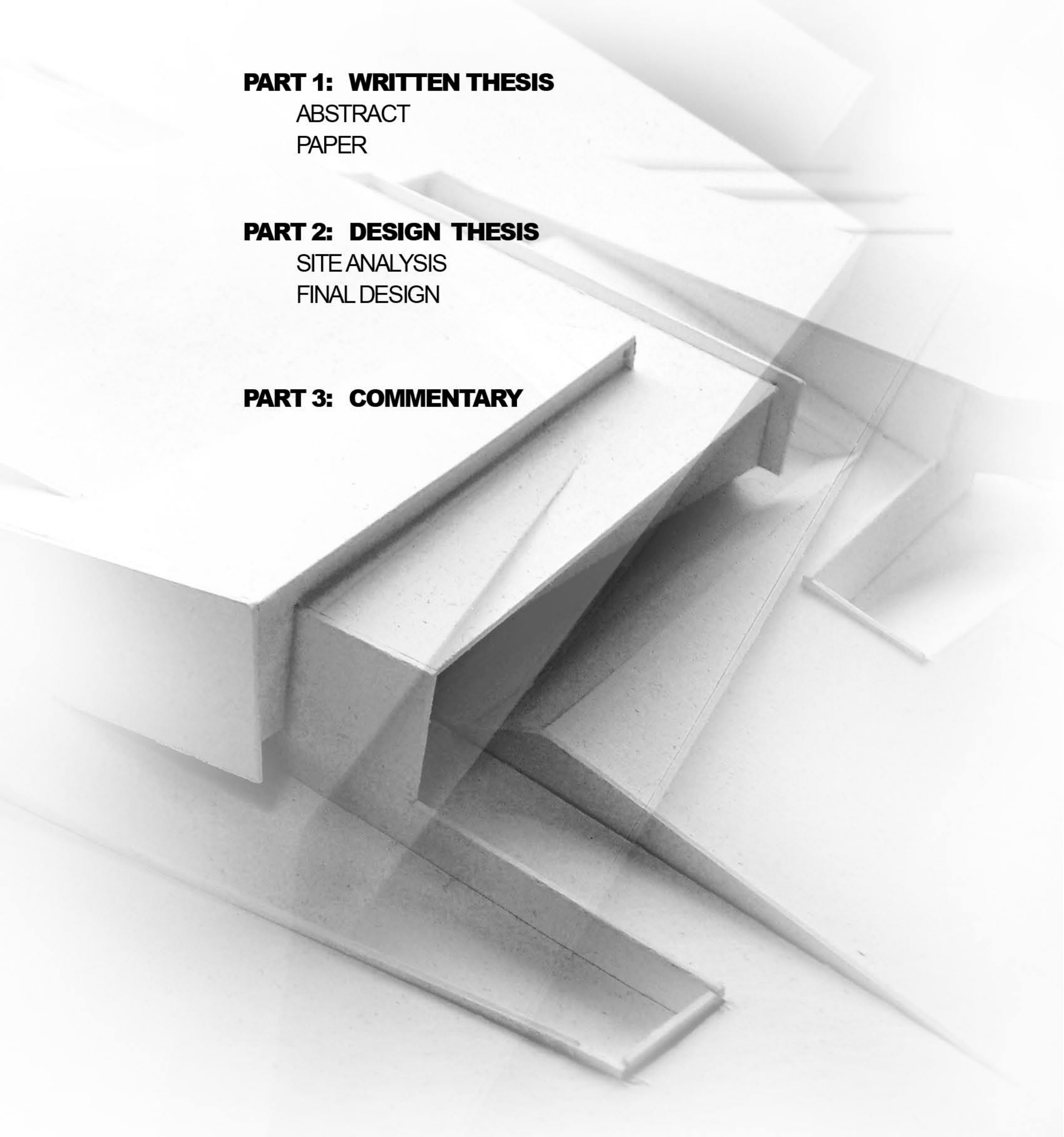
BOTTOM-UP VS. TOP-DOWN

DESIGN THESIS: BOWLING GREEN STATE UNIVERSITY ARENA:
INTEGRATING ARENA SPACE WITH THE ACADEMIC ENVIRONMENT

ALEX HOGREFE

Committee:
MURALI PARANANDI
JOHN HUMPHRIES
MARY-BEN BONHAM

THESIS
TABLE OF CONTENTS



WRITTEN THESIS
ABSTRACT & PAPER

BOWLING GREEN STATE UNIVERSITY ARENA:
INTEGRATING ARENA SPACE WITH THE ACADEMIC ENVIRONMENT

ABSTRACT: EVALUATING THE DIGITAL DESIGN PROCESS: BOTTOM-UP VS. TOP-DOWN

Digital technologies have the ability to reverse, rearrange, and modify common analog design processes. Today's practicing architects still generally maintain a linear design process, limiting flexibility and freedom in intermediate design steps. The introduction of parametric and generative design through scripting alters the design process into a nonlinear path that extends idea manipulation and exploration late into the project development. A look at both the bottom-up approach and the top-down approach reveal a deeper understanding of common digital typologies. Foster and Parnters' City Hall project provides an interesting example of the advantages of parametric models in a top-down process, where as Haresh Lalvani's AlgoRhthms looks at the application of generative design in a bottom-up approach to the project. Both case studies are examples of cleanly executed solutions, made possible by digital design processes. Looking at software commonly used in architecture today, an in-depth study exposes a gap between NURBS based modeling programs such as Rhinoceros and parametrically based programs such as Revit. The introduction of scripting software aids in filling this gap, as well as becoming a powerful means to introducing more powerful generative and parametric capabilities. The scripting software, Grasshopper, was used to conduct empirical research where real problems were solved using digital design methods. The experiments provided a firsthand look at the playful, yet rich exploration that generative and parametric design fosters.

Evaluating the Digital Design Process: Bottom-up vs. Top-down

With the increasing use of digital technology in the architectural design process, how can we better implement computational use to generate articulate, rational, cohesive designs that are responsive to their surroundings? Most architectural practices are content with using proprietary software for strictly representational purposes while others are employing software that does not complement or benefit the design process. The introduction of parametric and generative design methods through scripting alters the design process into a nonlinear path that extends idea manipulation and exploration late into the project development. As a result, it is essential for designers and specifically architects to better understand algorithmic processes and its implementation within scripting software. It is with this knowledge that architects can gain an unprecedented freedom in design, and create customized utilities suited for their specific needs

If architects become more involved with scripting, ideas can be extrapolated where new and innovative digital typologies can be formed. Emergence, morphogenetics, mass customization, and responsiveness, can now be seriously considered and incorporated in design processes. However, certain questions arise from the integration of these digital typologies. Of the methodologies presented, which methods are the most beneficial in terms of design flexibility and exploration. What software is best suited to handle these new methodologies? An exploration of applications currently being used in the architectural field such as Revit, Rhinoceros, and Ecotect will give a better understanding of what software today allows us to do in architecture, and what it could do if combined with scripting software such as Grasshopper. Case studies of projects by Haresh Lalvani and Foster and Partners provide examples of clean integration of computational methods into their design processes.

Opposing arguments against a deeper integration of computational design into architecture quickly arise that have negatively influenced the advance of computer engagement in early design processes. It is argued that it would be more realistic and time saving for architects to collaborate with software engineers instead of learning the skills themselves. Also, as computers are weaved deeper into the design process, architects will lose control and abstractness of their designs compared to traditional analog methods involving freehand sketching. And finally, the computer's involvement with the design process removes authorship from the architect. This paper will examine these arguments more closely, and reveal that it is worthwhile to look deeper into the new methodologies that are only possible with the advent of the computer.

Most users of CAD (computer aided design) software use algorithms built into the application without knowledge of how the algorithm actually works. With proprietary software, the user can only do what the program allows them to do. Much of the software on the market today offer quite a bit in terms of representational modeling. However, parametric and generative capabilities are severely limited in much of this software. An understanding of how code operates within the software gives the user the capability to explore, alter, and manipulate functions to better suit their needs through scripting applications.[1] A more in-depth look at digital design typologies exposes an underlying design methodology consisting of the bottom-up approach, the top down approach, or a combination of the two.

TOP-DOWN APPROACH

Looking first at the top-down approach, this method is described as the breaking down of a system to gain insight. It is typically a linear, hierarchically driven method that is used most commonly among practicing architects. The top-down approach begins with an initial parti or big idea, where it is successively rationalized and refined through progressive steps. As Andrzeh Zarzycki explains, the number of design paths a design can follow is severely limited with this linear approach. This is due to the difficulty to escape the momentum of predictable moves from step to step. [2] To alter initial ideas means altering all sequential steps. The introduction of parametrics into the top-down approach begins to disrupt the linearity of the process. Parametrics extends idea manipulation and exploration late into the project maturity therefore reordering the conventional process into a nonlinear path. This is possible because parametric modeling can establish associative links between large and small scale elements of the model. If an element is modified, all other geometry associated with it is updated as well. Where conventional processes move from design, to rationalization, and finally representation in a very linear fashion, parametric methods allow the designer to move backwards from rationalization to design, or from representation to rationalization. This allows for more fitting alternatives to be tested with little to no effort.

The introduction of parametrics into the top-down process can be observed through a case study of City Hall in London described by Hugh Whitehead in *Architecture in the Digital Age: Design and Manufacturing*. The Building was developed using Microstation. The form was created using a torus patch which is essentially a slice from a donut-like shape. From there, the architects (Foster and partners) created a parametric control system which allowed the team to precisely record dimensions. Whitehead explains that spending a day developing a custom-built parametric model produces a base for testing hundreds of alternatives saving time and energy in the long run. The ability to program in this manner was indispensible. By utilizing the script, the architects were able to play with proportions and still maintain precise control of the project. The digital model was then linked to a CNC machine to test geometries for windows where final solutions were resolved. This dialogue from the digital model to the fabrication machines at a small scale gave the team confidence to build at full scale. The final shape of the building was based off an axis which leaned toward the sun. This allowed for the building to present minimal surface area to the sun while still allowing for maximum views of the city. When summing up the design process of City Hall, Whitehead explained that "These examples illustrate that the synthesis of form is considered from many different viewpoints – functional, spatial, sculptural, structural, and environmental. In trying to combine all these aspects in an optimal solution, we have to build tools that cannot be found in off-the-shelf software." [3]

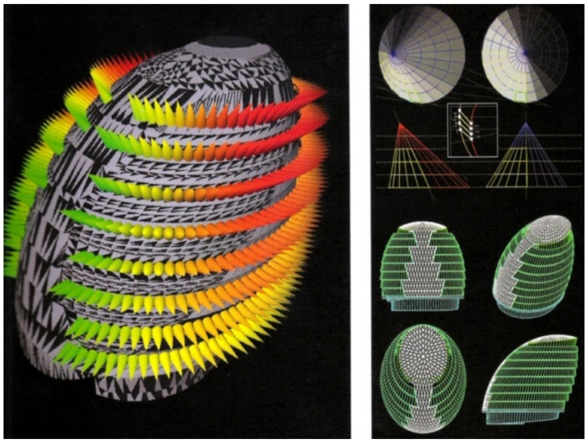


Figure 1. City Hall, London. Parametric models developed to study glazing patterns and heat gain. Source: Architecture in the Digital Age

BOTTOM-UP APPROACH

The bottom-up approach represents the inverse way of thinking compared to the top-down approach. The bottom-up approach is described as the combination and piecing together of smaller components to create a grander, more elaborate system. This method often results in unpredictable, unexpected outcomes, averting most designers from using this method. However, it is unpredictability and chance that the design process needs to avoid scripted, tunnel-vision-like thinking. Andrzeh Zarzycki explains that generative methods allow designers to develop new ideas from past experiences, without replaying them. He states that the bottom-up approach allows designers to think latterly, transcending the inertia of past ideas, and allowing for design leaps.[4] While this generative method can be used to explore forms for aesthetic purposes, the bottom-up approach can provide an opportunity to yield quicker and more precise results when applied to non-aesthetic purposes such as environmental performances. In such cases, generative techniques rearrange the process by which the built form is developed. Instead of developing a form and testing its performance whereby changes will be made to the form, and then tested again, the generative approach looks at what performances need to be achieved, and generates a form around those requirements. The result is an unexpected form, but one that realizes the optimal performance criteria. Ecotect is a building analysis program that has recently become readily available to architects.

The program directly brings into cyberspace, real world conditions such as rain, wind, solar heat gains, and sun paths allowing architects to analyze their designs. Taking data that software such as Ecotect can provide, and applying it to generative models, means that forms can be developed based on performative needs. As the levels of complexity from this type of generative modeling increase, and decision making is based off of other generated information, arguments of authorship quickly surface. However, the designers control is not being erased, it is simply being shifted. The computer is being used as a tool, in which the designer is establishing the rules, by which it follows. The complexity and density of information using generative processes is increased. While the designer is not calculating the intricate mathematical algorithms, they are still setting up the system organization, adjusting variables, and altering a range of starting conditions in an iterative process to achieve the desired outcome.

The bottom-up approach can best be seen through Haresh Lalvani's project, AlgoRhthms. Here, Lalvani uses morphologenesis (defined as the development or evolution of form over time) as a way to explore an efficient and economic way to produce compound curves in collaboration with Milgo/Bufkin, a leading metal fabrication company. Looking at the way most compound curves (a curve where a straight line cannot be found in any direction) are produced today, which is by the use of dies, the process is only economical if the same curve is mass produced thousands of times, as seen in the car industry. But to produce customized compound curves in this matter is far too expensive. Lalvani began looking to computation algorithms for an answer. He looked at inexpensive uses of digital fabrication, such as water-jet cutting, laser cutting, and press braking to produce developable curves (Developable curves are curves produced without deformation, but instead by bending or folding a flat sheet of material). Combing the idea of developable curves with digital fabrication allowed him to bring down the cost substantially, as well as create the opportunity for mass customization. To bring these ideas to life, Lalvani derived an algorithm that defines a group of interrelated, transforming shapes connected to a digital fabrication process. Tied to the first algorithm, a second algorithm formed a library of developable surfaces. By linking the algorithms to the fabrication process, an infinite number of shapes made up of developable curves can be extracted from a family, and immediately fabricated, allowing for mass customization. The bottom-up approach used for this project provided control of intricate operations required to develop the forms, although the final outcome was not necessarily known.[5]

Figure 2. Haresh Lalvani's 'AlgoRhthms'. Source: Architectural Design

WORKING DIRECTLY WITH ALGORITHMIC PROCESSES

Architects seem to have an increasing amount of responsibility and an overwhelming knowledge of many diverse areas of the field. Is it necessary for architects to understand algorithmic processes and scripting? In many cases, it would seem more practical for an architect to collaborate with an expert proficient in computer science to produce algorithmic architecture. This would allow both professionals to do what they do best: the architect to design, and the programmer to write software that achieves the architect's needs. CEB Reas refutes this idea by stating similar cases between artist and computer programmers in the 1960's. Most of the computer generated art at this time was done in collaborations between the scientists and artist. However, it was difficult for the artist to verbalize or describe what it was that they wanted the technologist to do. Reas says that "every artist must decide whether he or she will work collaboratively or directly with software….. working directly with code leads to a deeper understanding of the conceptual potential of the medium."[6] Similarly, it is just as important for the architect to understand programming and write their own code to fully take advantage of the benefits and precisely express what it is they want, instead of translating their ideas to a computer programmer.

SOFTWARE ANALYSIS

With the advantages of different digital design processes, software used today in the architecture field must be examined in order to understand how to benefit from computer software as a design tool. Revit is an example of a relatively new software developed by Autodesk that is meant to be very intuitive using parametric 3-D modeling. Door schedules, elevations, floor plans, ceiling plans, sections, and many other aspects of construction documentation are all linked together so that a change anywhere updates drawing information everywhere else. The capabilities of Revit have led to the integration of the software into a high percentage of architectural firms. The program combines multiple design stages so that the architect can spend less time on construction documentation and more time designing. Although Revit can deliver efficient documentation of a project through parametric modeling, it is also marketed on their website as a tool to "design freely."[7]The problem lies in the fact that the software requires the architect to recognize details about the building before they even start modeling. This removes a great deal of abstractness very early in the design process. For example, to add a wall in a Revit model, one must decide what kind of wall they will use, the thickness of the wall, and materials and components the wall is made up of. Understandably, these components are parametrically based so that the properties can be changed later if need be, but the vocabulary is already planted in the designers head. The applications use of architectural vocabulary such as walls, doors, and windows, encourage the user to avoid use of more creative and inventive components and forms. All default walls are designed to be vertical, 90 degrees from the ground. All default floor slabs are horizontal with no slope. This encourages the architect to design within even tighter limits. A design that has odd angles or irregular forms exponentially increases the difficulty to use the program. Custom walls and floors must be created, as well as custom doors and windows to fit these walls. An advantage that can be extracted from Revit is its in-depth employment of parametrics. Parametrics allow for the change of dimensional information, while still maintaining relationships defined by the user. In the case of Revit, parametrics are used to continuously update drawing information in real-time whenever a change is made to the model. As a result, constant updating and manipulation of form can be achieved right up to the construction phase. The program has redefined efficiency in architecture, by combining all components of a building (structure, mechanical, electrical, plumbing systems) into one, complete model. However, the architectural vocabulary and decision making forced upon the designer render the program a representational software, not a design software, making its use only appropriate in late project phases.

3-dimension modeling Applications such as Rhinoceros and FormZ allow the user to create controlled surfaces quickly and easily. These applications offer accurate modeling of a preconceived form, but to adjust a form, the model must be rebuilt or reworked. This may not be a daunting task for basic forms or of a small quantity, but to manipulate complex forms of a large quantity would be an overwhelming, complicated, and impractical task. This inability to adjust complicated or mass quantities of forms pushes designers to be more hesitant to continue exploration and testing of new ideas. Designers inexperienced with these programs find it difficult to rationalize or generate tectonics from the forms they create with these applications. It is easy to get lost in the form, without being able to trace where the geometry came from, or how to accurately reproduce it. An important missing part of these applications is the use of parametrics. Neither Rhinoceros nor FormZ maintain a high level of parametric features.

The polar opposite characteristics of representational and 3-dimension modeling programs reveal a large gap in popular architecture software. An understanding of scripting can aid in linking the advantages of parametric modeling found in Revit with the easy to use NURBS based modeling applications. Here, scripting becomes the means by which digital design methods enter into and influence the design process. To better illustrate this, Selected examples of empirical research will be given to explain both top-down and bottom-up design methods using the scripting software, Grasshopper. Simply stated, Grasshopper is graphical algorithm editor. The program operates within the program Rhinoceros, bringing to Rhinoceros parameter control, programming functions, generative and randomness capabilities. While most scripting programs require architects to switch mindsets leaving visual modeling for a harsher coding interface, Grasshopper maintains a graphical approach lessening the difficult transition as well as the learning curve.

EMPIRICAL RESEARCH

Experiment 1 is an example of the top-down approach taking advantage of parametric capabilities. This grasshopper definition takes a large number of objects, in this case louvers, and orients them towards a single point. Parameters are set up to control the amount of louvers along a given distance, their size, and spacing. The louver locations are also controlled by a curve. Changes to the curve automatically update the placement of the louvers, not affecting the parametric properties. The attractor point can be moved anywhere in space, adjusting the louvers accordingly. Possible applications for this script could be applied to controlling sun light entering the space. Using environmental performance software such as Ecotect, sun path data could be entered into the grasshopper definition, and optimal orientation of the louvers could be achieved. With this parametric set up, the partnership with the computer can really be appreciated. Working with a high number of objects such as louvers would make adjustments without parameters not only time consuming, but also difficult when calculating angles of orientation. Parameters in this case allows for a more playful exploration.

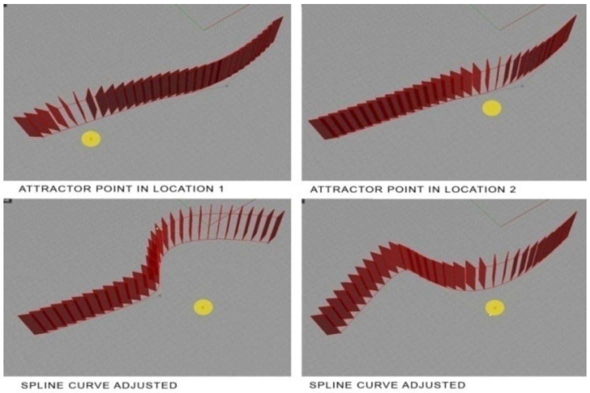


Figure 3: Screen shots of Experiment 2 Grasshopper definition. Source: Author

Experiment 2 looks at the generative, bottom-up approach to designing by establishing rules that guide the design to a final, unexpected form. This experiment began with hand carved boxes that I have been designing and building for the past two years. A rule set up initially was that no box design could be duplicated. As the boxes evolved, they took on characteristics of previous boxes, but maintained their originality through transformations. I was interested to see if generative modeling techniques could be used to foster new formal ideas for the box designs. I began studying the boxes already physically produced to determine the behavior and manipulations performed on them. The way the boxes are constructed, which is by subtractive means through sanding, limits the types of behaviors and manipulations that can be used. In fact, there were only three behaviors used: move, rotate, and scale. However, it was the combination of these behaviors that created the many different forms. From this understanding, I was able to set up parametric controls replicating these behaviors in the Grasshopper model.

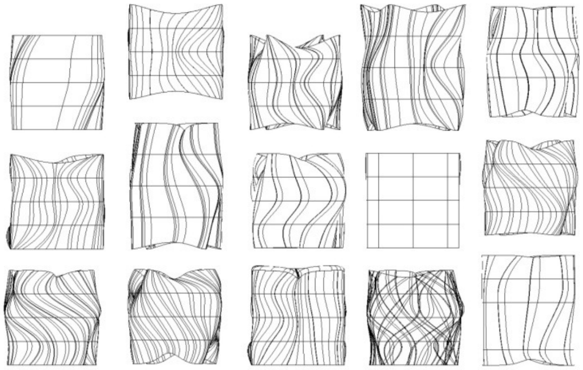


Figure 4: Sample of box forms generated by the Grasshopper definition. Source: Author

Once the parameters were in place, I began manipulating the digital model in search of new forms. Initially, I adjusted the parameters manually, but quickly realized that I was restricting the possibilities of the script by trying to find patterns and similarities with the already fabricated boxes. To truly utilize the power of Grasshopper, I applied random number generators to the parametric controls to produce forms completely unexpected, but still preserving the same language of behaviors setup with the original boxes. The result was a seemingly infinite amount of configurations that stemmed from the complex, to basic formal gestures. An exponential increase in new ideas were spawned that would not have been realized from an analog process. The idea of this Grasshopper definition was not to generate final box designs that could be milled precisely by a CNC router, but instead, to generate new ideas that fit within an already established family of handmade boxes.

Experiment 3 looks at merging ideas from all previous experiments into the development of a high-rise design. I began building a Grasshopper model that could do two things: utilize the bottom-up approach to generate random sloping surfaces and make use of top-down processes that would allow for flexibility in design development. The final definition solved both issues simultaneously. The Grasshopper definition generated sloping vertical landscapes by inserting a range of heights that could be altered to provide steep slopes, or very little sloping. The script randomly created variable slopes within the inputted range of heights. This gave the vertical landscape elements a very natural, wandering look and avoided me as a designer subconsciously creating patterns or repetition, which would ultimately ruin the affect. Formally, the tower was to take on characteristics of two elements of the surrounding context; a bridge pier and a circular highway ramp. The two forms were to be combined, but it wasn't understood how this would be accomplished. Parameters were set up in the Grasshopper definition to allow a wide range of manipulation of the complex forms as shown in figure 5. Floor heights, number of floors, floor slab thickness, location of towers, and footprint geometry were also parameterized which was crucial to the development of the towers as we began looking at program, as square footage. The combination of ideas from previous experiments into this grasshopper definition ultimately permitted a more expansive look at possible solutions for the design, formally and programmatically, using the computers advantages of organizing data and complex geometrical relationships.

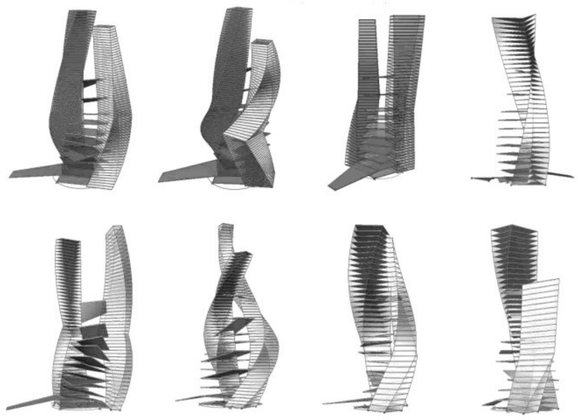


Figure 5: Skyscraper formal studies generated from Grasshopper. Source: Author

Through this empirical research, it was determined that both generative (bottom-up) and parametric (top-down) processes each have an important role in architectural design, and exemplify the importance for scripting, essentially altering the fluidity and relationship of traditional design development. A key reason why I, as a designer, averted digital technology's integration into my design processes early on in my education was because of its tendency to force detail too quickly, as well as hinder a sense of freedom and playfulness due to my lack of understanding of the software. Hand sketching seemed to offer an unmatched ability to connect mental ideas to physical visualization. However, the experiments above provide insight into how digital design processes can become abstract and playful as well as be advantageous to traditional analog design. It is not being said that sketching should be removed completely from the design process. However, it should not serve as the only means of design. By incorporating generative and parametric techniques, one can escape the nonflexible linear design process, and enter into a much richer exploration of design possibilities.

---

[1] Kostas Terzidis, *Algorithmic Architecture* (Burlington, MA: Elsevier Ltd 2006): 41.

[2] Andrzje Zarzycki, "Giving Our Ideas a Playground, Not a Contained Shoebox," *FormZ Joint Study Journal* (2007): 65

[3] Hugh Whitehead, *Architecture in the Digital Age: Design and Manufacturing*, Ed. Branko Kolarevic (New York: Spon Press):85.

[4] Andrzje Zarzycki, "Giving Our Ideas a Playground, Not a Contained Shoebox," *63*

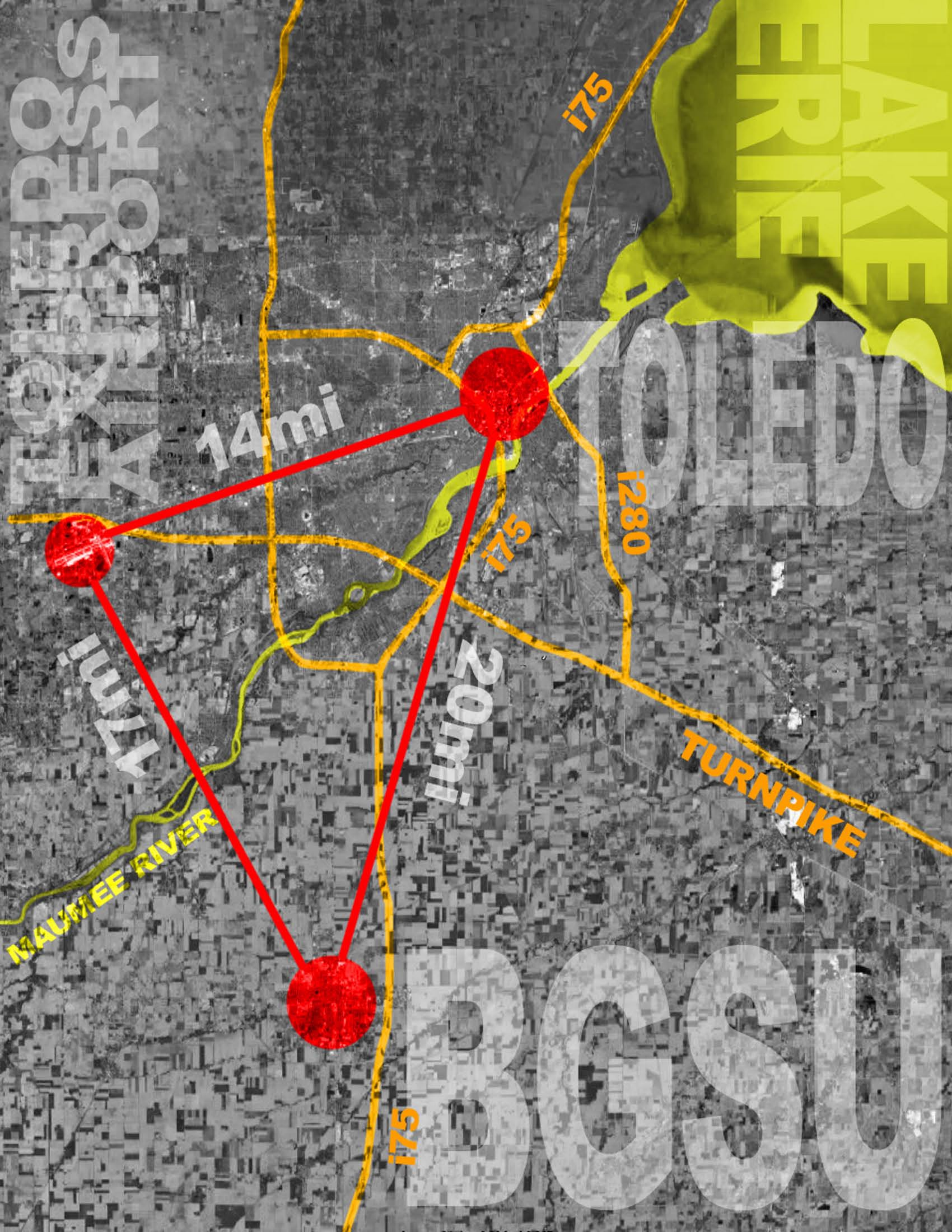
[5] Haresh Lalvani, "The Milgo Experiment: An Interview with Haresh Lalvani." *Architectural Design* (2006): 52.

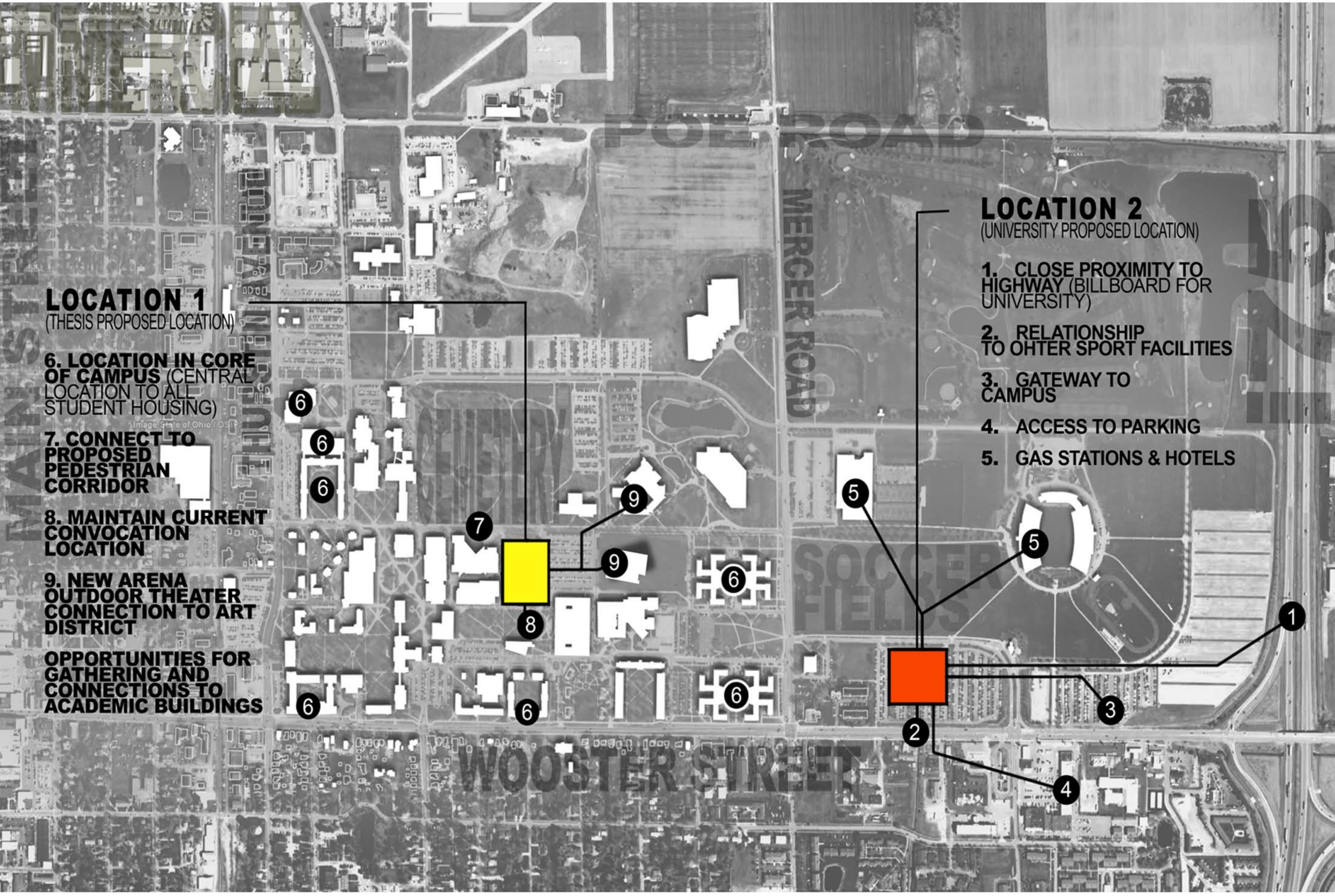
[6] CEB Reas: "Process/ Drawing: Programming Cultures," *Architectural Design (*2006): 33.
CEB Reas also collaborated with Ben Fry in the development of the programming language, Processing. This language has been very influential in getting non-experienced programmers to enter algorithmic thinking. The Processing language syncs well with many other applications, and can even be integrated into Grasshopper.

[7] Revit Architecture Product Trials.
http://resources.autodesk.com/architecture/Revit
(accessed March 3, 2009)

DESIGN THESIS
SITE ANALYSIS

BOWLING GREEN STATE UNIVERSITY ARENA:
INTEGRATING ARENA SPACE WITH THE ACADEMIC ENVIRONMENT


TOLEDO EXPRESS AIRPORT
LAKE ERIE
TOLEDO
i75
i280
i75
14mi
17mi
20mi
TURNPIKE
MAUMEE RIVER
BGSU
i75



LOCATION 1
(THESIS PROPOSED LOCATION)

6. LOCATION IN CORE OF CAMPUS (CENTRAL LOCATION TO ALL STUDENT HOUSING)

7. CONNECT TO PROPOSED PEDESTRIAN CORRIDOR

8. MAINTAIN CURRENT CONVOCATION LOCATION

9. NEW ARENA OUTDOOR THEATER CONNECTION TO ART DISTRICT

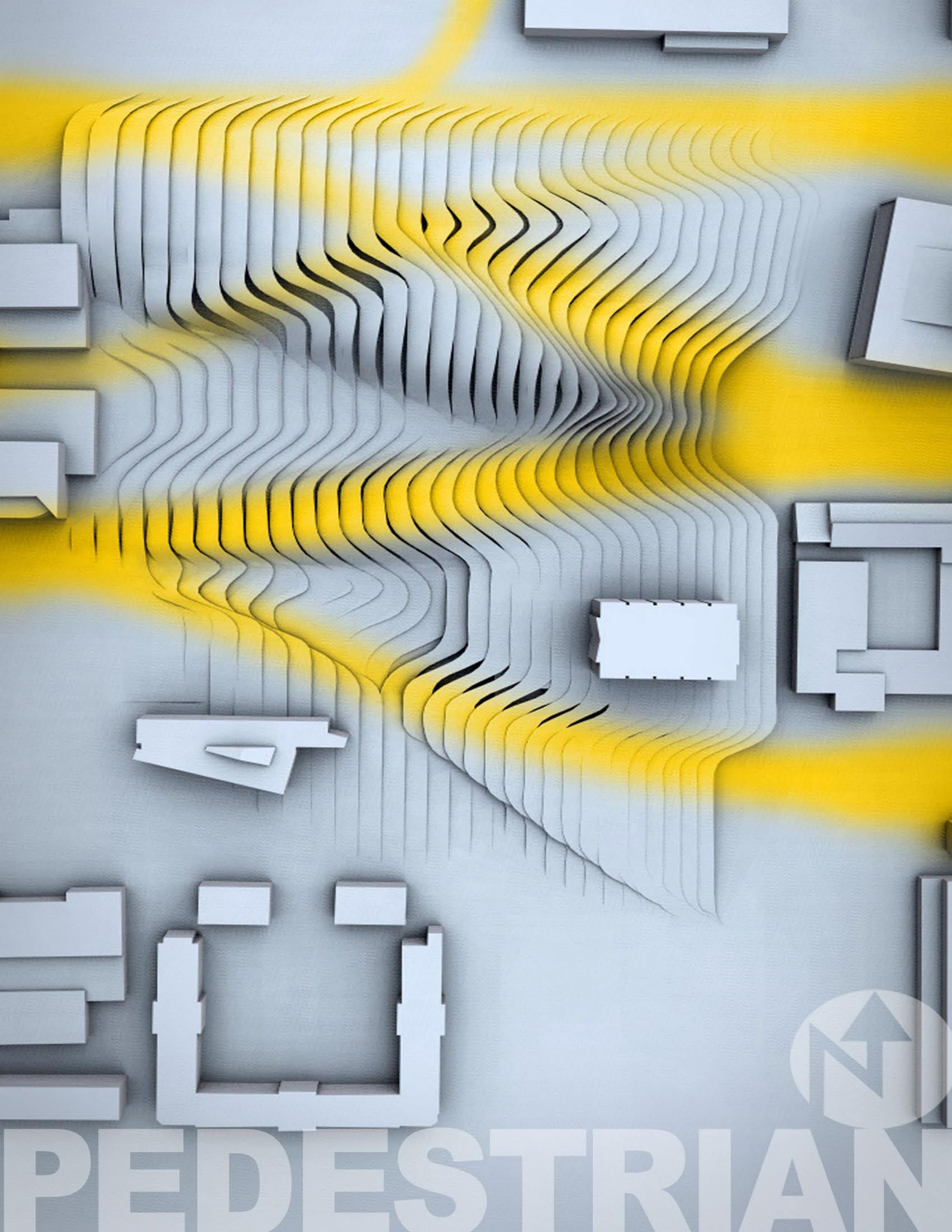
OPPORTUNITIES FOR GATHERING AND CONNECTIONS TO ACADEMIC BUILDINGS

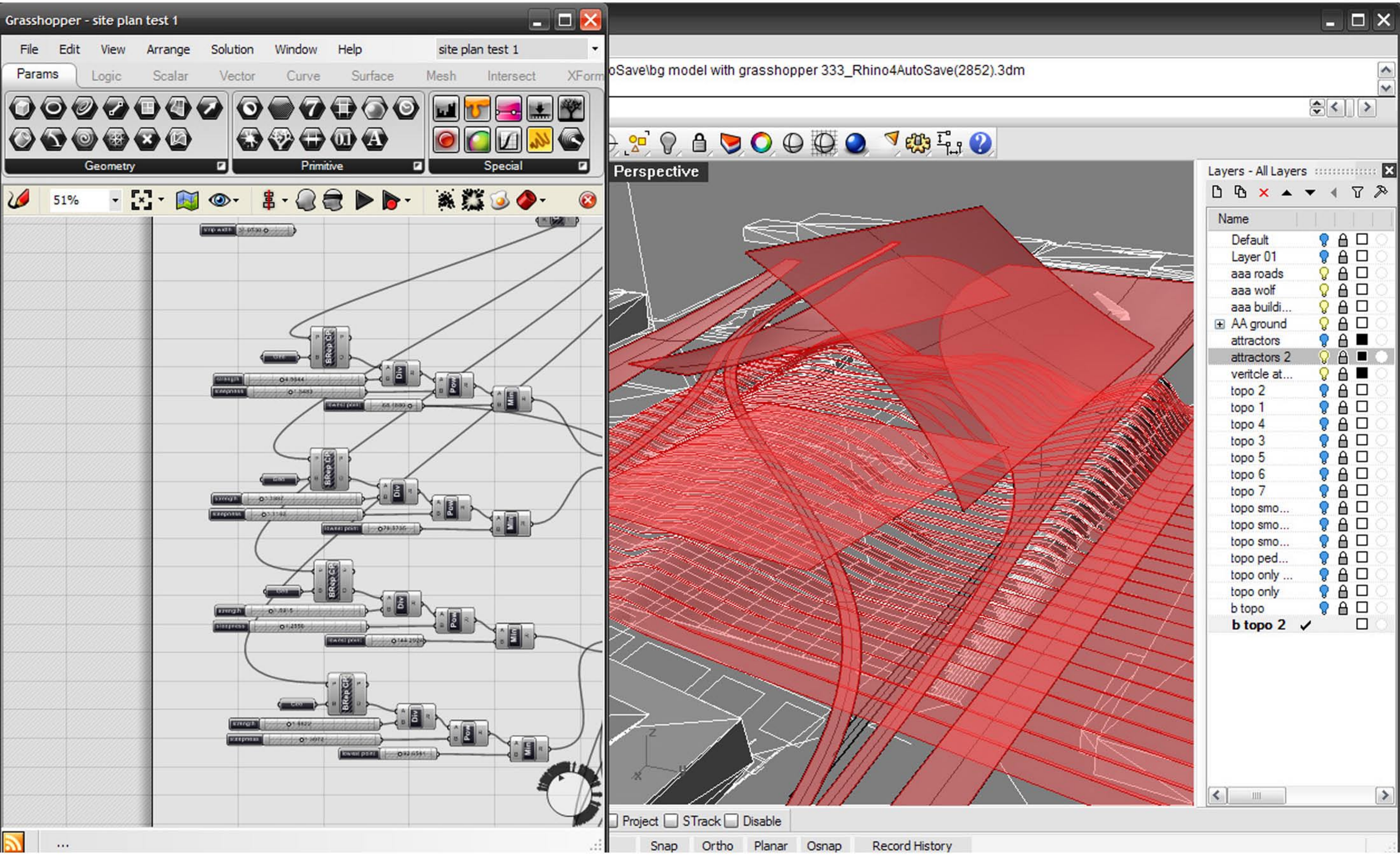
LOCATION 2
(UNIVERSITY PROPOSED LOCATION)

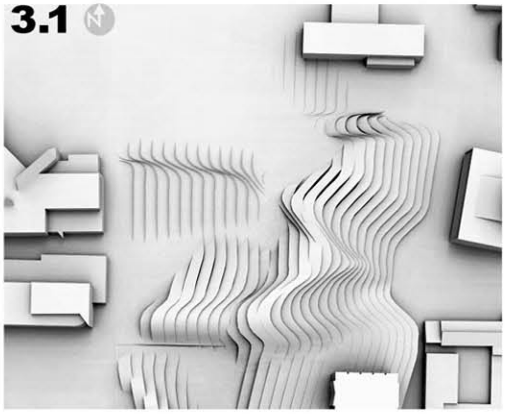
1. **CLOSE PROXIMITY TO HIGHWAY** (BILLBOARD FOR UNIVERSITY)
2. **RELATIONSHIP TO OHTER SPORT FACILITIES**
3. **GATEWAY TO CAMPUS**
4. **ACCESS TO PARKING**
5. **GAS STATIONS & HOTELS**

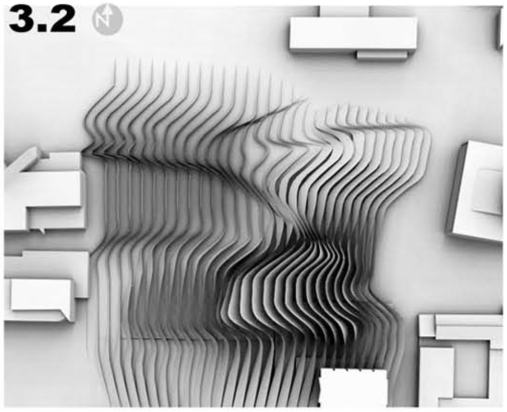

VIEW
AMPHITHEATER
SOUND BARRIER
EDGE
LIBRARY
ENTRY
STREET
CONNECTION
N

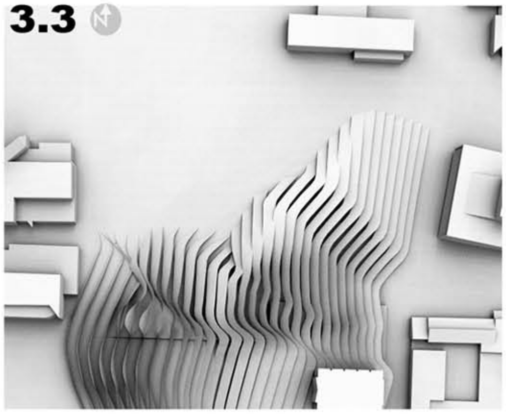
SITE FORCES

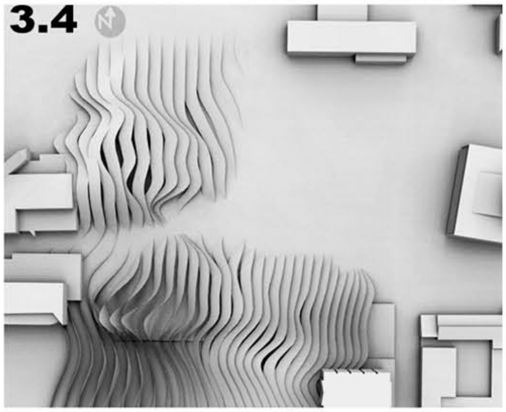

PEDESTRIAN
N

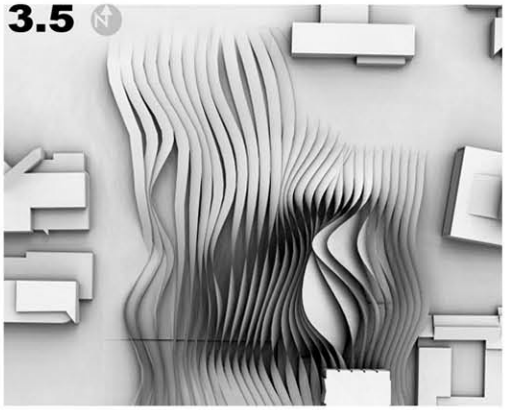

Grasshopper - site plan test 1
File
Edit
View
Arrange
Solution
Window
Help
site plan test 1
Params
Logic
Scalar
Vector
Curve
Surface
Mesh
Intersect
XForm
Geometry
Primitive
Special
51%
BRep CP
Div
Pow
Min
oSave\bg model with grasshopper 333_Rhino4AutoSave(2852).3dm
Perspective
Layers - All Layers
Name
Default
Layer 01
aaa roads
aaa wolf
aaa buildi...
AA ground
attractors
attractors 2
veritcle at...
topo 2
topo 1
topo 4
topo 3
topo 5
topo 6
topo 7
topo smo...
topo smo...
topo smo...
topo ped...
topo only ...
topo only
b topo
b topo 2
Project
STrack
Disable
Snap
Ortho
Planar
Osnap
Record History

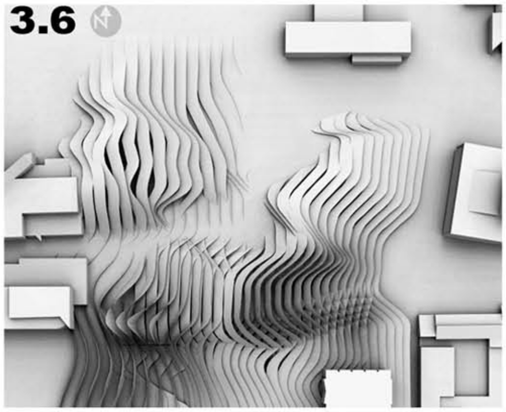

3.1


3.2


3.3


3.4


3.5


3.6

3.1- Low profile massing. Program located mostly underground

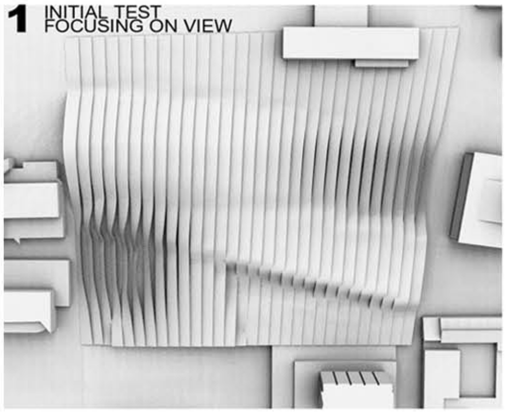
3.2- Massing located on west side of site. Less of a connection to Wolfe Center and library. Provides strong edge on south side of site closing the courtyard.

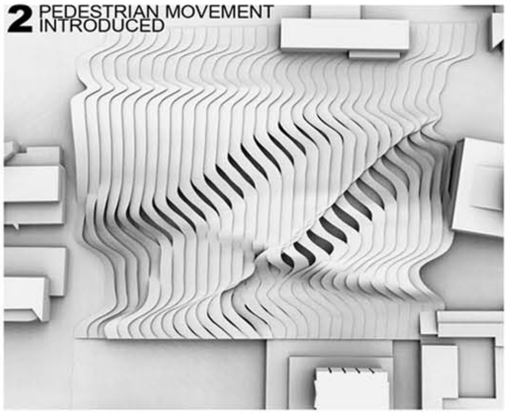
3.2- An attempt to maintain clear pedestrian paths by bisecting massing. Massing responds separately to Wolfe Center and 96ft. library.

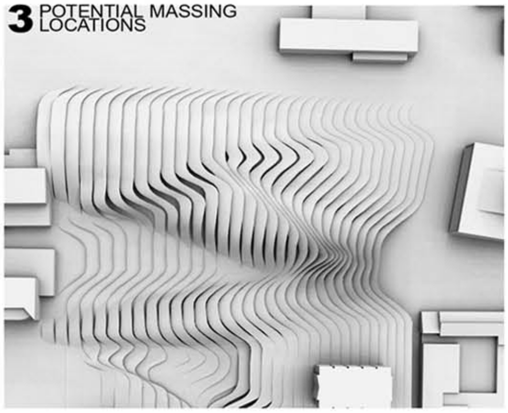
3.5- Deformation of system from overstressed site forces.

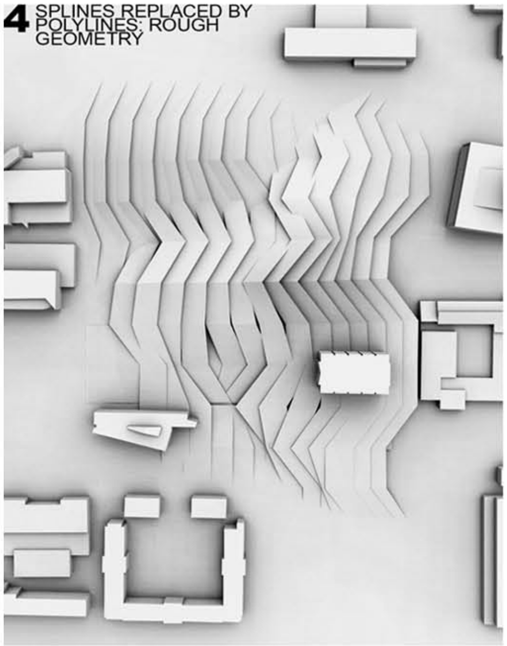
3.3- Low impact on the site. Forces push most of the massing near 96ft. tall library. Issues arise about providing enough massing for program.

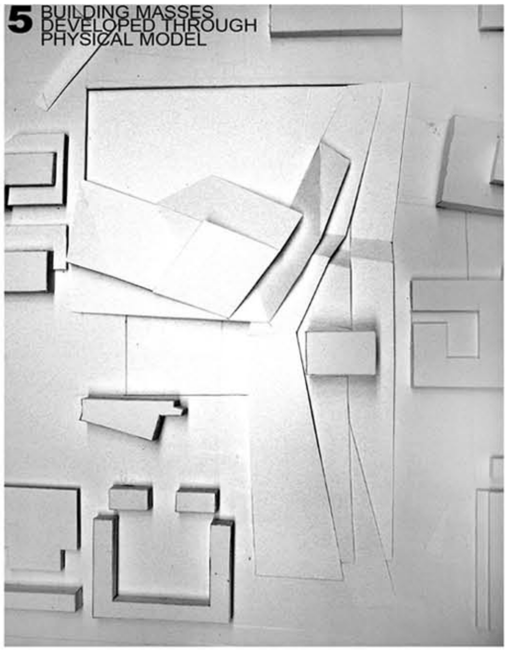
3.6- Layering of two systems. Ideas of spacial layering are pronounced, clarity of site forces are less determined.

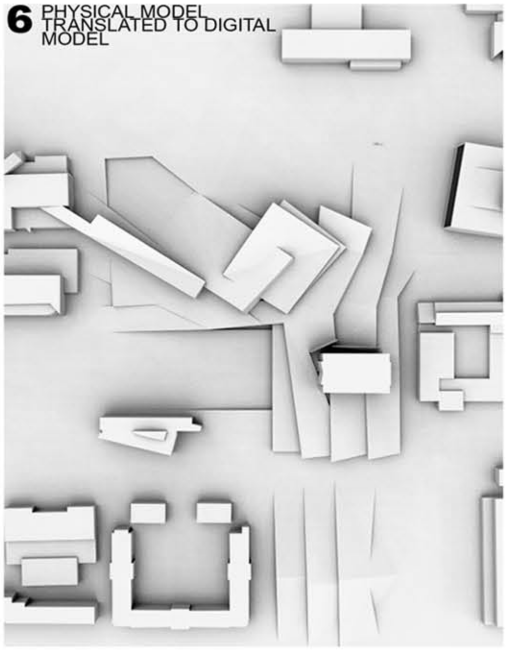

1 INITIAL TEST FOCUSING ON VIEW


2 PEDESTRIAN MOVEMENT INTRODUCED


3 POTENTIAL MASSING LOCATIONS

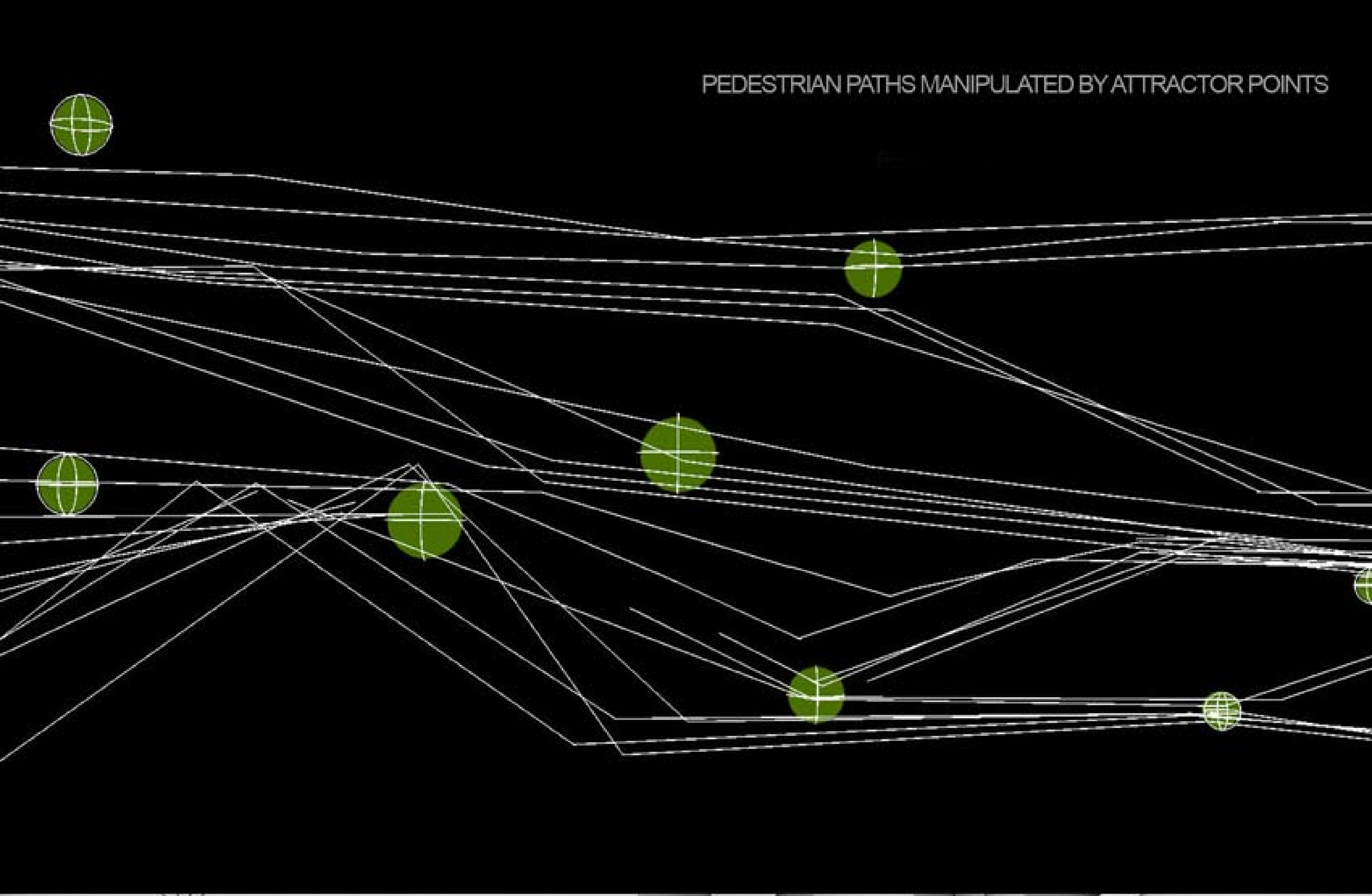

4 SPLINES REPLACED BY POLYLINES: ROUGH GEOMETRY


5 BUILDING MASSES DEVELOPED THROUGH PHYSICAL MODEL

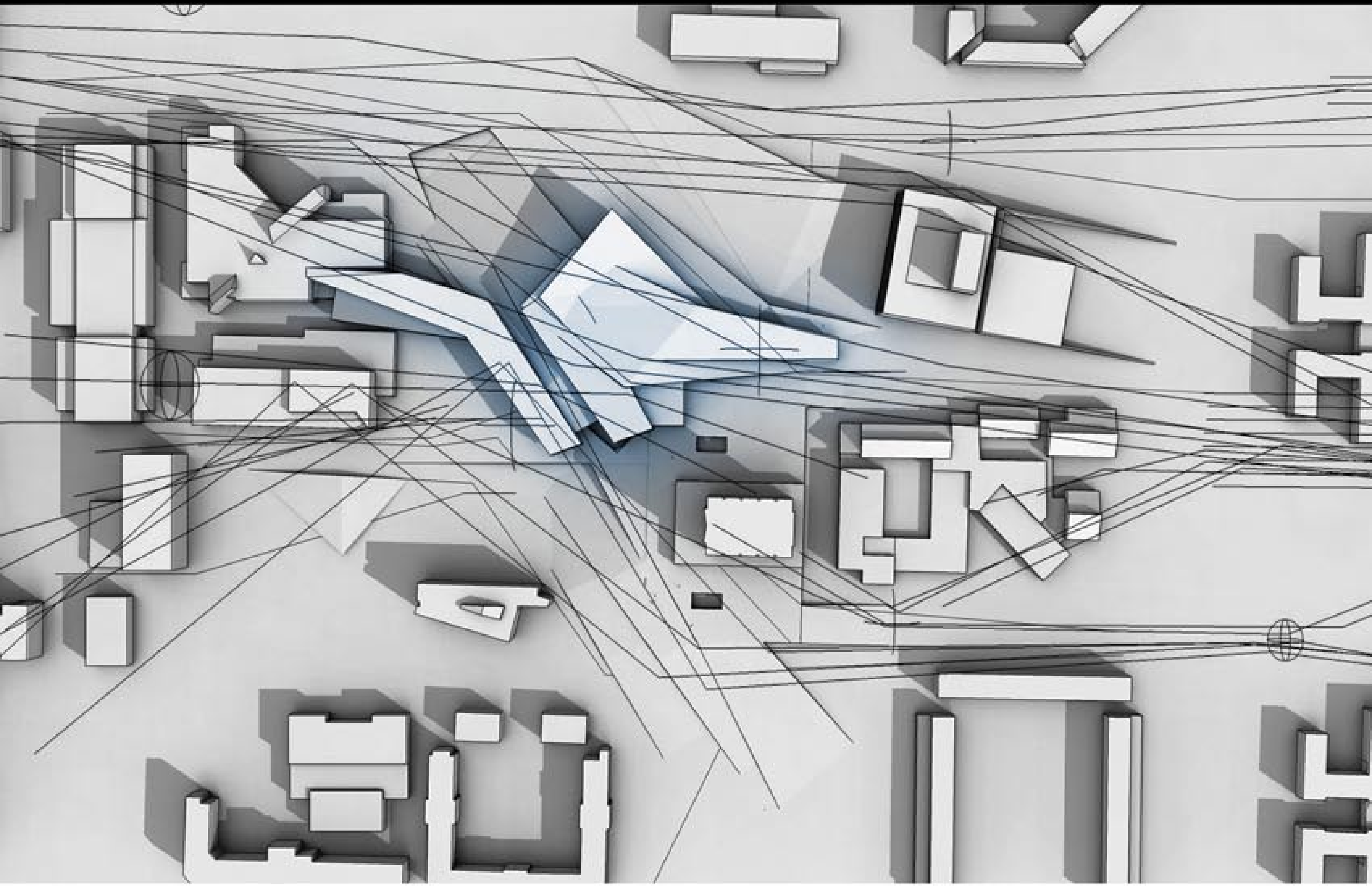

6 PHYSICAL MODEL TRANSLATED TO DIGITAL MODEL


EXISTING PEDESTRIAN PATHS
MAJOR INTERSECTIONS


NEW PEDESTRIAN PATHS
MAJOR INTERSECTIONS

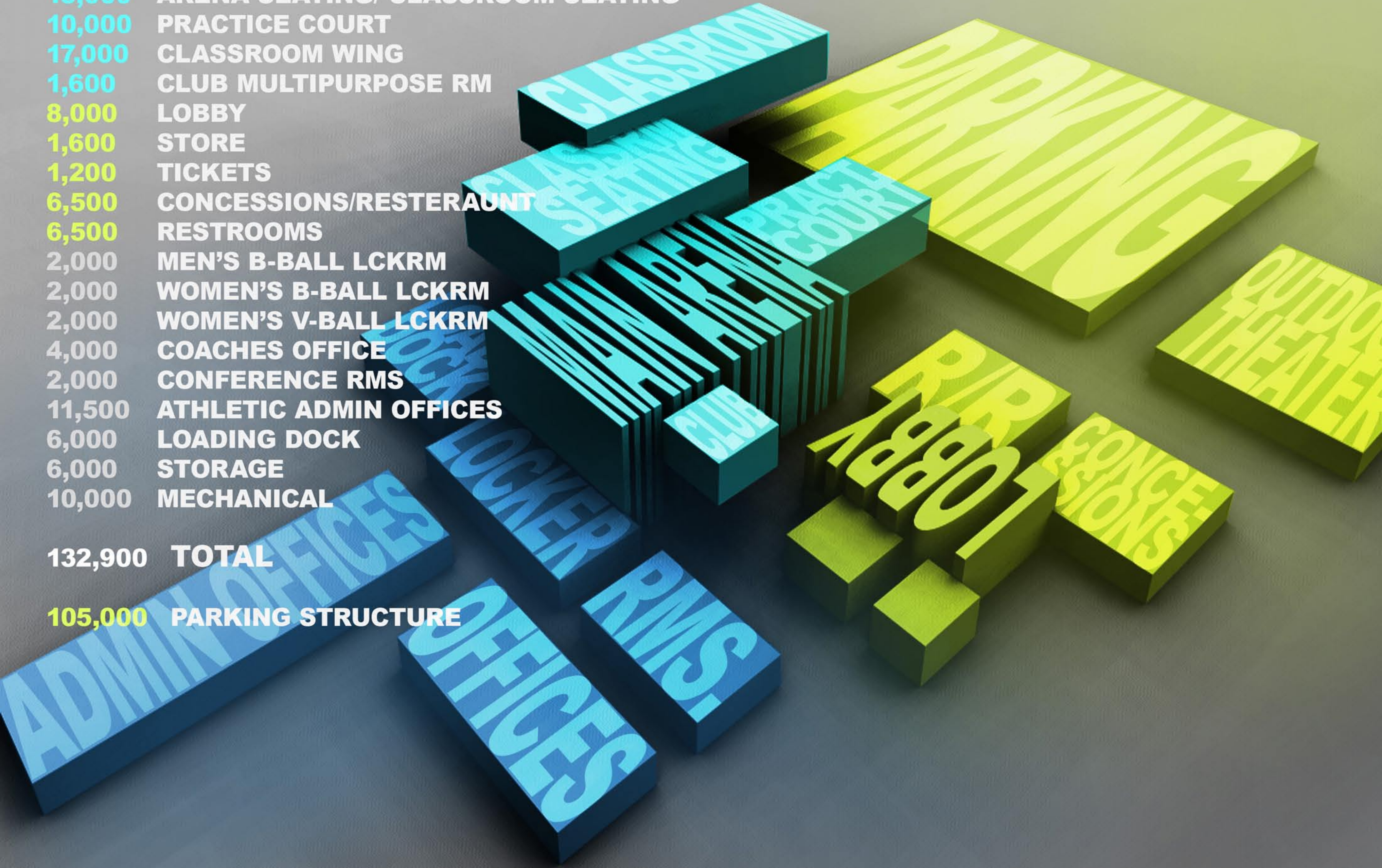


PEDESTRIAN PATHS MANIPULATED BY ATTRACTOR POINTS



PROGRAM

20,000	MAIN ARENA AND RETRACTABLE SEATING
15,000	ARENA SEATING/ CLASSROOM SEATING
10,000	PRACTICE COURT
17,000	CLASSROOM WING
1,600	CLUB MULTIPURPOSE RM
8,000	LOBBY
1,600	STORE
1,200	TICKETS
6,500	CONCESSIONS/RESTERAUNT
6,500	RESTROOMS
2,000	MEN'S B-BALL LCKRM
2,000	WOMEN'S B-BALL LCKRM
2,000	WOMEN'S V-BALL LCKRM
4,000	COACHES OFFICE
2,000	CONFERENCE RMS
11,500	ATHLETIC ADMIN OFFICES
6,000	LOADING DOCK
6,000	STORAGE
10,000	MECHANICAL
132,900	TOTAL
105,000	PARKING STRUCTURE



KEY CONCEPTS

SOUTH OPENING MERGES CLASSROOM BRIDGE AND ARENA ENTRANCE
LARGE SPACE IN LOBBY
ARENA SPACE CAN EXPAND TO LOBBY
SMALLER SCALE ENTRY
MORE DYNAMIC SECTION

ENTRY OPTION 1

KEY CONCEPTS

OPEN AIR PASSAGE ALLOWS VIEWING OF ACTIVITIES WITHIN THE ARENA SPACE
SMALLER LOBBY SPACE
LARGER SCALE ENTRY
CONNECTION BETWEEN ENTRIES
LEVEL ENTRY INTO BUILDING (NO STAIR)
SEPARATION OF CLASSROOM BRIDGE AND ENTRY

ENTRY OPTION 2

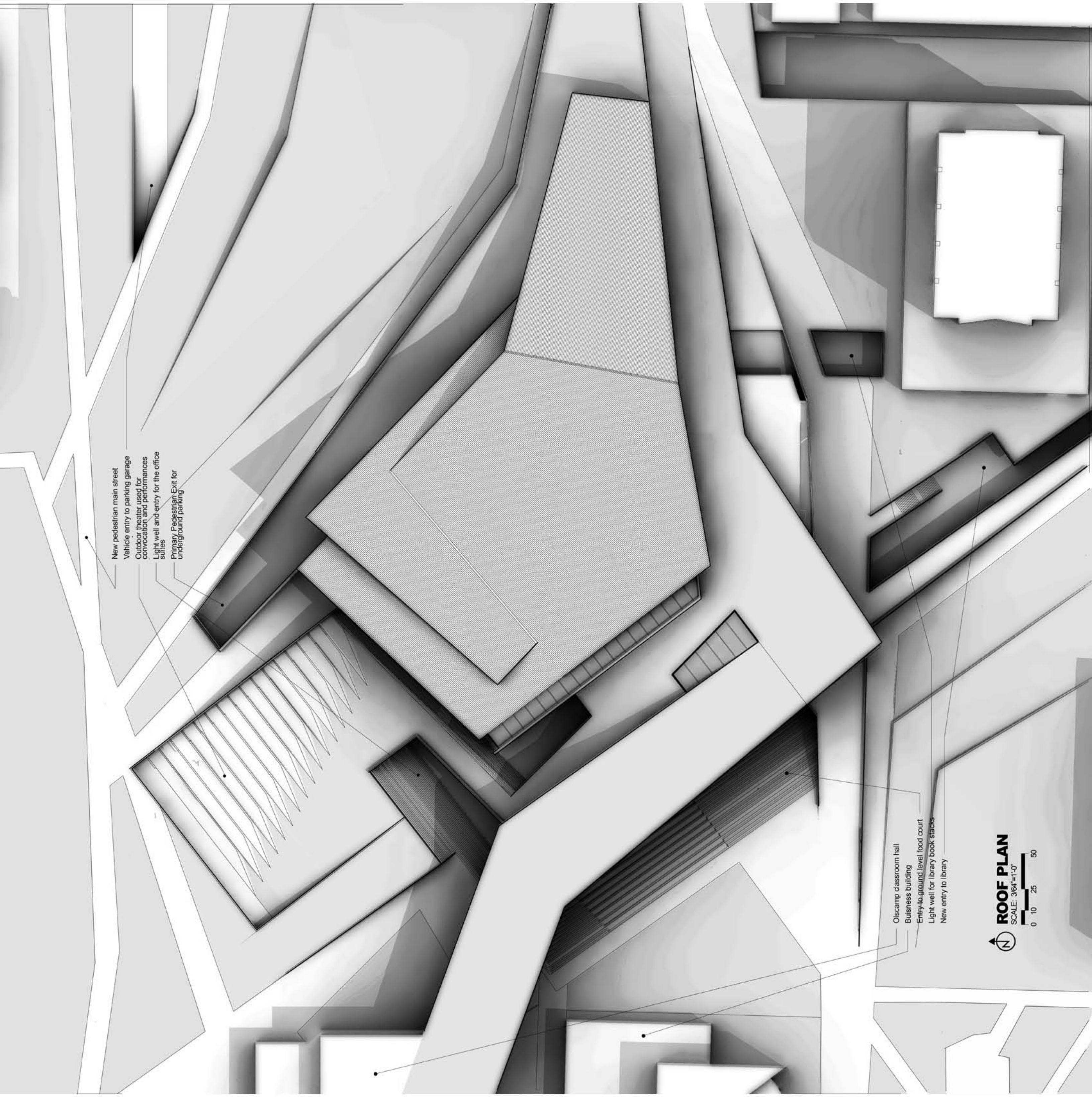
KEY CONCEPTS

SOUTH OPENING IS LARGER IN SCALE WITH OVERHANG MARKING THE LOCATION
LARGE VOLUME IN LOBBY ALLOWING ARENA SPACE TO EXPAND INTO LOBBY SPACE
MORE NATURAL LIGHT ENTERS LOBBY SPACE
CLASSROOM BRIDGE AND ENTRANCE ARE SEPARATED
ENTER LEVEL TO ARENA COURT AND PRACTICE COURT

ENTRY OPTION 3

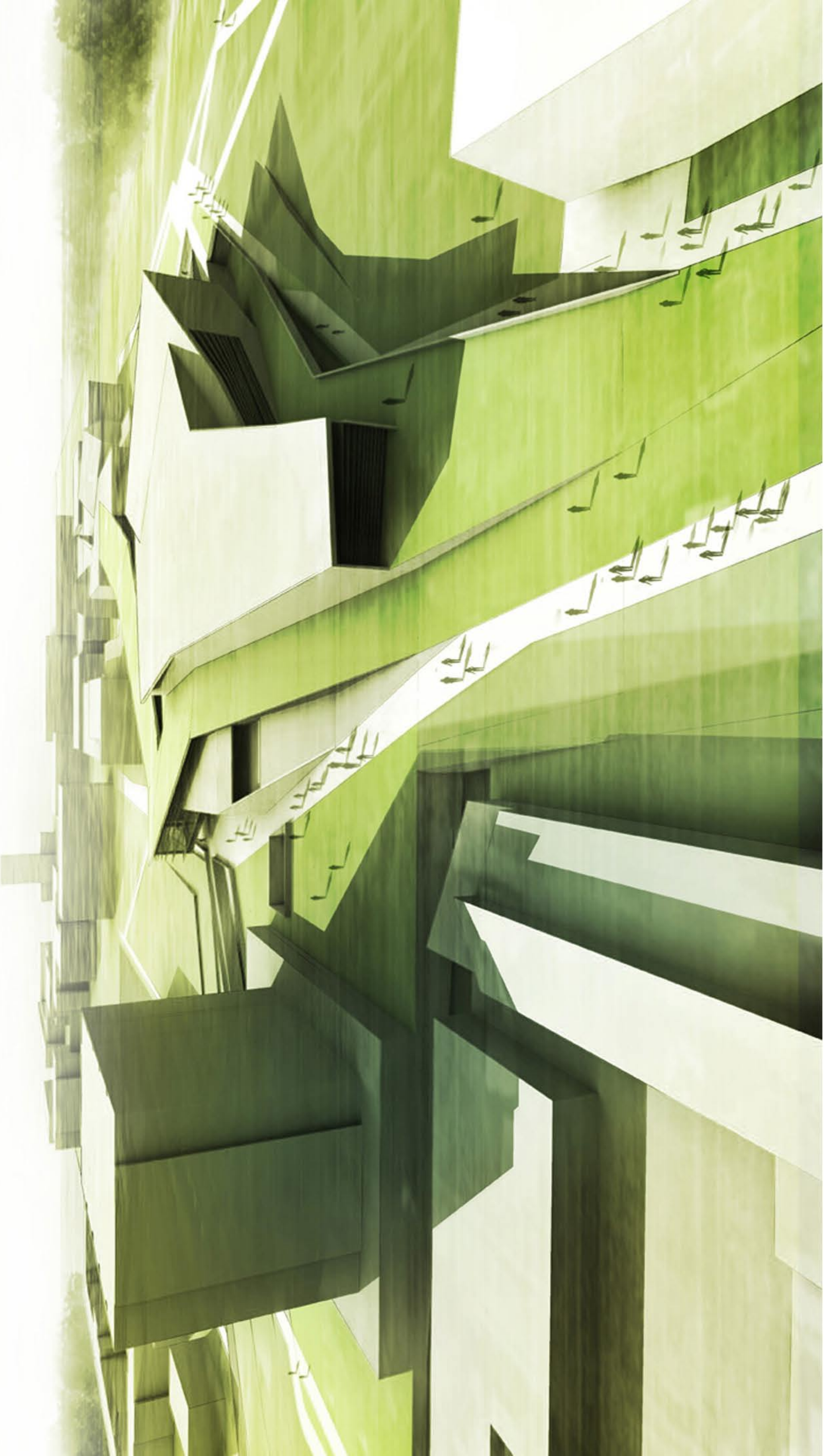
DESIGN THESIS
FINAL DESIGN

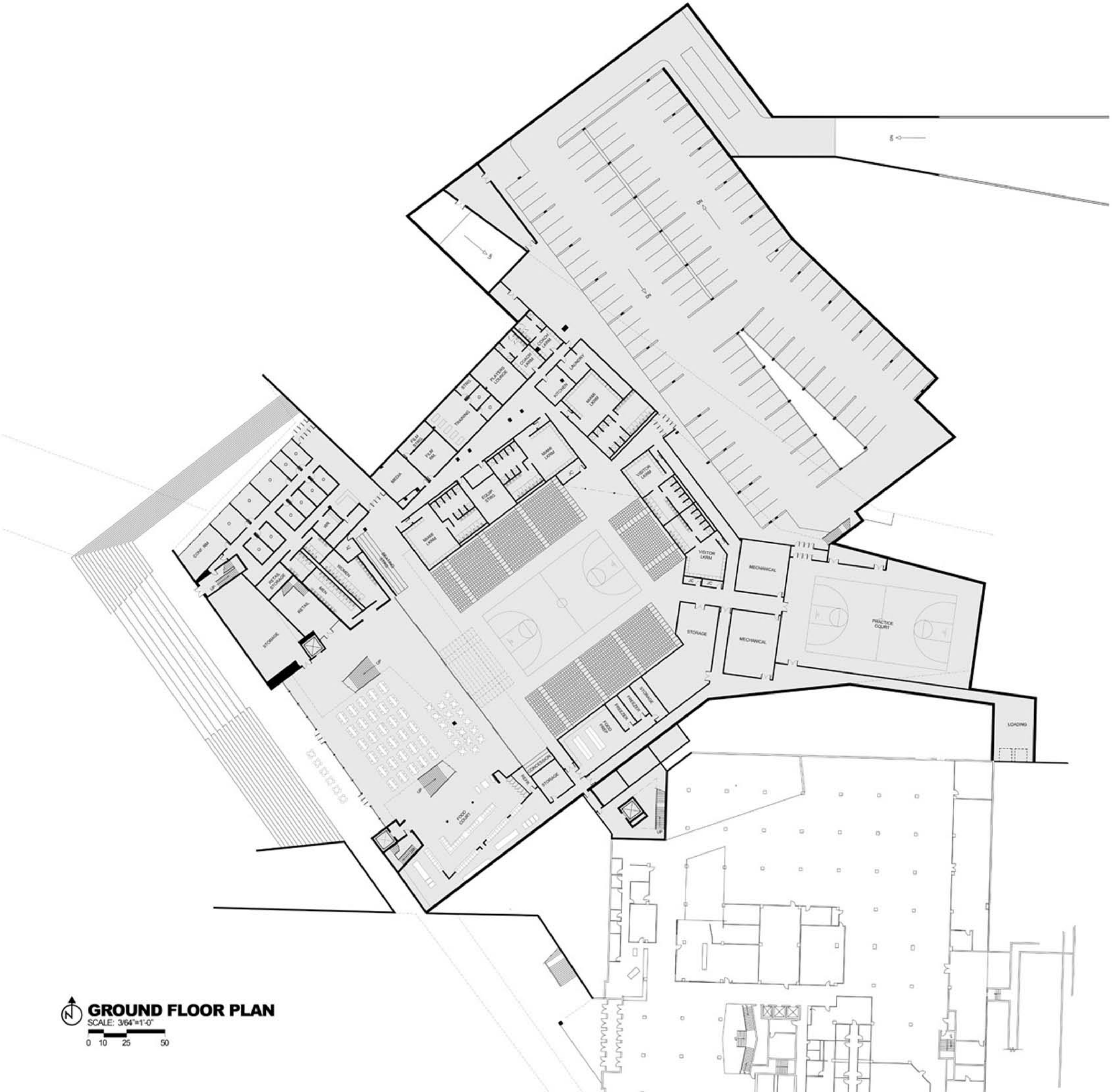
BOWLING GREEN STATE UNIVERSITY ARENA:
INTEGRATING ARENA SPACE WITH THE ACADEMIC ENVIRONMENT



ROOF PLAN

SCALE: 3/64"=1'-0"

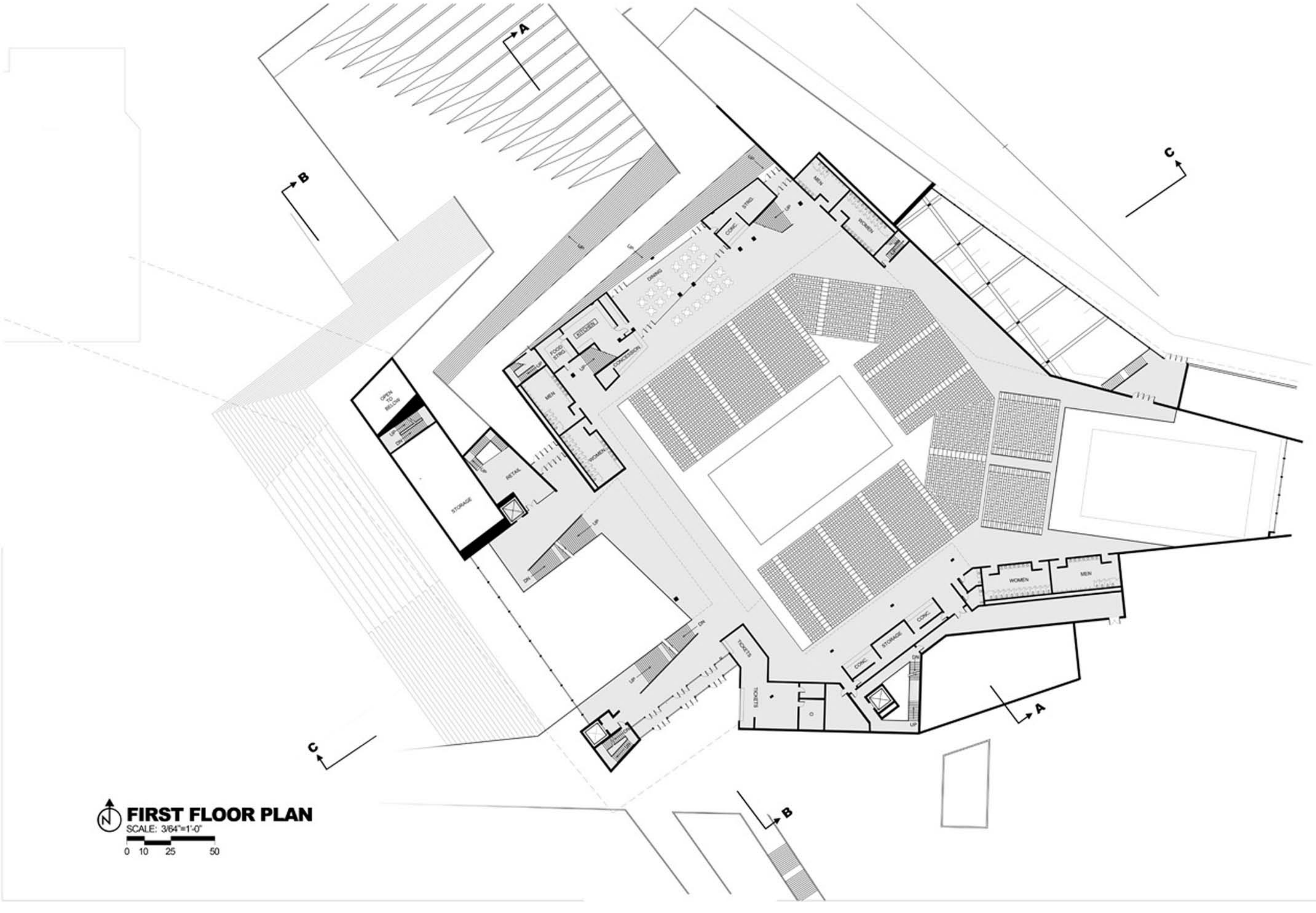




GROUND FLOOR PLAN

SCALE: 3/64"=1'-0"

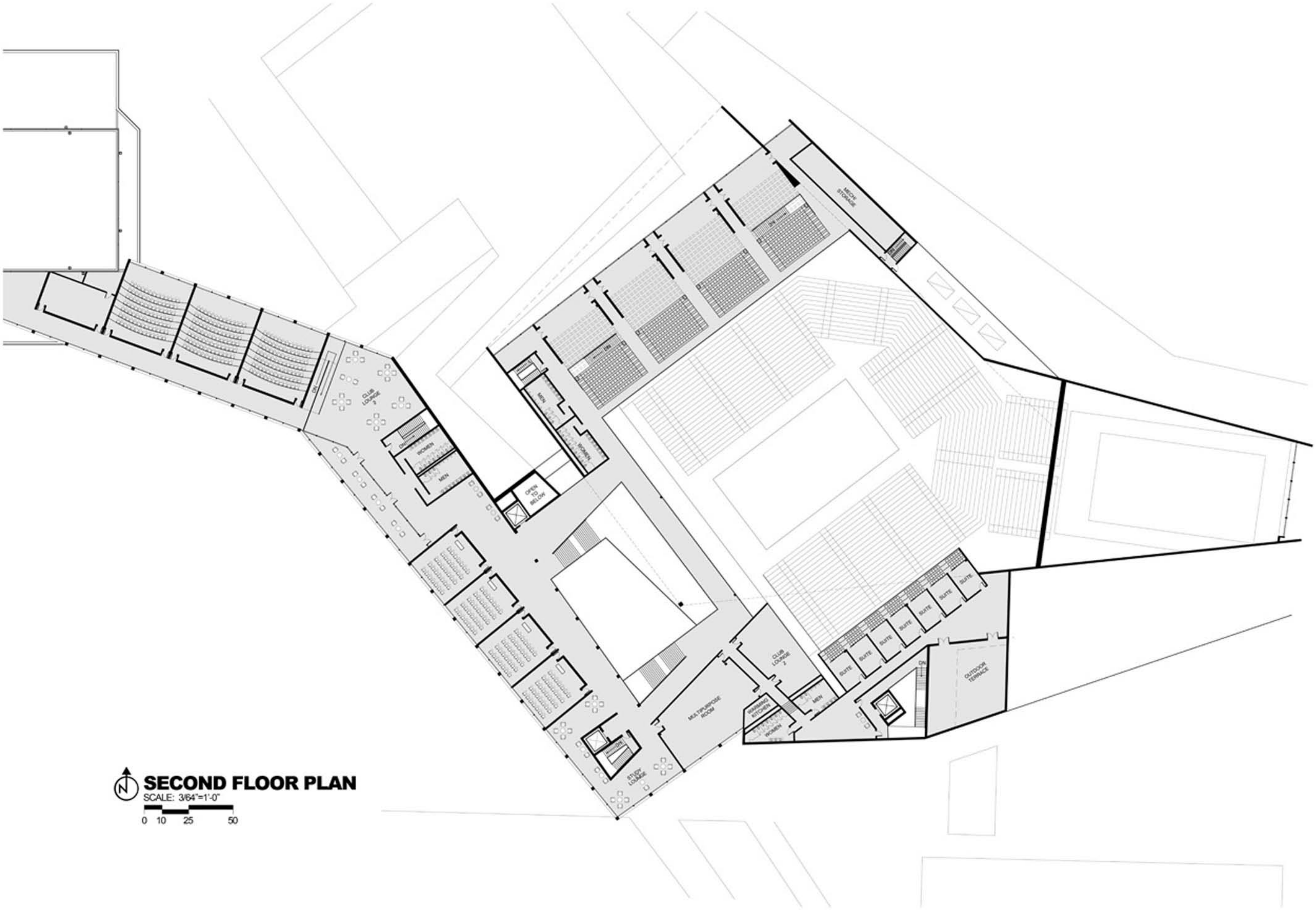
0 10 25 50



FIRST FLOOR PLAN

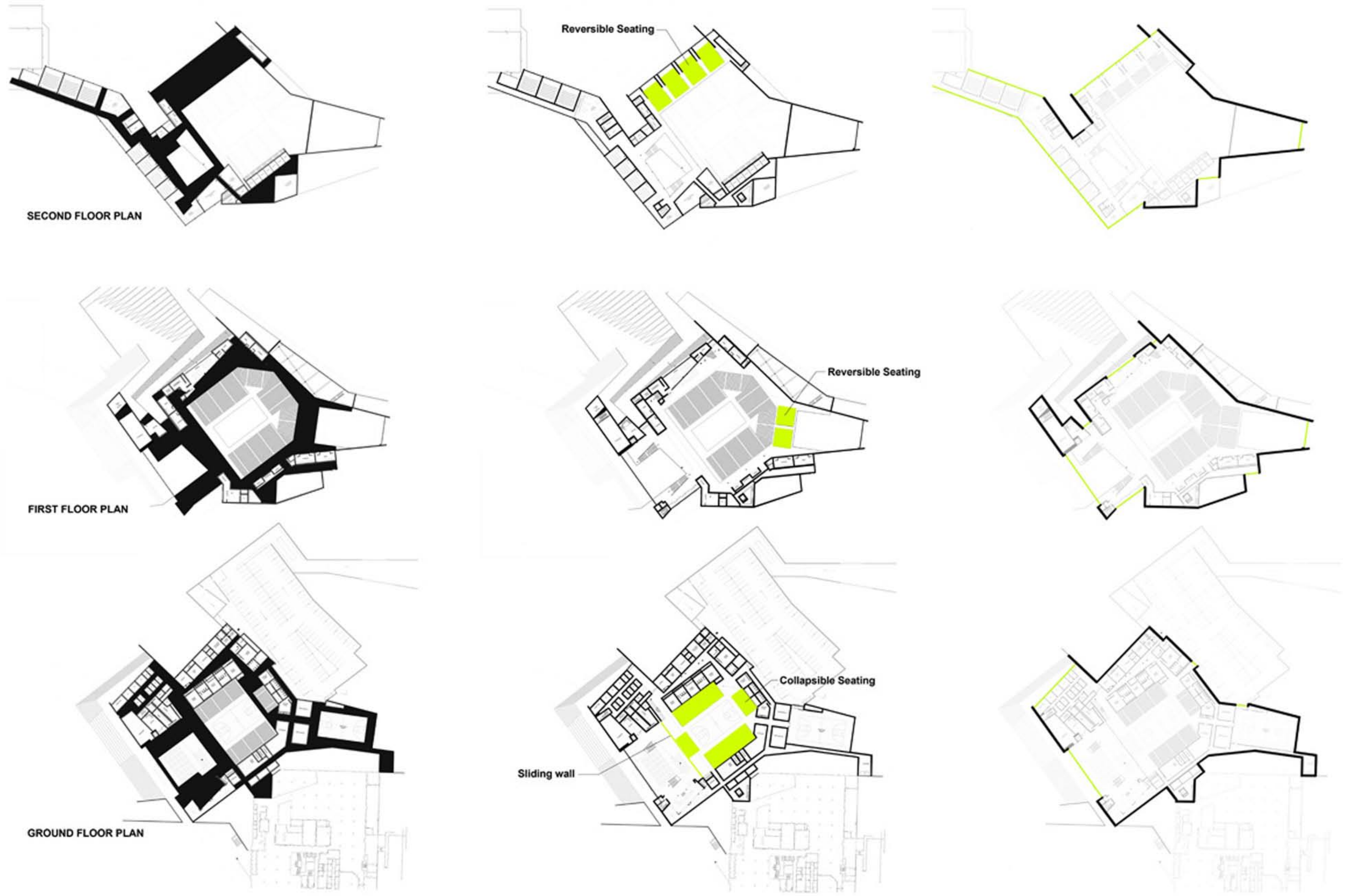
SCALE: 3/64"=1'-0"

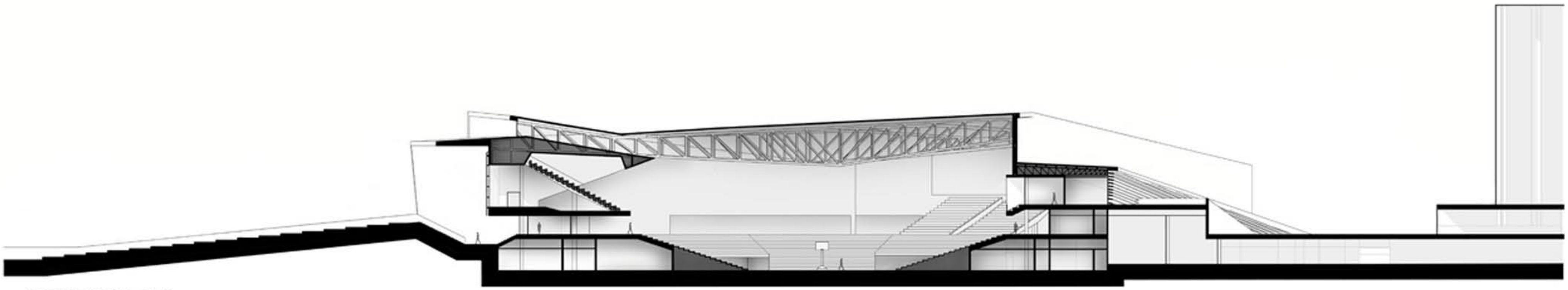
0 10 25 50



SECOND FLOOR PLAN

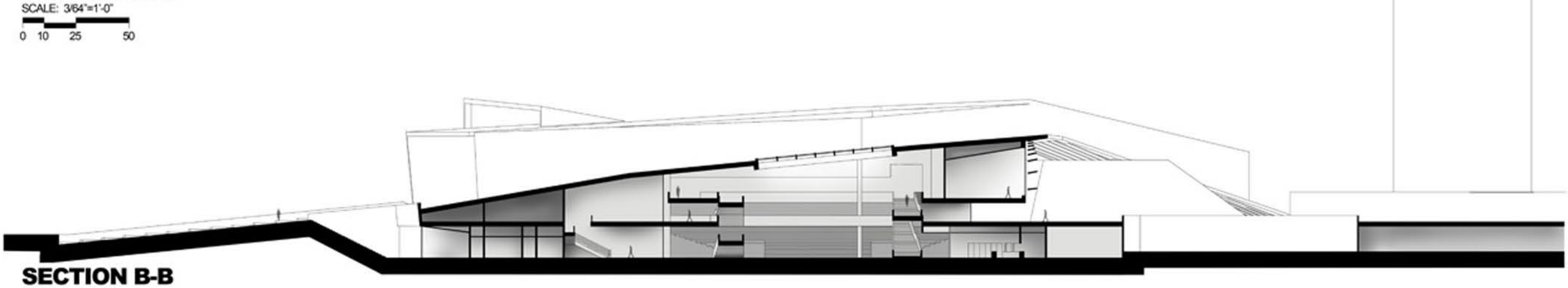
SCALE: 3/64"=1'-0"


CIRCULATION
KINETIC COMPONENTS
ENCLOSURE
Reversible Seating
SECOND FLOOR PLAN
Reversible Seating
FIRST FLOOR PLAN
Collapsible Seating
Sliding wall
GROUND FLOOR PLAN



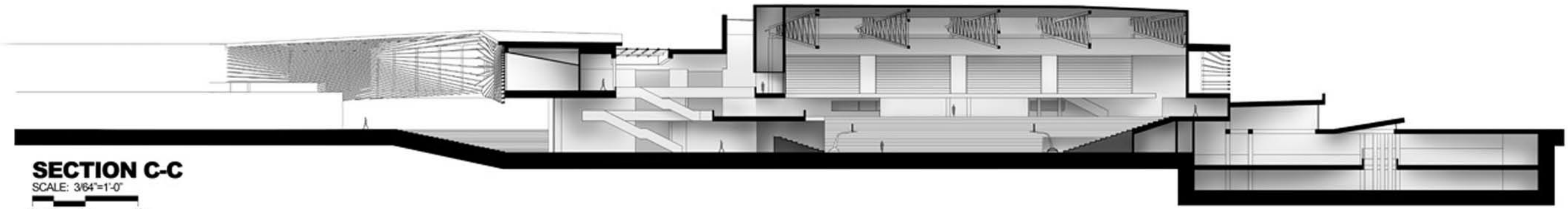
SECTION A-A

SCALE: 3/64"=1'-0"



SECTION B-B

SCALE: 3/64"=1'-0"

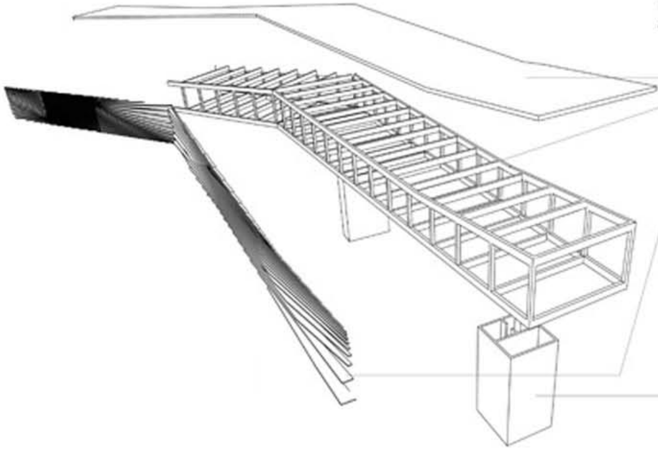


SECTION C-C

SCALE: 3/64"=1'-0"

CLASSROOM BRIDGE

A Vierendeel truss is used to span the two buildings

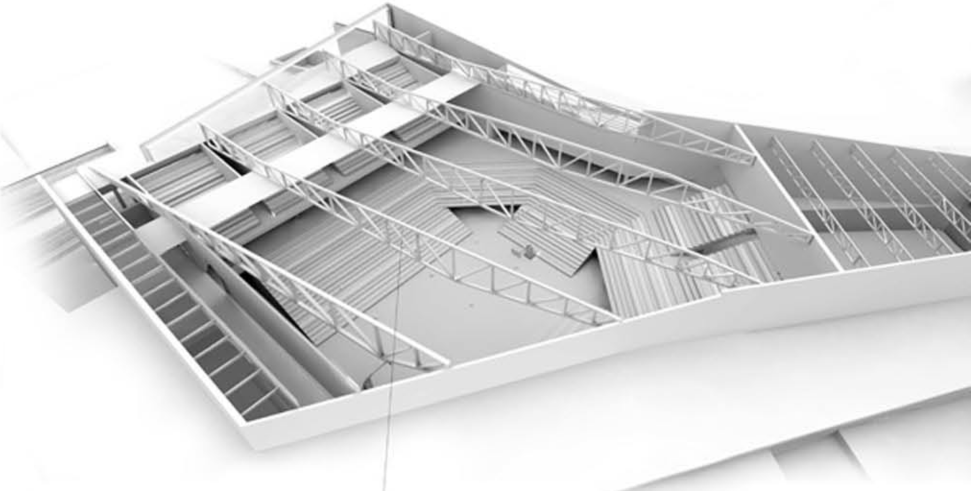


Green Roof

Vierendeel Truss- avoids diagonal members that would detract from the attached louver system

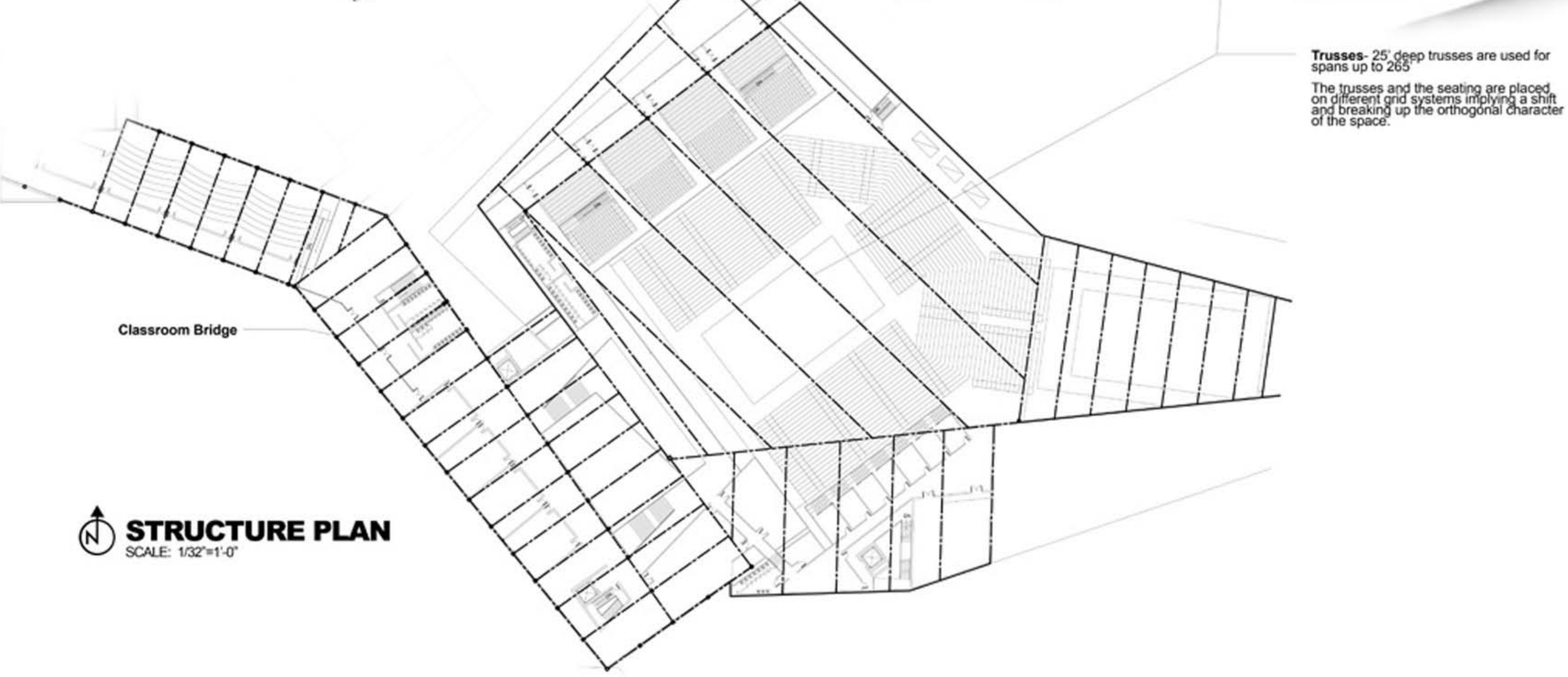
Louver System-attaches to the vertical member of the Vierendeel truss

Piers- carry the loads to the ground and are composed of the elevator and stair shafts



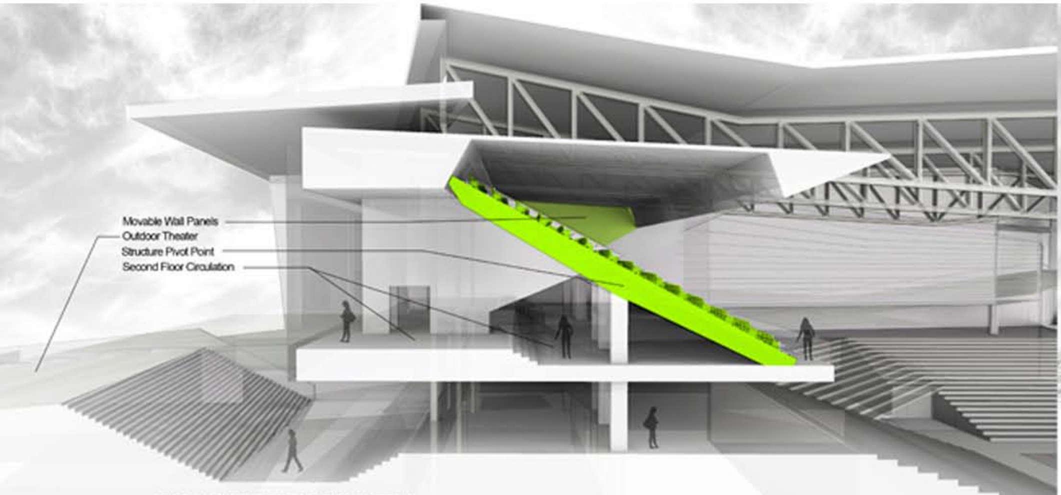
Trusses- 25' deep trusses are used for spans up to 265'

The trusses and the seating are placed on different grid systems implying a shift and breaking up the orthogonal character of the space.

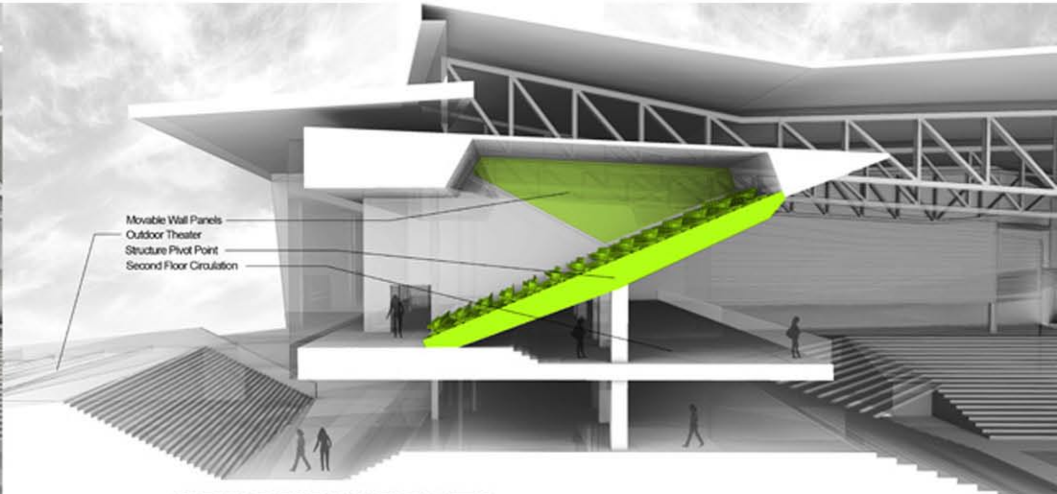


STRUCTURE PLAN

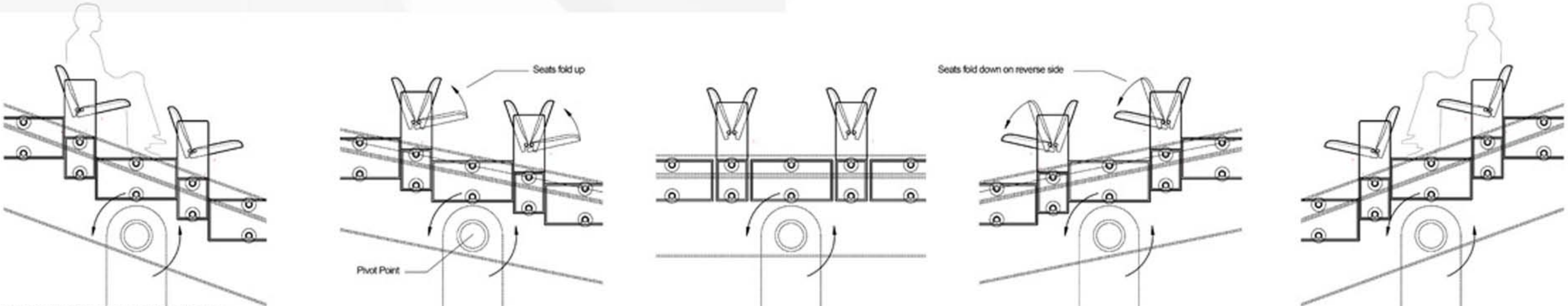
SCALE: 1/32"=1'-0"



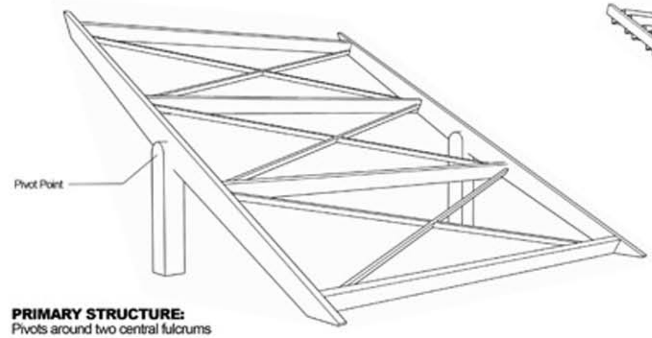
SECTION: COURT SEATING SETUP



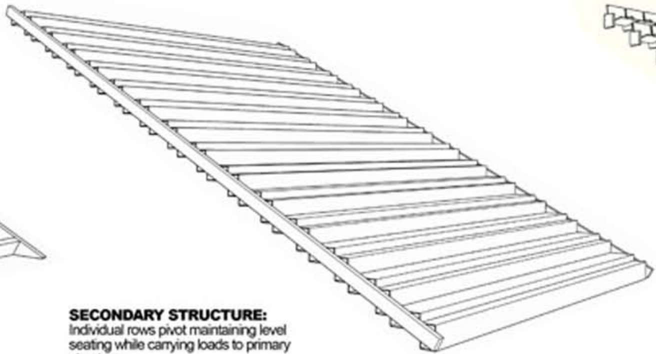
SECTION: LECTURE HALL SEATING SETUP



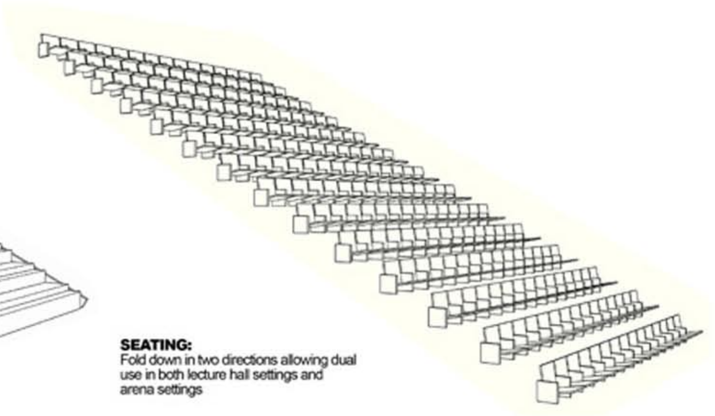
TRANSFORMING SEATING DETAIL
SCALE: 3/4" = 1'-0"



PRIMARY STRUCTURE:
Pivots around two central fulcrums



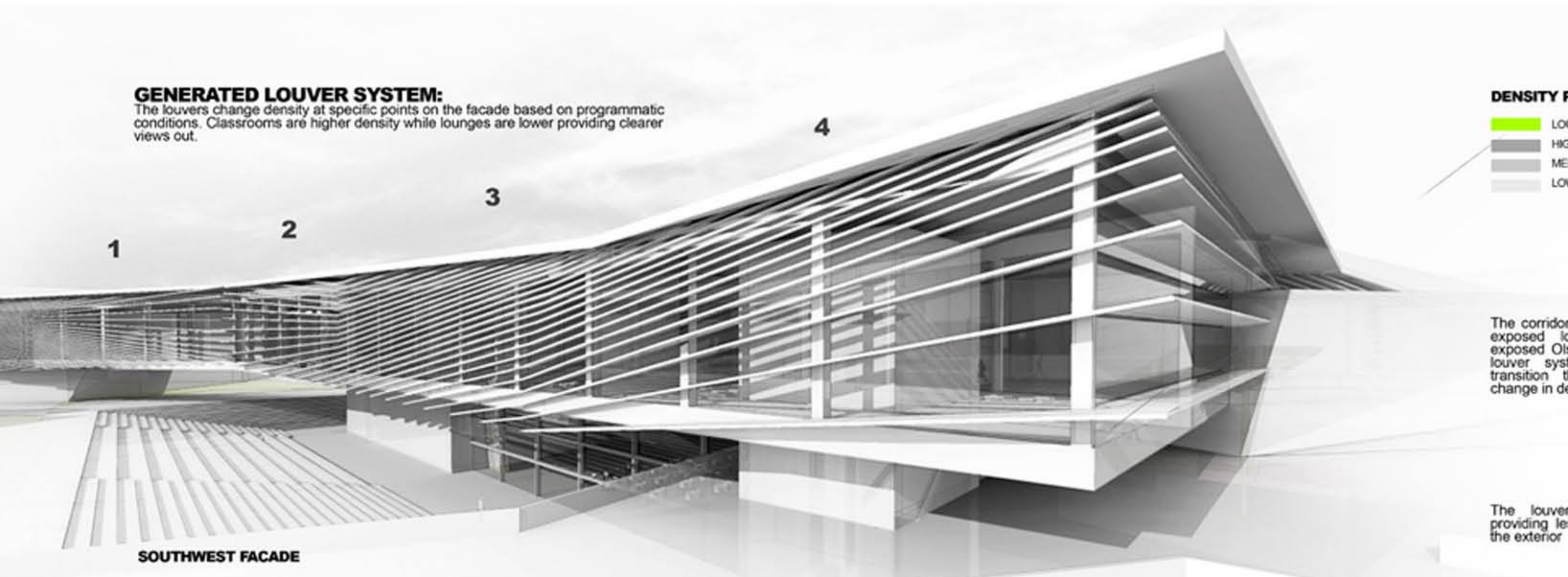
SECONDARY STRUCTURE:
Individual rows pivot maintaining level seating while carrying loads to primary structure.



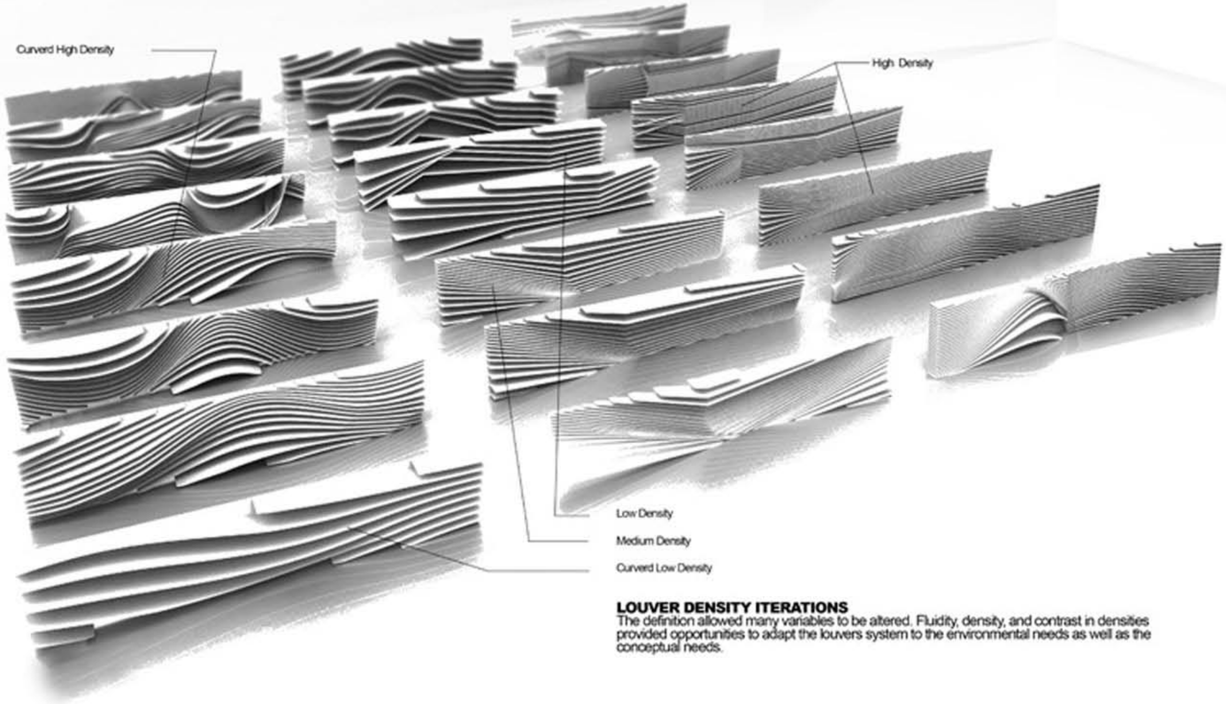
SEATING:
Fold down in two directions allowing dual use in both lecture hall settings and arena settings

GENERATED LOUVER SYSTEM:

The louvers change density at specific points on the facade based on programmatic conditions. Classrooms are higher density while lounges are lower providing clearer views out.



SOUTHWEST FACADE



LOUVER DENSITY ITERATIONS
The definition allowed many variables to be altered. Fluidity, density, and contrast in densities provided opportunities to adapt the louvers system to the environmental needs as well as the conceptual needs.

DENSITY PLACEMENT

🟩 LOUVER LOCATION
▨ HIGH DENSITY
▨ MEDIUM DENSITY
▨ LOW DENSITY

1 The corridor transitions from the exposed lounge to the less exposed Olscamp Hall. Here the louver system articulates this transition through an extreme change in densities.

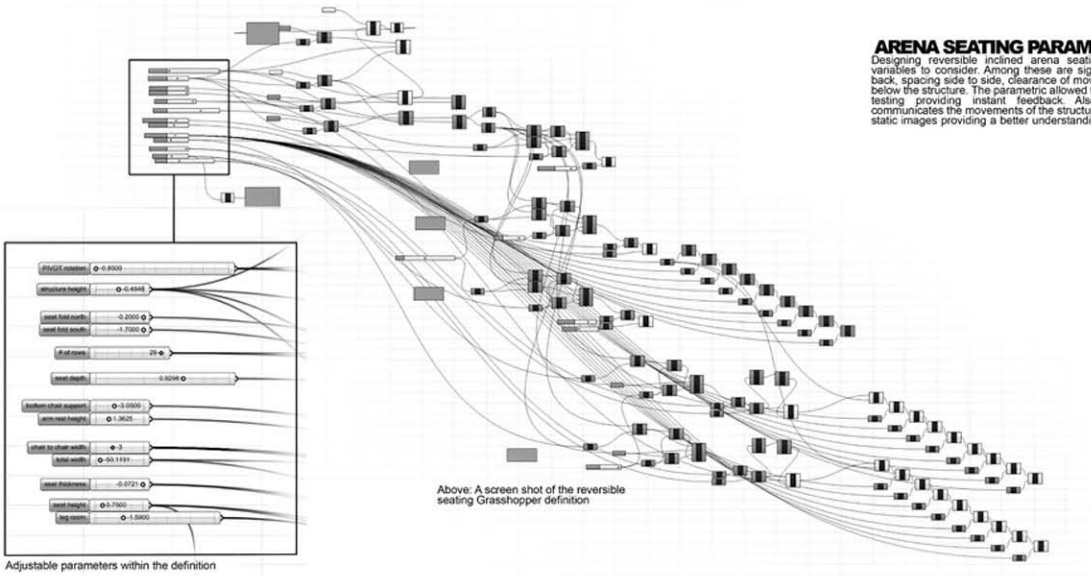
2 The louver system opens up providing less obstructed views to the exterior

3 Classroom spaces require more controlled lighting and less distractions from the exterior. Therefore, the density of the louver system is substantially increased.

4 This location sits at a major intersection of pedestrian traffic while the interior function consists of multipurpose rooms and lounge space. Therefore, the louver density is low allowing the building to open up and become primarily transparent.

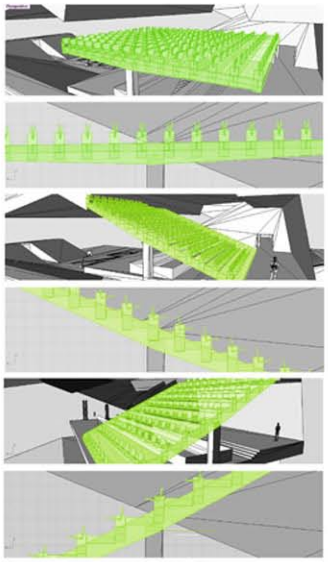
ARENA SEATING PARAMETRIC MODEL

Designing reversible inclined arena seating brings with it many variables to consider. Among these are sight lines, spacing front to back, spacing side to side, clearance of moving parts, and headroom below the structure. The parametric allowed for quick adjustments and testing providing instant feedback. Also, the model visually communicates the movements of the structure more successfully than static images providing a better understanding of all the mechanisms within the seating.



Adjustable parameters within the definition

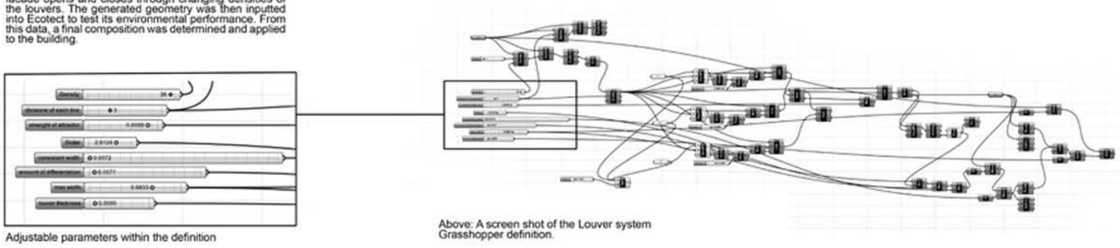
Above: A screen shot of the reversible seating Grasshopper definition



Screenshots of geometry parametrically controlled by the Grasshopper definition

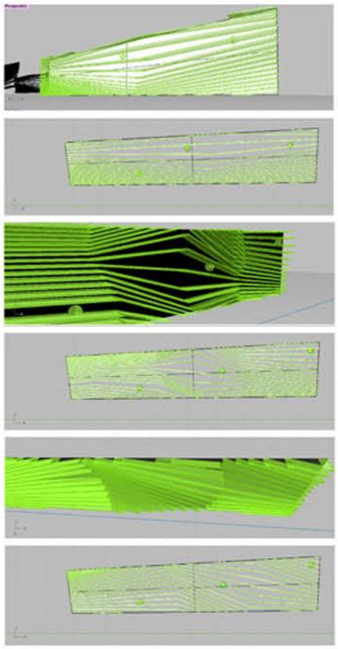
GENERATIVE LOUVER SYSTEM

The generative definition allowed for quick explorations of complex geometries. Based on attractor points, the facade opens and closes through changing densities of the louvers. The generated geometry was then inputted into Ecotect to test its environmental performance. From this data, a final composition was determined and applied to the building.



Adjustable parameters within the definition

Above: A screen shot of the Louver system Grasshopper definition.



Screenshots of geometry parametrically controlled by the Grasshopper definition

ECOTECT ANALYSIS: BOWLING GREEN OHIO

SUMMER : June-August

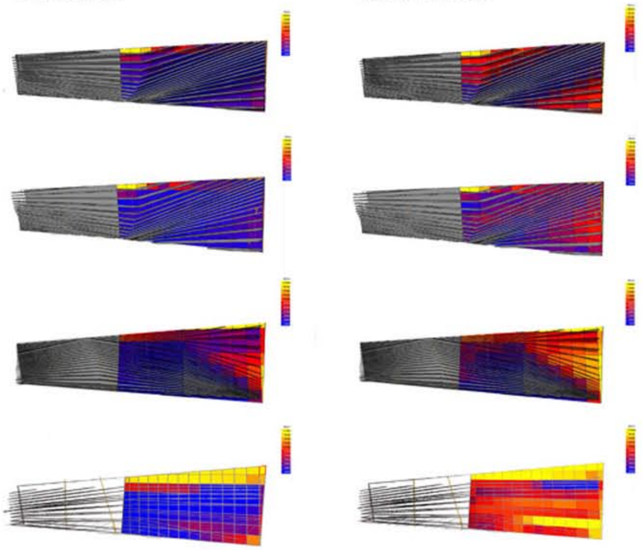
WINTER : December-February

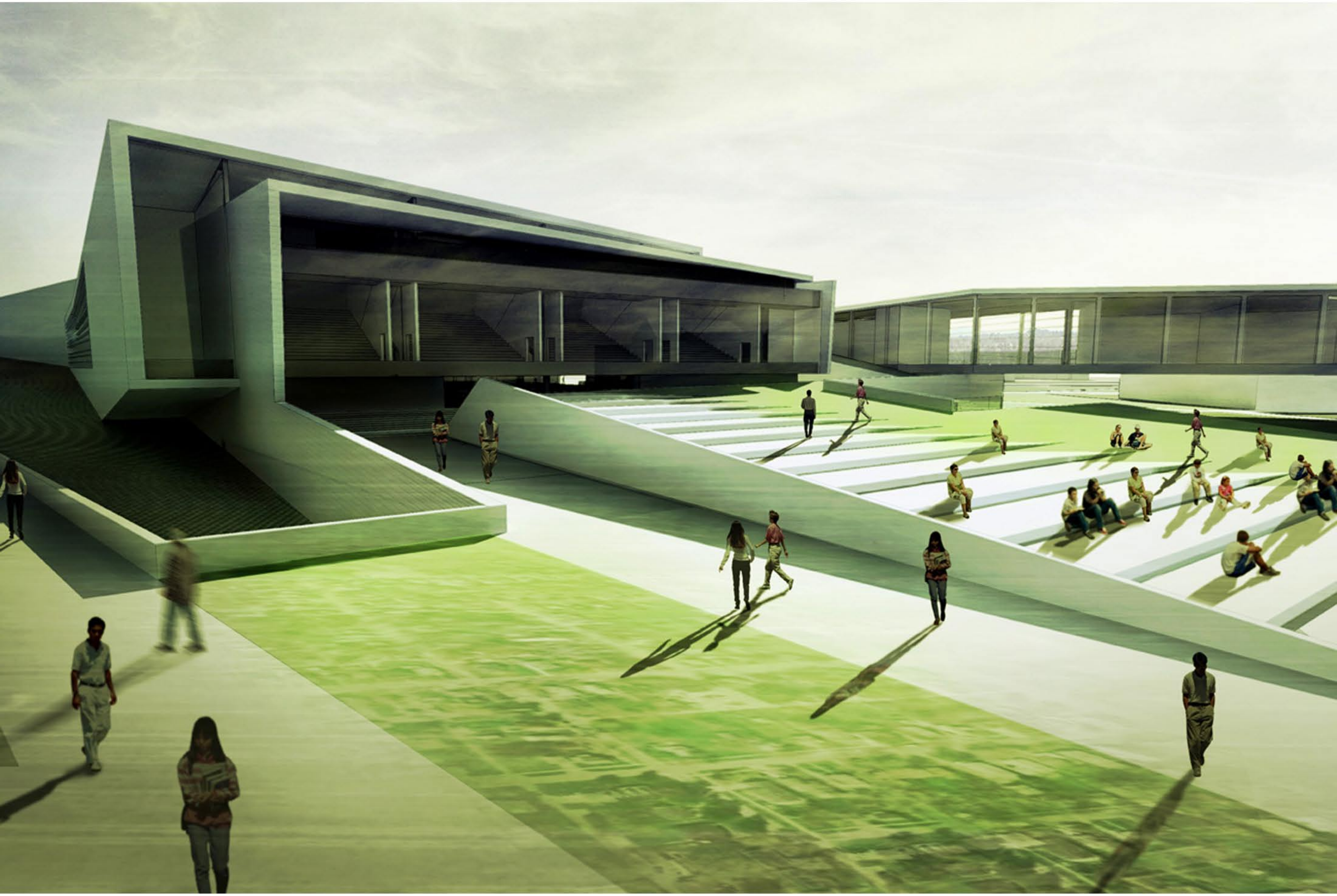
Single width louvers system. The single width does not provide the 3-dimensionality of the multi-width system, but performs slightly better.

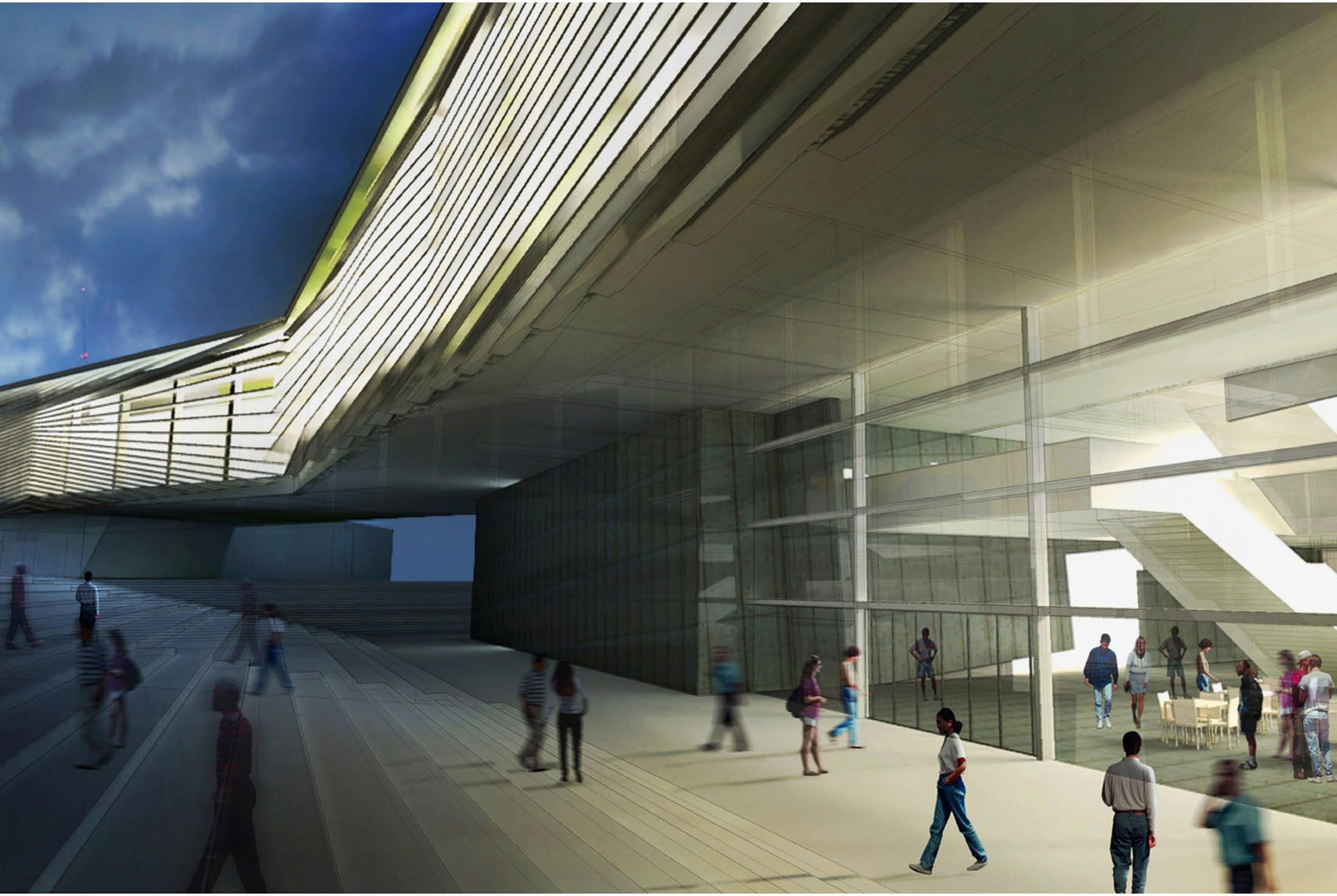
Multi-width Medium Density. Medium density allows for large fluctuation in louver spacing.

Multi-width High Density. The high density reads more as a surface instead of a system of components.

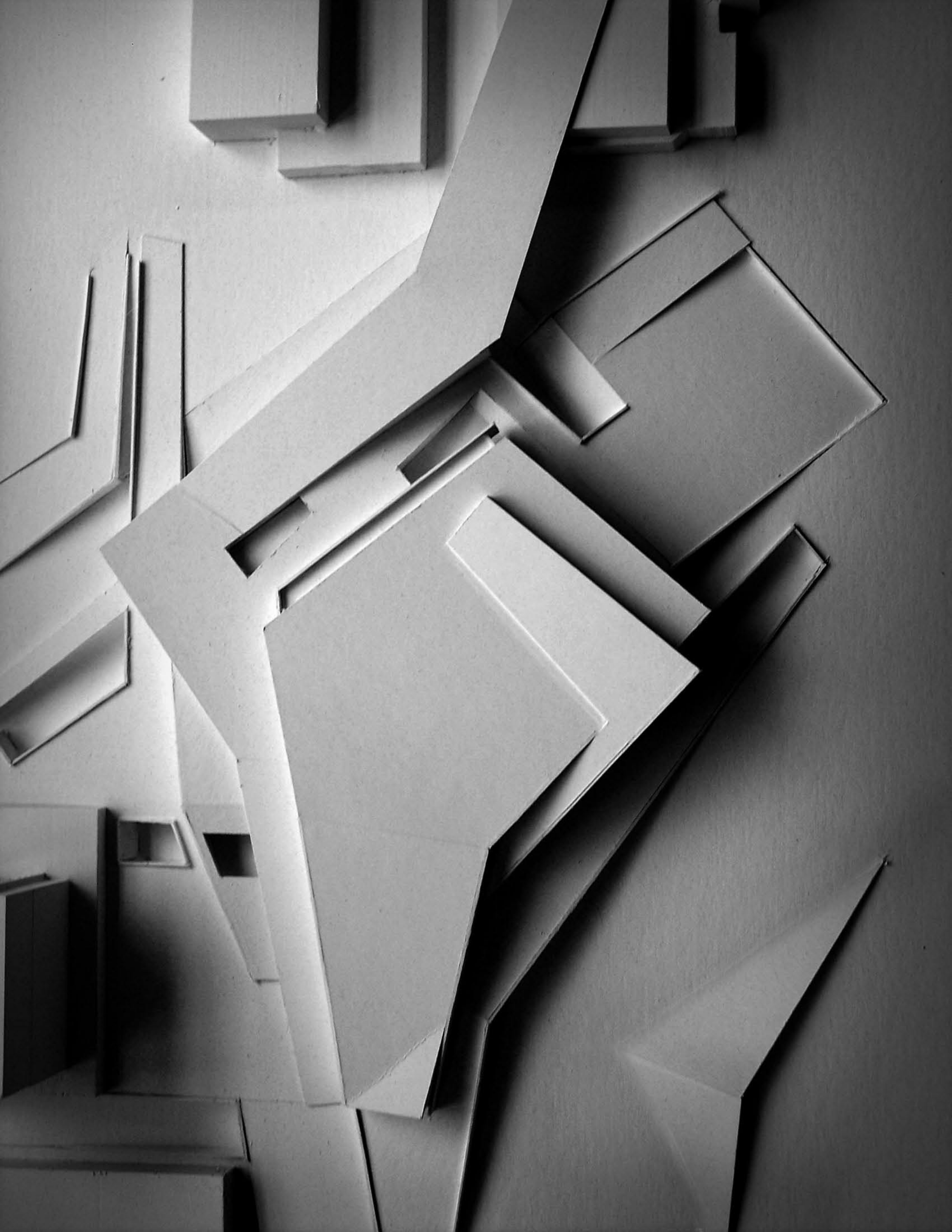
Multi-width Low Density. Low density requires wider louvers, and does not provide the proper visual interruptions for certain functions.

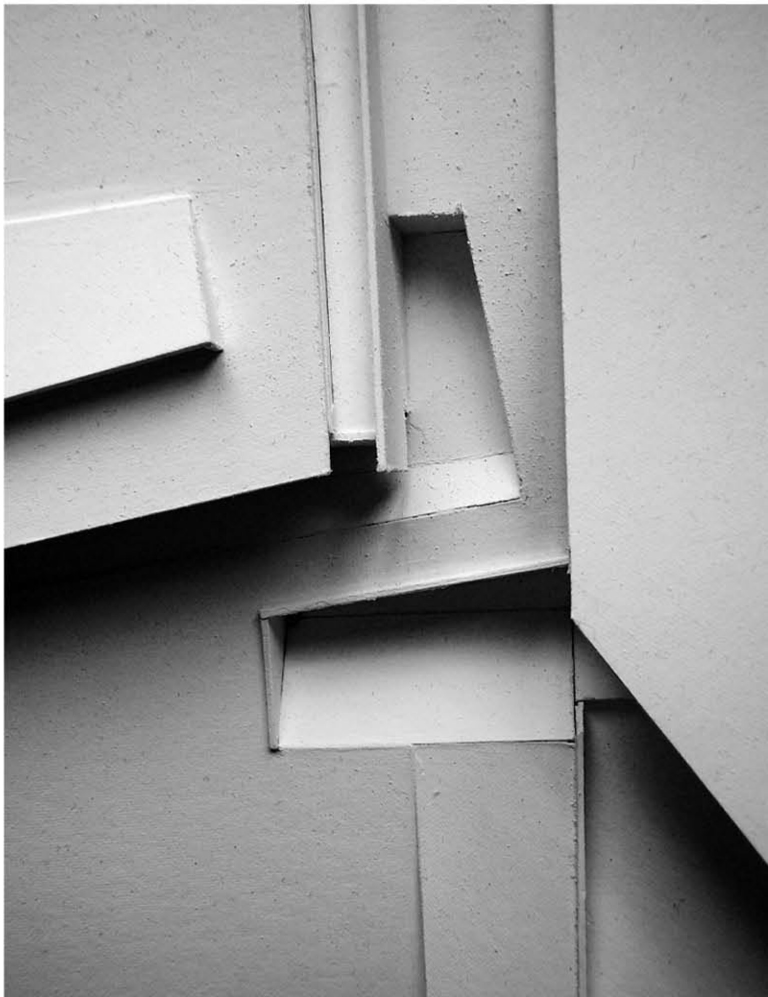






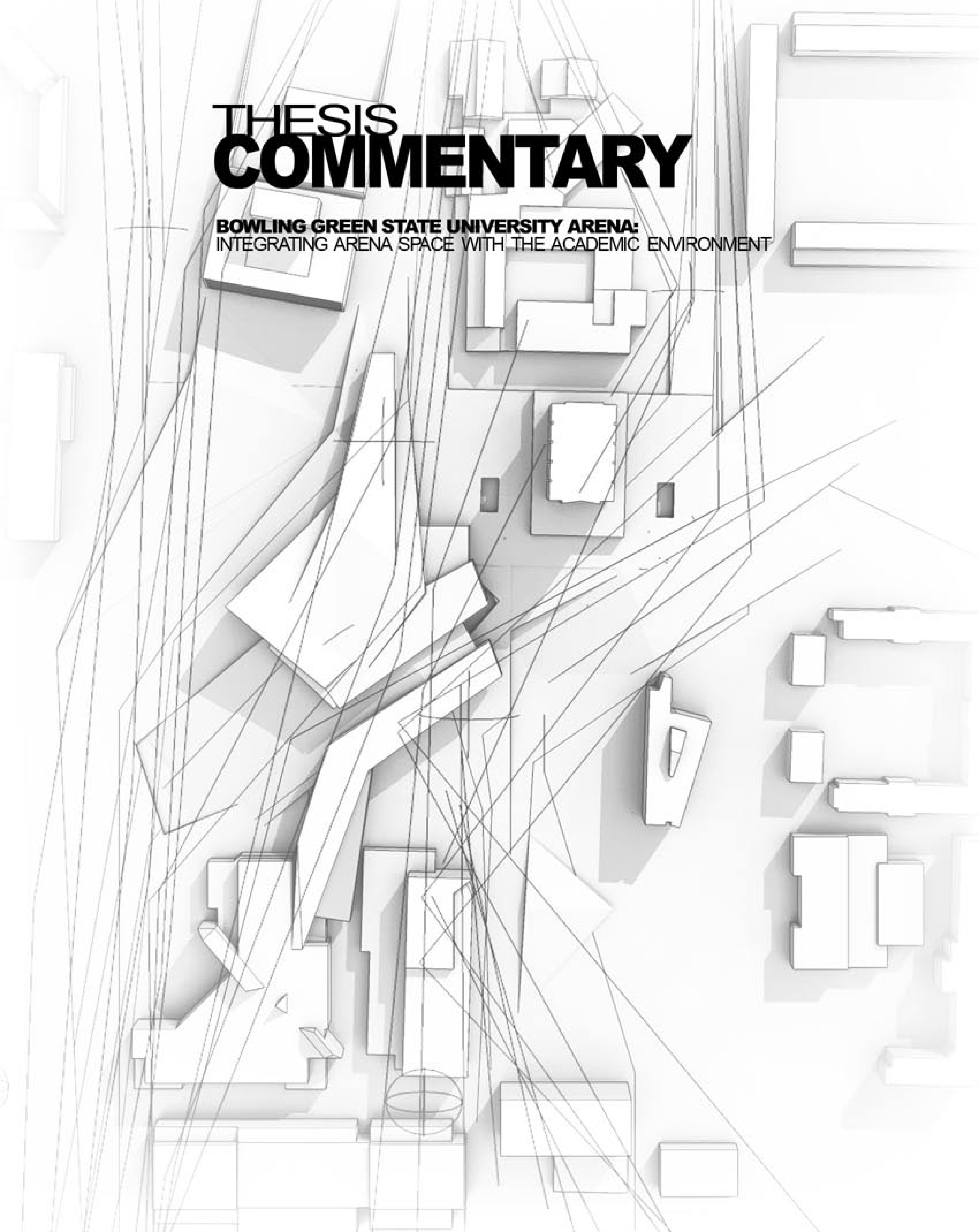

BILL FRACK COURT
FALCONS





THESIS
COMMENTARY

BOWLING GREEN STATE UNIVERSITY ARENA:
INTEGRATING ARENA SPACE WITH THE ACADEMIC ENVIRONMENT



COMMENTARY

ALEX HOGREFE

The written portion of this thesis explored parametric and generative design processes and there incorporation into traditional analog processes. Through the experiments performed, I reached the conclusion that these techniques could play a pivotal role by enriching design possibilities and offering a means of escaping design patterns for ideas that the designer may not have come up without these digital tools.
I hoped to gain from the design portion of this thesis a better understanding of how to and where to use these digital techniques.

Initially, I chose to design an arena primarily because of its many well defined variables which work well with parametric design. For example, seating in an arena requires proper site lines. Therefore, certain variables are used such as pitch, row spacing, seat width, egress, etc. There was also an underlying issue I wanted to address involving how university arenas are placed and used on campuses.

Bowling Green State University has begun construction on their new arena at a location near interstate 75 acting as a billboard for the university as well as a gateway into the campus. The problem is that this location is very disconnected from the main campus. While the arena may be attracting students to the university, it is not enhancing the academic environment to the degree that it could be. This project studies how the campus could benefit from moving the arena to the core of the academic environment, and engaging the arena with everyday student life.
Generative techniques were first used to determine site planning. A script was set up that used site forces (views, pedestrian paths, edge conditions, buildings) to manipulate geometry laid across the proposed new site. Ultimately, I did not use the forms generated from this script. However, the process of going through these steps and evaluating the outcomes led me to the final location and orientation of the arena. This script allowed me to think about the location of the building in a different way than I was accustomed to. I feel this way of thinking led to a provocative, yet well functioning site plan.

Pedestrian paths became a site force that was a leading factor in site decisions as well as formal decisions. A second script was created to explore how pedestrian paths through the site could become more streamlined and efficient. I was interested in where these paths would converge as well as how the arena mass could fit within these paths influencing how the users move through the site. The result was a nontraditional arena massing that concealed its large volume through a series of topographical and formal moves. The ground was slanted up to form roofs, theater seating, and circulation in and out of the building.

Much of the arena has double functions to allow for more flexibility and create a stronger connection to the academic campus. One important issue was to have the building open up to students during non-athletic events. A food court was placed on the ground floor of the arena with clear views to the basketball court. While students are eating, they can watch teams practice or other events going on in the arena. During game day, the space can be used as the main lobby. During conventions or concerts, the court floor can be expanded into the food court space nearly doubling the floor size.

On the third floor, box seating and club lounges overlook the court. By directly connecting these spaces to the nearby library, the box seating provided ideal environments for group study rooms and private study lounges during non-athletic events. These spaces flowed conveniently into the classroom wing of the arena.

The biggest issue I came across with campus arenas was how to utilize the seating. A large portion of the square footage was seating, but was only being used a small percentage of the time. The solution was designing a structural system that converted arena seating into lecture hall seating while also being sound insulated and conducive to learning. Here, Grasshopper played an important role in controlling the complex movements of the structures allowing me to analyze site lines, slope, chair folding mechanisms, and clearances between moving parts. I was able to test and tweak many different setups before deciding on the final solution. The invertible seating works on the same principle as a teeter-totter. Through one simple move, the arena seating can be inverted to sound isolated lecture hall seating facing the opposite direction. The lecture hall seating also provides proper site lines to the outdoor theater providing conditioned seating for shows outside. The lecture hall seating connects directly to the classroom bridge, which extends into Olscamp Hall, a multi-classroom /lecture hall building. The many double functions of the arena encourage users to interact more dynamically with each other, as well as with the building.

Finally, the concept of shifting pedestrian paths was continued to the façade, in the articulation of the south facing louver system. The problem consisted of designing a shading system that offered different levels of openness or privacy based on functional needs. Grasshopper provided the tools needed to 1) control the complexity of the geometry, 2) test out the environmental issues of solar heat gains, and 3) meet the functional requirements of privacy. A script was created to change the density of the louver system based on the placement of attractor points at precise locations on the façade. Classrooms required the louvers to be dense to avoid visual distractions inside the space looking out. However, in lounge spaces, views to the outside were desired. Therefore, the louver system is less dense providing clear views out. Since the louvers were changing densities throughout the façade, it was required to have individual louvers change width according to their proximity to one another in order to provide consistent shading. Therefore, the grasshopper definition accounted for this adjustment as well. Many iterations were generated and then tested in Ecotect to determine their legitimacy in environmental performance. Through this analysis, a final louver system was determined and applied to the south arena façade.

The integration of generative and parametric design into my design process was difficult and uncomfortable in the beginning. After 6 years of architecture school, I had grown accustomed to my own unique process of designing. Implementing generative design techniques brings with it a sense of uncertainty and loss of control. However, I learned this was due to me forcing these techniques where they were not needed. For example, early in the design portion of the project, I wanted to generate a building form from site forces by a single Grasshopper definition. But, I did not know exactly how I wanted to go about doing this, nor did I have any rules that would guide me to a successful solution. While the script ultimately gave me a better understanding of how I wanted to place the building on the site and led me in a direction I probably would not have gone without it, it was a failure in the sense that it was too vague and abstract, and did not offer any formal solutions.
The following script (the pedestrian paths definition) had a much clearer and more defined problem to be solved. I knew I wanted to test out more efficient pedestrian paths. I knew that where these paths converged, I wanted the building to open up through transparency and entry. And finally, the building form could be defined by the areas between the generated pedestrian paths. These rules allowed me to create a script that performed better, while extracting much more useful information.

An important idea that took me a while to grip was the idea that one script was not going to design the entire building. What I mean by this is that there are too many variables or decisions to be made during a project of this magnitude. It is unrealistic and irresponsible to expect a single script to do too much. From my experience with this project, understanding the questions is essential opposed to hoping that a generative script will just create something that is interesting and *could work*. And as obvious as that sounds, I set out at the beginning of the project having exactly that mindset that generative design would produce unexpected forms that would work better than what I as a designer could come up with. What I realized is that generative and parametric processes took questions I had, and allowed me to quickly test out many solutions, no matter the complexity of the procedures being performed. In other words, they were abstracting the problem allowing me to be more playful with the design. In the case of the louver system and invertible seating, I was able to test out 20 or 30 possibilities each, while still maintaining full control of the complex geometry over a very short time span. Without these tools, both the louver system and invertible seating could not have been developed to the degree that they were.

My understanding of my own design process and where I want to take it, from the end of the written portion of this project to this point, has changed dramatically. I am much more comfortable with the integration of generative and parametric design into my previous design process. I am much more willing to step outside my comfort zone knowing now, appropriate times to do so, and the infinite results that can be achieved. It is this understanding and connection with one’s own design process that I feel allows a designer to design with more originality and sensitivity to the project needs.