The Pennsylvania State University

The Graduate School

College of Arts and Architecture

THE EPHEMERAL DREAM SPACE:

(RE) ACTIVATING AN EVOCATIVE ARCHITECTURE THROUGH COMPUTATIONAL DEVICES AND BODILY INTERACTION

A Thesis in

Architecture

by

Nasim Motalebi

Submitted in Partial Fulfillment

of the Requirements

for the degree of

Master of Science

August 2017

The thesis of Nasim Motalebi was reviewed and approved* by the following:

Marcus Shaffer
Assistant Professor of Architecture
Thesis Adviser

Jose Pinto Duarte
Stuckeman Chair in Design Innovation; Director of the Stuckeman Center for Design Computing

Andrew Hieronymi
Assistant Professor of Visual Arts in New Media

Ute Poerschke
Professor of Architecture; Director of graduate studies

*Signatures are on file in the Graduate School.

ABSTRACT

This thesis proposes an architecture that fosters a connection between the virtual and the physical world for people (bodies) who are absorbed by the possibility of doing almost anything and of being almost anywhere in the virtual world. The proposed architectural space, presented in the form of a physical prototype, is one that simulates the wildly transformative capabilities of a "Virtopia"[1]. Virtopia is the virtual utopia, where digital code gives birth to buildings and cities in the shape of anything imagined by architects; spaces that defy the rules of the physical world. It is the aim of this research to explore digital augmentation of the physical in order to promote spaces that are transformable, connected, and responsive to the bodily movements and desires of their inhabitants. Today, technologies such as programmable actuators, open source software, sensors, and so on, counteract the physical limitations of transformable architecture and provide the freedom for architects to bring physical manifestation to a Virtopia.

With a focus on horizontal space definers[2] and bodily interaction on the ground plane, an interactive, shape-shifting, and evocative floor was designed and built. The floor, aimed to promote the culture of floor living as a social instigator and symbol of communal life, is designed to respond to the movements of those sitting on it. Using visual sensory systems and programmable actuators, the floor is an ephemeral landscape that physically transforms its shape and creates different experiences for the users [specifically for two users in this prototype].

This research aimed to create a design example of an architectural space that could react to the embodied interactions of its users. The thesis is structured to demonstrate the study on material processes, sensory inputs, programmable outputs, and actuators. Towards the end, the final decisions and construction process are explained.

Keywords: Virtopia, DreamSpace, Ephemeral Architecture, Interactive Architecture, Computation, Bodily Interaction.

List of Figures

List of Tables

ACKNOWLEDGMENTS

My kindest regards go to my research advisor Marcus Shaffer, Associate Professor of the Stuckeman School of Architecture at the Pennsylvania State University. His honest and professional advice steered me to where I stand today as a researcher. I am very much honored to have had such mentor and support during my master's education.

In addition, I would like to thank my committee whom their passionate input led this research towards a validated work: Jose Pinto Duarte, the chair of the Stuckeman Center of Design Computing, for his patience and trust in my decisions. And Andrew Hieronymi, Assistant Professor from the Penn State School of Visual Arts, for his kind guidance and creative criticism.

I would also like to acknowledge Benay Gursoy for her fresh approach and perseverance, Jamie Heilman, Negar Fadaei, Allen Sutley, and Steve White for their 24/7 assistance and hands-on advice from the Digital Fabrication lab and the Stuckeman woodshop, Ute Poerschke, the Graduate Advisor of the Stuckeman School of Architecture, for the kindest advice, and Mehrdad Hadighi, the head of the Stuckeman School of Architecture, for his thoughtful support at all times. I am honored to have known these characters and worked with them for the last two years.

Finally, I must express my gratitude to my parents for their unconditional love and support which encouraged me throughout my master's education. They were my main source of inspiration, to grow strong and overcome all hurdles along the way.

Nasim Motalebi

1. INTRODUCTION

Buildings, inherently known as static artifacts[3], have raised issues in adapting to the inhabitant's desires and reflecting their impact on their living condition. The architecture of brick and mortar provides space for individuals to adjust and re-adjust their living environment, re-furnish, decorate, and so on. However, this static architecture rarely adjusts itself to the behavior and movement of the individual inhabitant. This opens up interesting questions: how can architecture, as the second skin to the collective body[4], be adjusted for the individual behavior in a social context? Can computationally augmented physical architecture – an ephemeral DreamSpace – enhance and transform human behavior in a social context?

The aim of this research is to design a collective (in a social sense) architectural element that readjusts itself to the movements of individuals in space; a communal DreamSpace. The significance lies under an architecture that combines the potentials of the virtual with the physicality of our constructed world – a Virtopia. Such architecture could become the ultimate platform for social activity, capable of being manipulated by and for individual behaviors in communal space. This manipulation of space goes beyond the image of the architectural space; function and the spatial experience are enhanced using computational devices. While the DreamSpace has been advanced, both physically and virtually throughout the history of the 20th and 21st century architecture, today computational tools have reinvigorated this project; the DreamSpace has come back to life with digitally mediated technologies and their potentials.

1.1. Virtopia as an Architectural Project

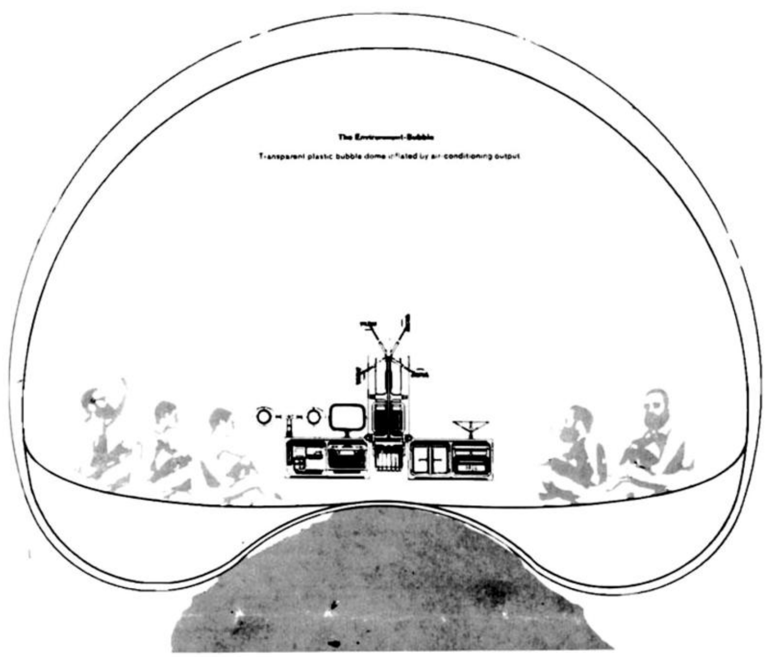
1.1.1. The 20th Century: Hitting the Limitations of the Physical Space

In the early 1960s, frustrated with capitalism and homogeneity of Modernism and the mass produced products, architects and designers in the capital west wanted to liberate themselves from an authoritarian society. The mid-60s was the peak of the Psychedelic Movement, where youth found their liberation in psychedelic drugs, communal activities, and the arts. Not only did the Psychedelic Movement Shift perceptual awareness, but it made a cultural turn that was aligned with a Post-Modern, anti-authoritarian movement. The Psychedelic Movement then was known as a counterculture lifestyle that had influenced art, design, clothing, and architecture and the built space.

The members of the Psychedelic Movement, the Psychedelics, envisioned spaces that blurred the boundaries between reality and illusion. Their extraordinary experiences from LSD[5] – another type of virtual world – demanded and inspired extraordinary settings. The Psychedelics were hypnotized by the idea of manifesting their dreams in the material world, and this led to a disconnection from "reality". How could they go back to their ordinary everyday environment after a "journey to the deepest recess of inner space"[6]? Alastair Gordon writes of the 1960's Psychedelics, whose drug-altered minds were driven to imagining liquefied simulations of the material world; spaces that were flowing, non-hierarchical environments in a constant state of mutation. Those space or the quality of spaces was to be *elastic*, *acrobatic*, *infinite*, and *ephemeral*; the old measurements of space were ridiculed. They wanted spaces that could be anything and spaces that could adapt any function within itself. *CrashPads* were an early example that embraced different activities onto itself[7]. Such communal spaces, plus the drug-enhanced experiences of space, impacted how designers approached architecture and defined new spatial experiences.

[Figure 1] Architecture Manifestations. Top from left to right: Grid Architecture, No-Stop City. Bottom from left to right: New Babylon, Crashpads, Interiors by Verner Panton. Details can be found in appendix A. (Collaged by Author, 2017)

The manifestations for a *No-Stop City* by Archizoom[8] (1969), the *New Babylon* by Constant Nieuwenhuys[9] (1959-74), The *Phantasy Landscape* by Verner Panton[10] (1968), and the *Environmental Bubble* by Banham and Dallegret[11] (1965) all tried to design unconventional spaces that blurred the boundaries between dream and reality. Even though the efforts of those architects were valuable in shaping a new manifestation in architectural theory, they were not entirely successful in achieving what they envisioned.



[Figure 2] The environmental Bubble. Banham Reyner,"A house is not a home", Art in America 2, no4, 1965.

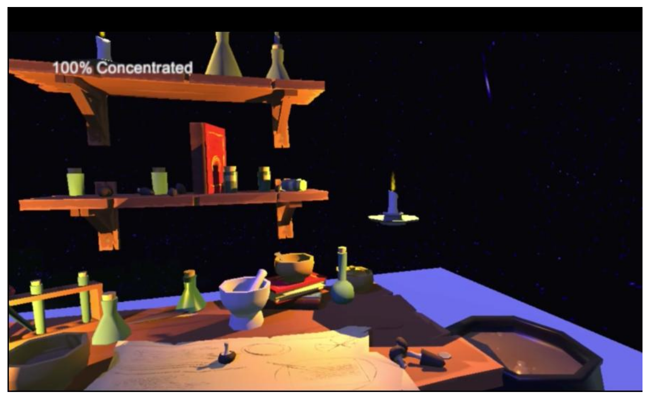
The social divide against the use of technology prevented technological manipulation of architecture for manifesting the DreamSpace. While some entirely rejected technology as an evil force against spirituality, some others wanted to use technology to counteract authoritarian architectural space. In the Environmental Bubble, technology becomes the centerpiece of the architectural space to an extent that the physical space starts to disappear into the potentials of the virtual space. This was within itself the beginning of an era in which the virtual took over the physical in order to satisfy the ambitions for a DreamSpace.

1.1.2. The 21st Century: The Disembodied Architecture

Today, in the 21st century, we have access to a world of connected computer networks that is "cyberspace"[12]; a term first introduced by science fiction to describe virtual environments. Amy Bruckman describes this virtual environment as one where "the local rules of social interactions are built, rather than received"[13], indicating that the conventional and inherited rules of a (physically-oriented) society do not apply to cyberspace. For people in a physically-oriented society who use technology to access a virtual world, there is a desire for spaces that correspond to the potential inherent in the virtual, spaces where everything is possible, identities can be reformed, and places can be accessed from anywhere in anytime. Such freedom from physical confines offers a new medium to project ideas and fantasies that cannot be represented, built, or supported in the physical world. This is a reason to why most people are disappointed with the limitations and conventions they encounter as they step back to reality. Sherry Turkle speaks of cyberspace's seduction of its so called "users" - turning them into "dwellers on the threshold between the real and the virtual"[14].

Treesense by Liu et. al.[15] explores the idea of reformed identities in a virtual environment. In this project, the virtual environment has freed the human user from reality to be a rising seed, sprout branches, grow to a full size tree, and be violently chopped down. While being a tree in reality might sound absurd, this tactile virtual reality system allows the users to explore "the bodily experience of being a tree"[16]. This experience has been aimed to correct people's behavior towards the natural environment by empathizing with them [Figure 3A].

In a similar project, PsychicVR by Benavides et. al.[17] is an immersive environment that allows the users to practice mindfulness. By using virtual reality head sets and sensing real time brain activity, this project tries to give life to the fantasy of having superhero powers in a world similar to our own. The appeal of this project is the possibility of doing anything and being anything in a virtual world that is not offered in reality [Figure 3B].



[Figure 3A] Left- Project Treesense, Xin Liu, Yedan Qian, Pattie Maes, "*Treesense: Eliciting Body Ownership Illusion in Virtual Reality with Electrical Muscle Stimulation*". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/465

[Figure 3B] Right- Project PSYCHICVR, Xavier Benavides, Judith Amores, Pattie Maes, "Psychicvr: Increasing Mindfulness by using Virtual Reality and Brain Computer Interfaces". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/425

While these projects and other similar ones take the user to a nonexistent virtual world, recently there has been a focus on combining the virtual with the physical environment. Augmented reality technologies such as the HoloLens[18] have enabled adding or manipulating what the users perceive from the Physical world [Figure 4]. However, such manipulations are bounded to the virtual environment and are not represented in the physical environment. Such disconnection from the physical world could result in alienation from reality; a phenomenon that happened in a similar but different context back in the 1960's.

[Figure 4] Hololens. https://www.microsoft.com/en-us/hololens

1.1.3. (Re)Activating Architecture through Computational Devices

Today technological advancements in the realm of computation provide us with the ability to manipulate physical matter. The disconnection between the Dreamspace and reality can now be approached using computational devices such as programmable actuators, open source software, sensors, and so on. It is the purpose of this research to develop an architecture – in the form of a physical prototype - that would understand the behavior of a user, and ultimately respond to their psychological desires through movement is space[19]. After defining the social and cultural purposes for the design, this prototype was developed to augment physical space, and react to people's movement on a horizontal spatial plane in a public context; a playful, evocative, and digitally enhanced floor that would act as a context for bodily expression and non-verbal communication. Bodily expressions, known as a means to non-verbal communication, are a user specific behavior. Depending on the cultural background of the user, (their sex, age, and their personal background) each person (body) will have a specific comfort zone. Hence it is important to design spaces – ephemeral spaces in particular – in accordance to the user's definition of personal and social space. Personal space as an invisible boundary can then be "constructed" – enforced or relaxed - in an ephemeral structure through architecture.

In the next chapter, the approach towards designing actuated spaces has been explained, and the literature has been reviewed. Subsequently, in chapter three, the methodology has been broken down into a design logic and material explorations. In chapter four, based on the methodology provided in chapter three, an actuated floor is selected as a suitable prototype that could sense and appropriately respond to a user's horizontal positioning in space. Consequently, chapter five would be the documentation and evaluation of the design's performance.

2. ACTUATED ENVIRONMENTS

> "Responsive, sometimes called adaptable, or reactive, means the environment is taking an active role, initiating to a greater or lesser degree changes as a result and function of complex or simple computation"[20]
>
> Negroponte, 1975

In his book "The Architecture Machine" Nicholas Negroponte (1975) defines the term responsive architecture as one that changes in response to a computationally defined command or stimuli. This stimulus can be triggered by an environmental variable, or human factors, or even both. Depending on the input computer command (stimuli) and the output (change), responsive design has been divided into sub-categories. Today a rich literature on responsive design and its categories exist[21]. In this section I will take a look at the typologies that exist in this area. However, this chapter is not meant to define those categories, but rather elaborate on their impact on this research.

The purpose of this chapter is to propose 'Interactive Architecture' as a solution to designing the Dreamspace in oppose to 'Responsive Architecture'. The definition of responsive architecture has condoned human behavior in relation to space and architecture. Therefore, interactive architecture has become an alternative that states human factors as the focal point in the design of reactive spaces. However, the contextual studies and the societal desires for a Dreamspace indicates that there still remains space for a new approach in interactive architecture; an evocative, activated architecture that is culturally responsible for the desire of shared, transformable, expandable, and limitless spaces.

2.1. TYPOLOGIES

In their book "Interactive Architecture", Fox et al.[22] categorize transformable spaces into those that respond to the pragmatic conditions and those that are triggered by the humanistic conditions. Based on these two categories there is a definition of "Responsive Architecture" versus "Interactive Architecture" respectively. While the first is concerned with the design of spaces that transform (adapt, react, etc.) in response to the physical and environmental variables, the latter is concerned with the design of spaces that physically and psychologically affect their inhabitants through their transformation. Both these categories are discussed in more details later on.

As to fully understand the implications of these categories, another complementary research on the typologies of responsive environments is brought in. A research done by Oh et al.[23] indicates Reactive, Conversational, and Social typologies for responsive environments. These three typologies can better explain the Pragmatic and Humanistic application of responsive environments. Using these terminologies and categories, I will initially clarify what is meant by "Actuated Environments" in this thesis. Consequently, examples of recent work in this field are analyzed and a comprehensive literature on "Interactive Environments" is provided. Later in the chapter, there will be a discussion on where this research stands in regards to the literature. Finally, the hypothesis and research contributions will be explained and clarified.

RESPONSIVE ENVIRONMENTS

HUMANISTIC APPROACH

[Interactive Architecture]

An architecture that actuates and transforms based on the physical and psychological desires of it's inhabitants.

Conversational	Social	Reconfigurable
Datagrove Does not facilitate the intimacy between the body and space as a transformable object.	*DIY City* People only make a temporal impact on the city image.	*Reconfigurable House* Inhabitants adjust their living conditions rather than the conditions adjusting themselves for the inhabitant.
Marling Individual impact on a communal space is merely visual.	*Boundary Functions* Individuals do not impact the physical space.	*Bubbles* As a multi-user interactive installation, it does not address social and spatial intimacy in its design.
		Dynamic Terrains The activated landscape does not include personal territories in a multi-user platform.
		Lift Bit An intermediary device has been added between the user commands and the spatial configurations.

PRAGMATIC APPLICATIONS

[Reactive/ Adaptive Architecture]

An architecture that learns from the environmental changes and adjusts itself to the needs of their inhabitants.

Islamic Museum of Art, Media ICT, Pneumatic Envelope, Dynamic Tower

EPHEMERAL SPACES

Interaction inherent in materials behavior and the relation between materials, space, and bodies.

[TABLE 1] The Typologies of Responsive Environments. (Author, 2017)

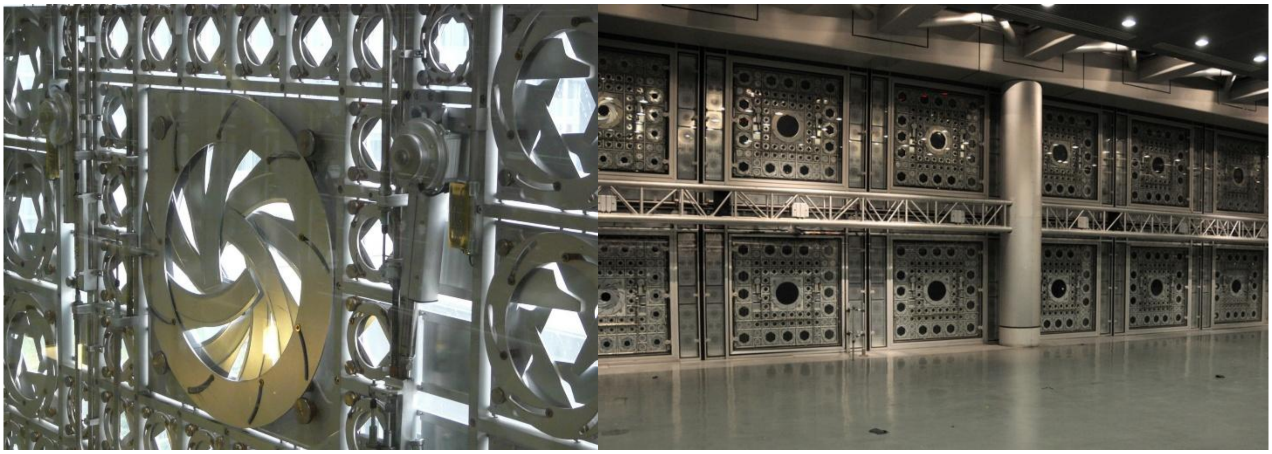
2.1.1. Pragmatic Applications

Primarily focused on solving needs and optimized solutions, pragmatic applications of responsive environments are those that utilize safety, security, shelter, space efficiency, transportation, and so on in addition to the economic aspects of design and architecture[24]. 'Adaptive' is a term used to describe an architecture that learns from the environmental changes and adjusts itself to the needs of their inhabitants. This typology of responsive environments is similar to what Oh et al. describe as Reactive[25].

A. Reactive/ Adaptive Architecture

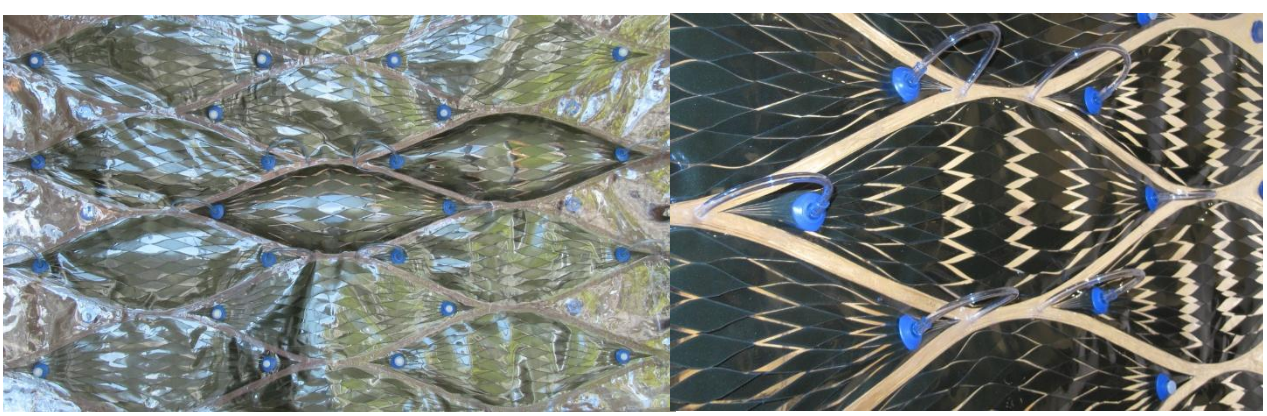
"*Reactive* refers to projects whose responsiveness is designed to have one-way information flow that has a clear distinction between input and output part. It is a problem-solving approach and is goal-oriented to achieve instant environmental modification"[26]. Such reactive environments are aimed as those that control environmental conditions in favor of the inhabitant's comfort. However, adding to this notion, stands "sustainable" architecture; designing reactive buildings that do not disturb natural resources and adapt to the changes in use and function of space.

The diaphragms in the facade of Jean Nouvel's Islamic Museum of Art [Figure5], the ETFE façade of the Media-ICT building [Figure 6], the Pneumatic Envelope responding to climate conditions [Figure 7], and the Dynamic Architecture in Dubai by David Fischer [Figure 8], are examples of reactive environments that are merely focused on the function of space and human comfort. Even though such designs consequently lead to an unprecedented experience of space, they are not necessarily predicted, nor the initial aim of the design. In other words, "there is no consideration of the behavior of the whole human-environment system. Therefore, the behavior of the system is open."[27] Even though such open systems provide the inhabitants with different spatial experiences-through the combination of environmental conditions and the architectural elements-they are mainly imposed from the environment and the designer. Disregarding the right of participation, the human factor is not accounted as a deciding factor in the spatial transformation. What will be discussed next is how design and architecture can be a conversation maker between the inhabitants and the desired environmental needs.



[FIGURE 5] Jean Nouvel's Islamic Museum of Art,
http://www.archdaily.com/162101/ad-classics-institut-du-monde-arabe-jean-nouvel

[FIGURE 6] Media ICT by Enric Ruiz-Geli**,**
http://www.archdaily.com/49150/media-tic-enric-ruiz-geli



[Figure 7] Pneumatic Envelope by Rick Sole,
http://rad.daniels.utoronto.ca/poly-glazed-curtain-wall/

[Figure 8]
Dynamic Architecture by David Fisher,
http://www.dynamicarchitecture.net/

2.1.2. Humanistic Applications

The humanistic application of responsive environments affects the inhabitant both physically and psychologically. This approach in architecture leans on the theories behind embodied interaction, arguing that bodily movements in space communicate different emotions and psychological information. Humanistic applications in responsive environments are then differentiated as interactive architecture. The role of interactive architecture is to actuate and transform based on the physical and psychological desires of it's inhabitants. Such physical transformations could be choreographed with the bodily movements in space as a medium for emotional and psychological expressions. This approach could be seen as one that develops a conversation between the inhabitant and the environment on one hand, or one that extends this relationship to a social context and between different bodies; as it promotes social intimacy between the users of space by understanding the cultural paradigms of bodily behavior in space. Next, the typologies of interactive architecture have been discussed as existing methods for designing transformable and activated spaces that respond to the behaviors of their inhabitants.

A. Conversational

The projects that emphasize feedback and communication between the human user and the environment are categorized as conversational interactive/responsive environments. Conversational environments are usually known as a coupled system and are not determined by the individual behaviors of each parts or actors. "The key point of these coupled systems is the whole effect"[28]. By developing a conversation between the entire system and the collection of actors, an unprecedented experience and understanding of space occurs which could not have been possible without the entire group of actors. To clarify this aspect of humanistic interaction, their actors, elements and conversation strategies, a few examples are discussed as below.

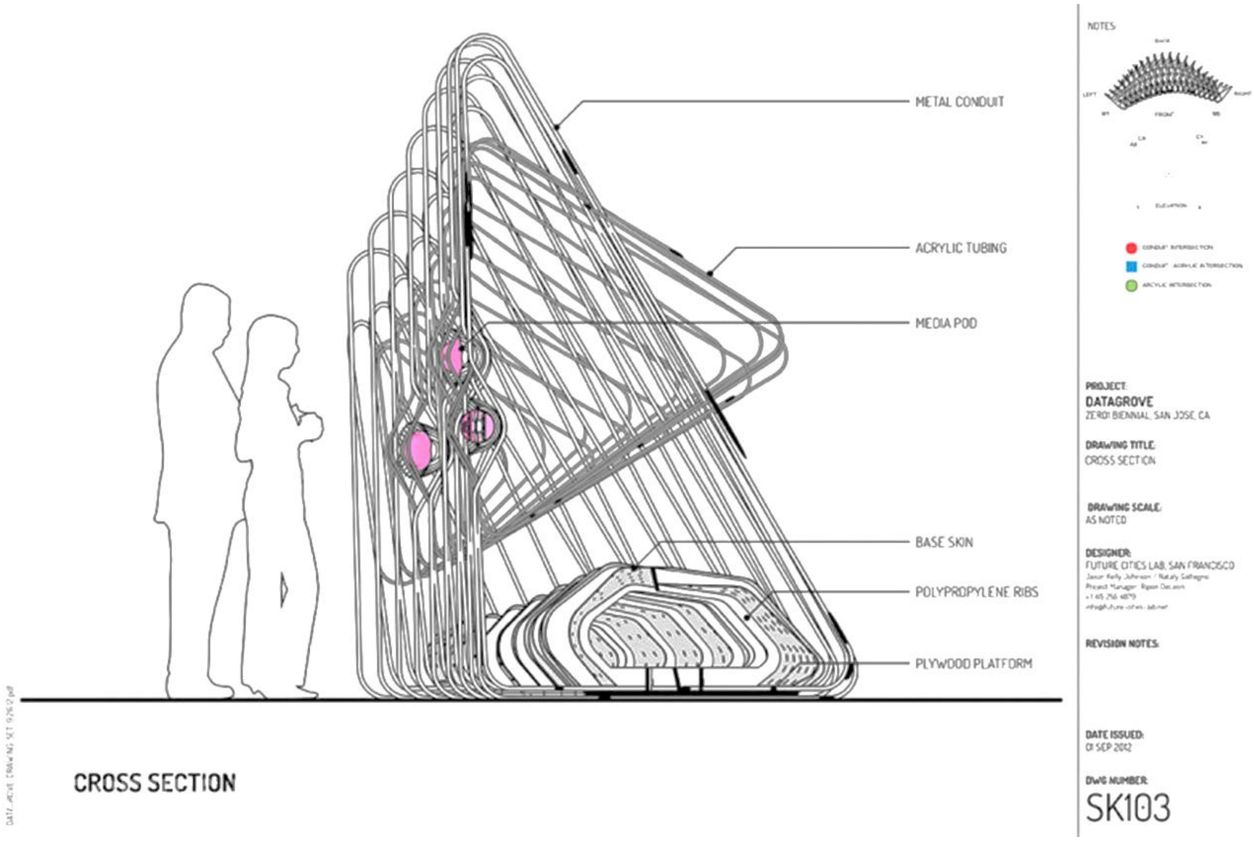
A.1. DATAGROVE

"The Datagrove thrives on information from its urban environment."[29]

Designed for the ZERO1 Silicon Valley's Biennale, Datagrove was an installation that gave ambience to invisible data and "atmospheric phenomena" through intensities of light and sound [Figure 9]. The installation responds to the proximity of the visitor's body to itself via whispers of sound. The sound being the voice of a computer generated persona, translates the local trends of Twitter feeds into speech.

Datagrove can be recognized as a conversational responsive project as it communicates invisible data from the actors to the actors themselves. The actors being those who use Twitter, become aware of this collective phenomena through experiencing this installation. In other words, the community of actors shape the responsive environment through light, color, and sound. This installation is an interface for communicating to an audience while it does not facilitate the intimacy between the body and space as a transformable object. It is important to note the intimacy between the body and space and its effects on social and individual behavior.

[Figure 9] Datagrove by Future Cities Lab, Each individual is interacting with a single Mediapod.
http://www.future-cities-lab.net/datagrove



[FIGURE 10] Datagrove elements and technology.
http://www.future-cities-lab.net/datagrove

A.2. MARLING

"Marling is a mass-participation interactive urban spectacle, sited in a public square in Eindhoven, Netherlands, brought to life by the voices of the public."[30]

Designed by Usman Haque et al. [31] Marling gives form to the voices of citizens. Through a colorful effect that hangs in the air above the crowd, people give life to an urban plaza through their actions and sounds [Figure 11]. Making people to believe in their collective effect in their environment, the virtual ambience changes as the people move and shout the way they desire. While redirecting the attentions to the sky, the horizontal space on top of people's head is remembered as a collective space definer. Marling is a successful example of how audio/ visual technology can create temporary effects that represent the individual's impact on space.

[Figure 11] Marling by Usman Haque. (http://www.haque.co.uk/marling.php)

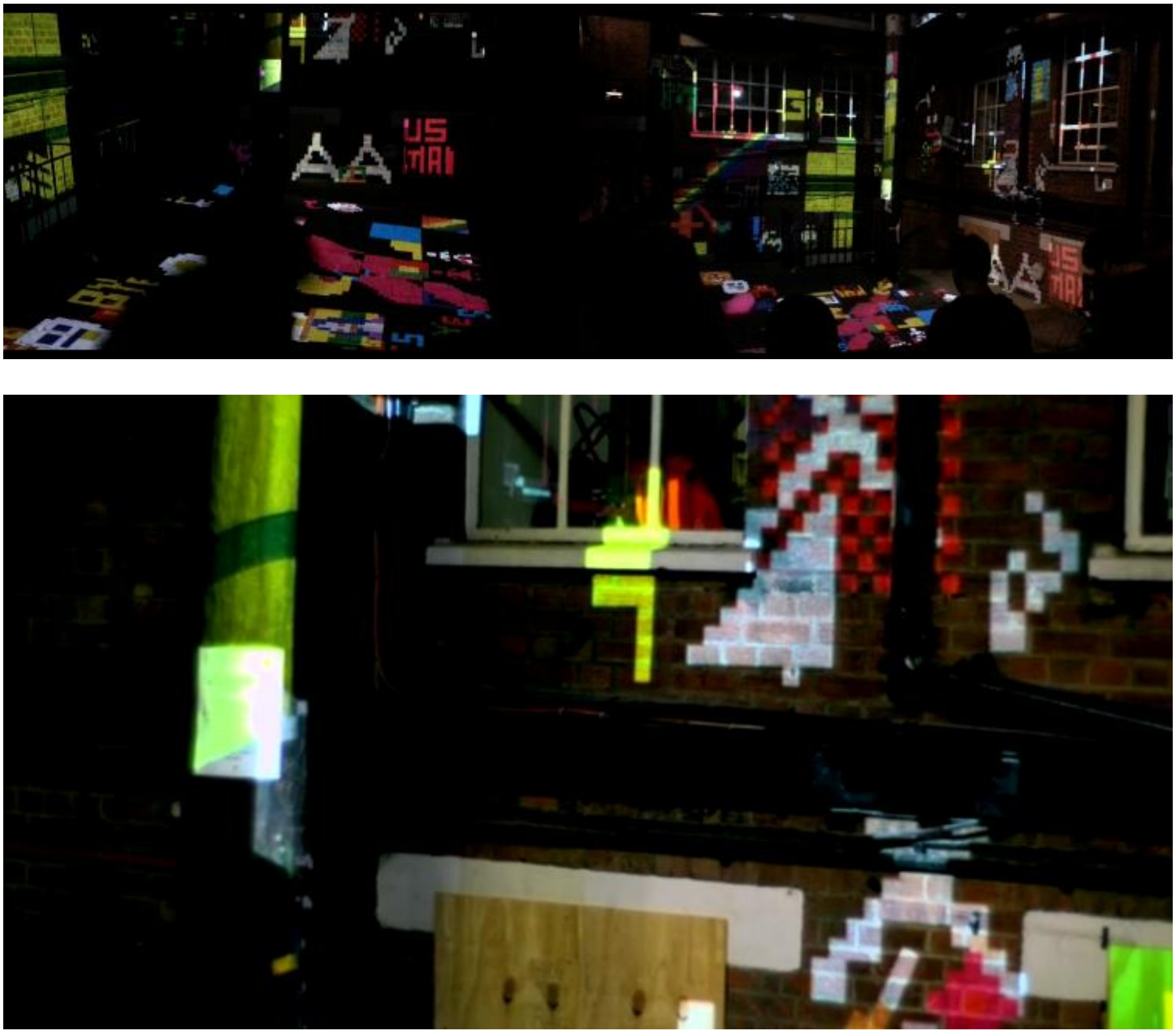
B. Social

"Social is defined as the gray zone between reactive and conversation."[32]

Projects in the territory of social interaction, involve multiple users who not only interact with the designated interface, but with each other. In these types of interaction, there is an act of participation that impacts the outcome of the of interactive space. Not only it is a matter of reaction to the acts of a specific user, but also it creates a conversation between the participants. The context of such interactions can be varied. The "DIY city" by Usman Haque[33], and "Boundary Functions" by Scott Snibbe[34], are some of many examples that fit in this category and represent participation and reaction in two different scenarios.

B.1. DIY CITY

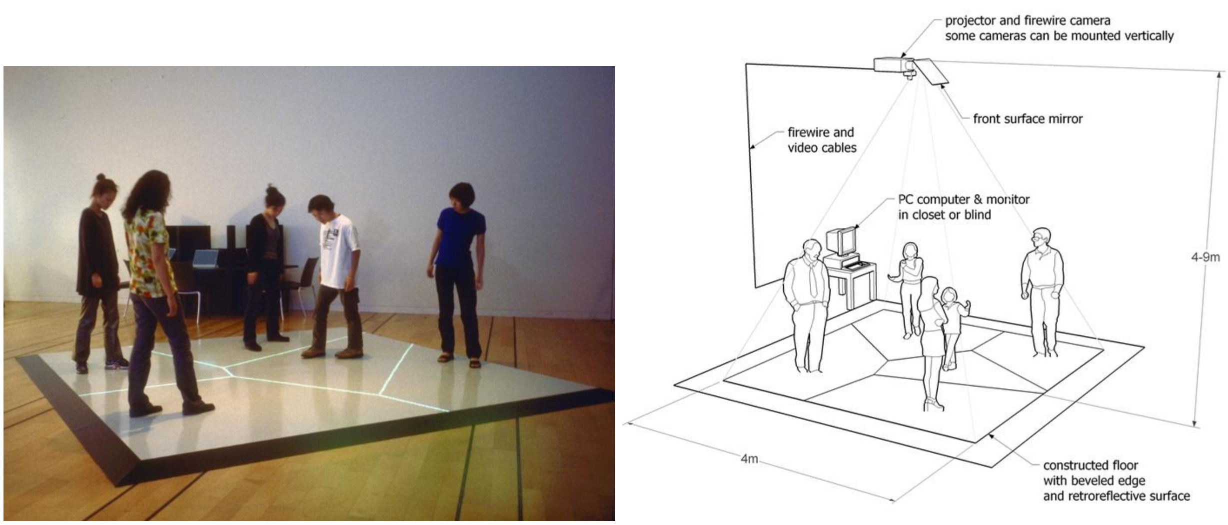
The DIY City 0.01a by Haque design and research group the streets become a social canvas for three dimensional projections. The citizens of the city can become active members to paint over the existing architecture in favor of their individual desires in the context of a social conversation in their neighborhood. While this re-imagination of the city is an ideal approach in urban design, this project is merely limited to a temporary image of the city.



[Figure 12] DIY City by Usman Haque for Special Movies group.
URL: http://www.haque.co.uk/diycity.php

B.2. BOUNDARY FUNCTIONS

Boundary Functions is an example of a social interaction that would not function with a single person. Challenging the idea of personal space, this installation tries to depict the line between one person and another in the physical world. While these lines disappear in the virtual space, the existence of a virtual network is based on the existence of a community of users. “In this way *Boundary Functions* is a reversal of the lonely self-reflection of virtual reality, or the frustration of virtual communities: here is a virtual space that can only exist with more than one person, in physical space.”[35] This conflict between the individual and the society has become a point of conversation in this work. It would be interesting to note such conflicts in the design of the physical space. There is a need for an individual’s discrete impact on space besides a community’s influence.



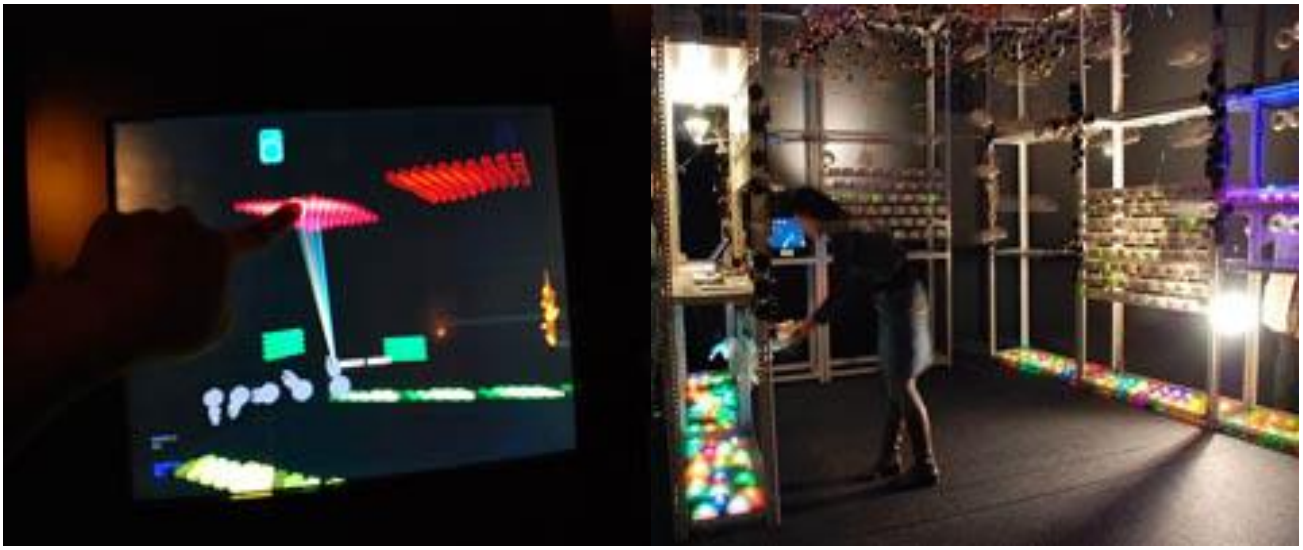
[Figure 13] Scott Snibbe, Boundary Function, 1998, Accessed May 2017. URL: https://www.snibbe.com/projects/interactive/boundaryfunctions/

C. Reconfigurable Spaces

In this section, all of the discussed typologies come hand to hand to present a typology that allows users to manipulate the physical space in a social context. Focusing on behavioral aspects of human motion in space, the architecture is enabled to react to the physical, psychological, and perceptual needs of its users. In different cases the human body plays a different role in the design outcome. In some projects the experience is a result of a conscience decision from the user. While in others, spatial experience is seamlessly interconnected to the user's movement in space. However most of these projects lack the factor of intimacy with space. Today the technology exists to create spaces that intuitively respond the desires of their inhabitants in a transparent way.

C.1. RECONFIGURABLE HOUSE

The reconfigurable house has been intended to give the freedom of decision making to the inhabitant by taking a DIY approach. The inhabitants have the chance the add as many sensors they like to the structure of the house as a counter act movement against smart houses that are unable to adapt structurally over time. However, such transformable spaces add another layer to the decision making process for the users and add to the complexities of the living conditions. It would be ideal to develop strategies and use computational tools that could adjust themselves overtime through understanding and learning the human needs and desires in oppose to adding to the physical complexities of space.



[Figure 14] The Reconfigurable House by Haque Design + Research
URL: http://www.haque.co.uk/reconfigurablehouse.php

C.2. BUBBLES

Bubbles is a pneumatic installation that responds to the presence of people in the urban scale. As visitors enter the installation area, giant inflatable sacks start to inflate and deflate above their heads. The individual visitor then has an impact on the overall experience of the space. However, this impact does not happen through direct interaction with the human body. The work is merely limited to detecting presence and the visitor become s spectacle to the shape-shifting space.

[Figure 15] Buubles by FoxIn. URL: http://foxlin.com/portfolio_item/bubbles/

C.3. DYNAMIC TERRAIN

Dynamic terrain is an interesting reference to the work that is envisioned in this thesis. This project tries to connect the digital to the physical space. The reactive platform of this project acts closely to the body and the users can control its behavior through touch sensors and a mouse. Even though the intentions of the designers were to design spaces to respond to the user's actions, this project merely follows the user's commands. This project opens up so many possibilities in terms of designing ephemeral- or as the designer puts it, virtual- spaces. One could be expanding the design as a multi-user platform in a social scenario. As we add users to an existing space, the issues of personal space, comfort and so on emerge that has not been investigated in this project.

[Figure 16] Dynamic Terrain by Janis Ponisch. URL: http://www.interactivearchitecture.org/dynamic-terrain-janis-ponisch.html

C.4. LIFT BIT

Designed by Carlo Ratti, Lift Bit is a modular furniture that could be adjusted according to individuals' desires. Each user is free to choose their desired position through a tablet app. The modules then start transforming to the desired position. Again in this project, we see the idea of implementing the user's process of decision making through commands; in other word this space favors the idea of control over spaces that react intuitively. The computational devices today can help designers eliminate the existing intermediaries for the sake of seamless interaction between the body and space.

[Figure 17] Lift Bit by Carlo Ratti Associatti, URL: http://lift-bit.com/

D. Ephemeral Spaces

Souza in his paper titled "Ephemeral Spaces"[36] categorized ephemeral spaces into different types. The one type categorized as 'Artworks and Installations' talks of structures that are "inflatable, or multi-cellular structures which utilize space, color, texture and sound as a means of communication with individuals who traverse the structure." An example of such work is done by Maurice Agis. Confronting the audience with an evocative approach towards culture in the city, he describes movement as a way to connect to the self, others, and indeed the space itself. As he describes his project Dreamspace [Figure 18] with Stephan Montague he shows us what ephemeral means in relation to the human body, to space, and to others.

"…. It is this sort of temporary structure which does not, however, represent any particular government authority or commercial power base and is therefore both revolutionary and free. An event generated in the public space in which the city's inhabitants can become directly involved defuses, if only temporarily, the authority of the established system".[37]

(Robert Kronenburg, Portable Architecture, Page 239
In reference to Maurice Agis' Dreamspace Project)

[Figure 18] Interior of the Dreamspace installation by Maurice Agis.
https://hiveminer.com/Tags/agis,dreamspace/Interesting

Agis himself was part of the psychedelic movement and it is needless to say that his design descriptions are similar to the anti-authoritarian design principles which were discussed in previous chapter. Dreamspace is a space designed to respond to the bodily movements of its users. While ephemeral to him is the temporary manipulations of space by the users, the space is constant. Unless there is a bodily interaction with the spatial elements (here the tensioned structure), the space would not change. Invisible qualities of bodily interactions, such as personal space, have not been considered. Other defining variables such as gender, age, height, and so on have not been considered in how the space can shift and change shape. Today, considering different sensorial systems, designers have the opportunity to design spaces that respond to not only different bodily movements, but to both visible and tacit characteristics of the users. Visible such as gender, or tacit such as personal space, emotions, and so on. Spaces can shift beyond the physical boundaries of their users; personal space can extend to more than the body and individuals can impact what spaces can be beyond the physical limitations. Being aware of such limitations and opportunities in design, this research aimed to give life to a space that considers such parameters. A sensory space that goes beyond "smart" and challenges the user impact when using an architectural space.

DREAMSPACE		
EPHEMERAL	**ARCHITECTURE**	
TRANSFORMABLE	MOVEMENT	RESPONSIVE
ELASTIC	PERSONAL SPACE	UNPREDICTABLE
ACROBATIC	COMMUNAL LIFE	LIMITLESS
INFINITE	TACTILITY	NO BOUNDARY
TEMPORAL	PHYSICALITY	
PLAYFUL	FUNCTION	

[TABLE 2] Qualities and Design Intentions for the Ephemeral DreamSpace. (Author, 2017)

3. METHODOLOGY

Focusing on a ground plane interaction between archetypical sitting postures of the human body and space [Appendix B], the efforts for finding ways for developing ephemeral structures and spatial experiences began. Inflatable articulations as those that convey a feeling of playfulness, temporality, and pleasure was chosen as source for developing the evocative, actuated architecture. In addition, finding the right sensory system and incorporating it into the design added more challenges to the development of a prototype. This chapter will continue by discussing the challenges, possibilities, and the investigation for a probable design approach for an architectural element mimicking the variability and potentials of a Virtopian Space.

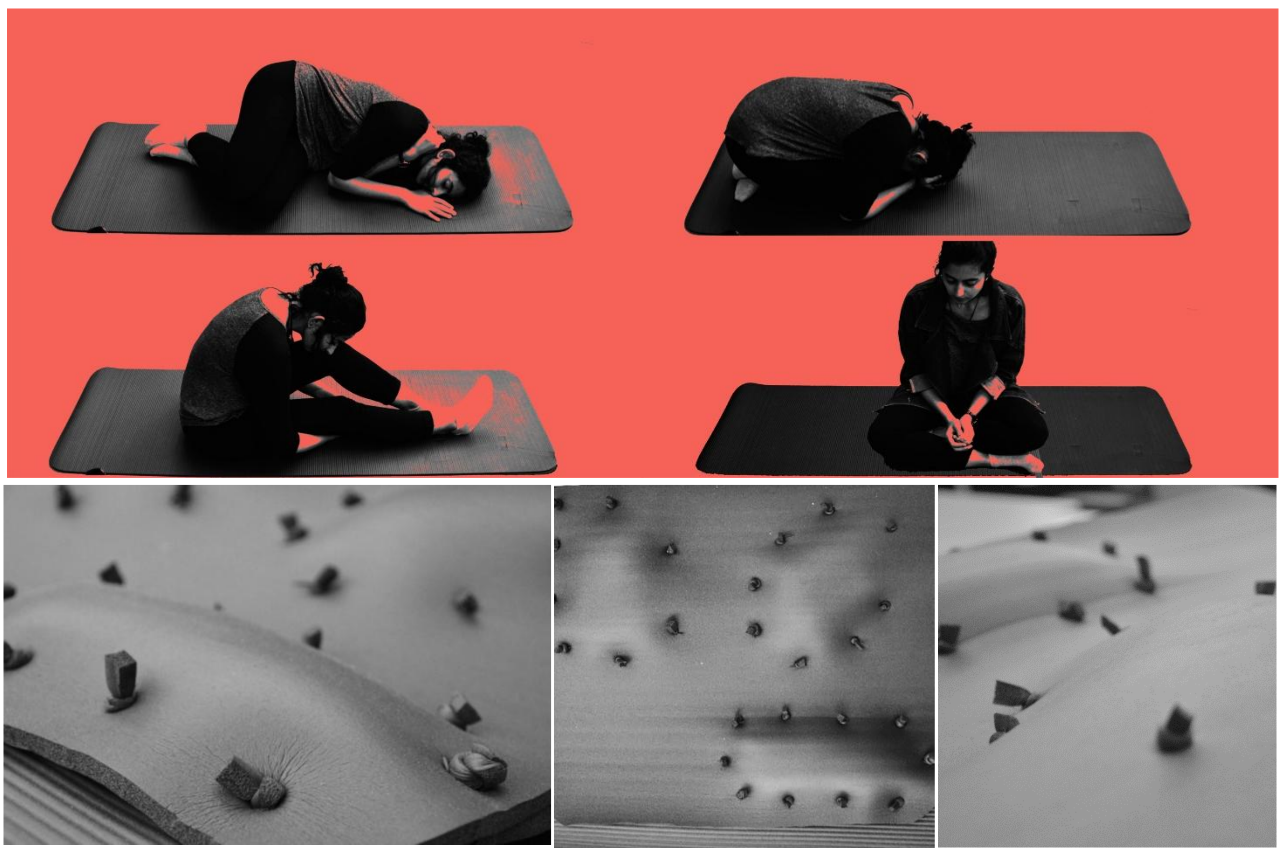
3.1. Design Explorations

This section of the research includes the trials, errors, and achievements that led to the making of the final interactive prototype. In order to develop a proof of concept, a true to scale prototype was built, a prototype of an ephemeral, flexible, and comfortable space that would motivate interaction in a space defined by the floor plane. Spaces that react to the user's will and personal comfort zones, spaces that appear and disappear due to the user's intentions, whereas inclined to the cultural postures and habits of the users. Challenges in this exploration included: detecting user's movement in space and defining a paradigm for defining personal space, and, finding a shape-shifting, ephemeral material that would act and react as an actuated structure that could respond to its user's actions.

3.2. Basic Prototyping

3.2.1. Considering Actuated Materials and Devices

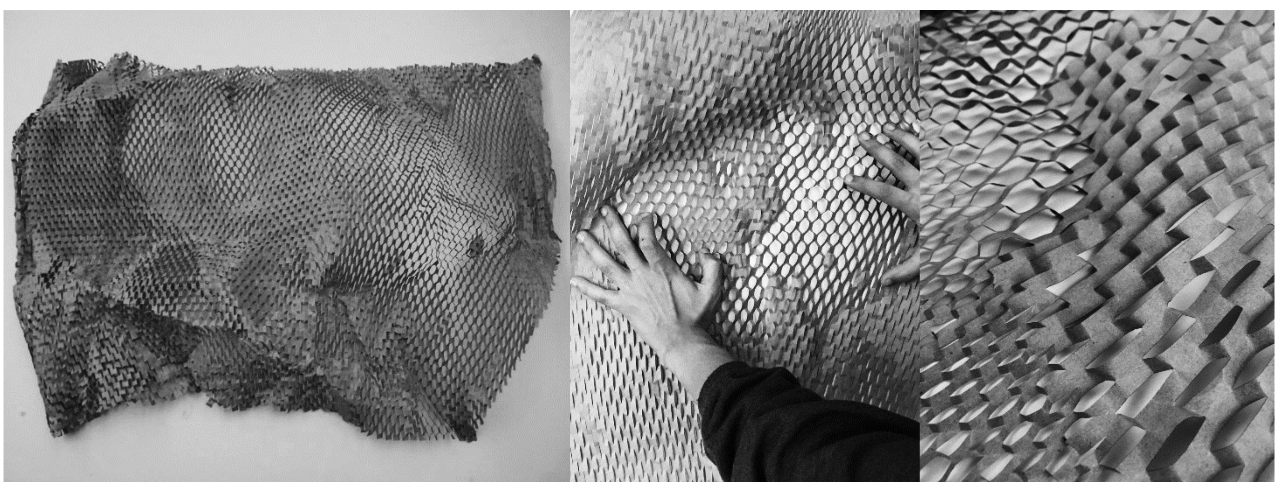
The initial step was to find appropriate materials that could physically act and react to stimuli. Simple prototypes were made to depiction what the final prototype could potentially look like. Using a commercially accessible material, a soft floor was envisioned that could embrace bodily interactions on itself in a genuine manner. Inspired by the bodily interactions of the yoga practitioners, a yoga mat was used to develop a rough concept sketch [Figure 19].



[Figure 19] Top: Bodily expressions on a yoga mat as a horizontal space definer. Bottom: Using double layer yoga matt filled with wool. (Author, 2016)

The initial model worked as a cushion with an inherent flexibility that would change shape by applying top-down pressure. However, this shape change was not digital in a sense to be computationally manipulated. A computationally augmented material meant manipulating the shape-change in order to respond to unexpected responses to human gestural movements. Hence forward, I tried to work with materials that could be integrated with computational devices such as sensors, conductive materials, and actuators.

One of the first developments was manipulating paper and cut art. Cutting paper, specifically with the laser cutter, in certain ways would change the paper from a two dimensional, flat material to one that can be shaped differently in the three dimensional space. This method could be a design solution to a flat/ horizontal/ 2-dimensional space definer, that changes shape to become a three dimensional form or pattern. Figure 20 shows the shape-change that was made using graph paper and laser cutter.



[Figure 20] Paper cut used to design a flat surface with a three dimensional shape-change. (Autho, 2016).

The transformation explored in the laser cut paper was promising. So one of the initial developments was an attempt to integrate paper with actuators such as servo motors or shape memory alloys. However, there are issues facing such actuation methods when it comes to design and architecture. Further on such limitation are discussed:

A. Limited Designs

In a comprehensive paper, Rasmussen et. al.[38] have done a literature review on shape changing interfaces. They clearly specify the fact that most of the research conducted in the field are not purposed for design but rather a test for technological and technical capabilities. Since technology is limiting, most design solutions are based on what the technology offers in oppose to designing technical systems as a solution to a design. In the project *Datagrove*[39] which was reviewed before, we can see how an audio-visual system was designed for a certain idea by using simple technologies. While projects such as *Sprout I/O*[40] [Figure 21] or *Shutters*[41] [Figure 22] have designed to include the abilities that Shape Memory Alloys offer. Consequently, such designs turn out similar due to their reliance on a technology, rather than developing new technologies and materials that are appropriated for the function, scenario, or aesthetics.

[Figure 21] Sprout I/O. MarceloCoelho, Pattie Maes, "Sprout I/O:A texturally rich interface.", Proceedings of the 2nd international conference on Tangible and embedded interaction. ACM, 2008, Pages 221-222. This project was developed around the capabilities of shape memory alloys.

[Figure 22] Shutters. MarceloCoelho, Pattie Maes, "Shutters: a permeable surface for environmental control and communication.", Proceedings of the 3rd international conference on Tangible and embedded interaction. ACM, 2009, Pages 13-18. This project was developed around the capabilities of shape memory alloys.

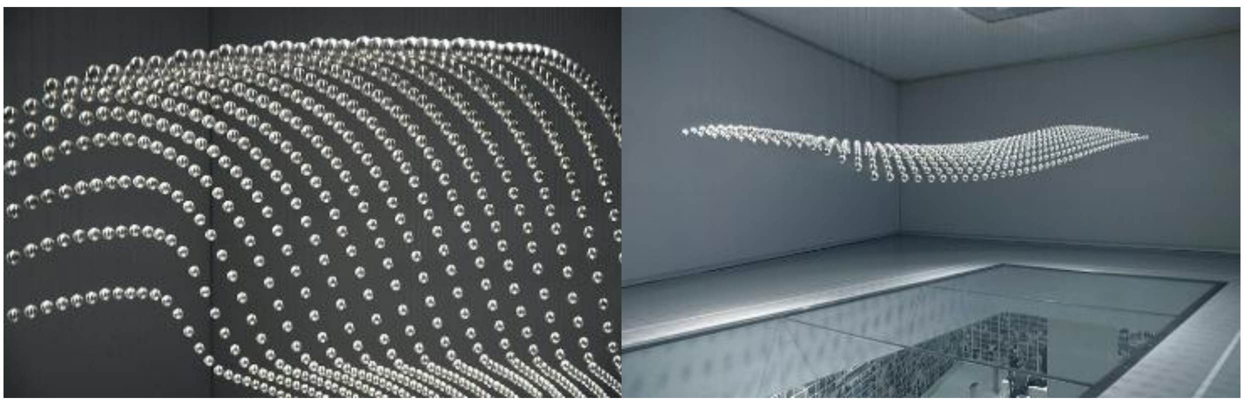
This research, however, is not aimed to deny the value of any of the two approaches in research and design; but choses to create a system that is appropriate for a context specific, social, and cultural design. While some designs limit the user's interaction with the shape-changing material, this research aims to go beyond such limitations and liberate the user to discover the interaction in a full range of body movement.

B. Expensive

The fact that some shape-changing materials are way too expensive limited my research content and their results. Materials such as shape memory alloys, piezoelectric materials are examples of which are used in small scale design prototypes and do not extend themselves into spatial or architectural design prototypes due to their costs. Therefore, such designs will not be tested in bigger scales until their production costs comes down. This is also true for other electrical actuators such as electromagnetic actuators, servos, and so on. Such actuators are capable of controlling each element of a design piece individually, only if each element is connected to an individual actuator; which is not financially efficient.

C. Complex Systems

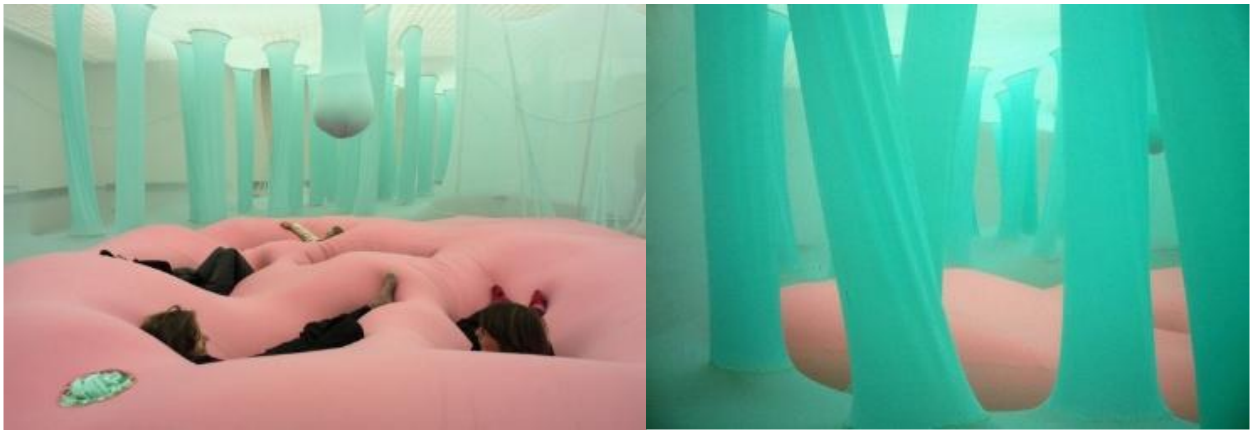
In addition, the complex systems that stems from such designs, prevents them from being accessible for further design developments by other users and designers. The Kinetic sculpture at the BMW museum is "a 6 square meter large installation in which 714 metal spheres are suspended from the ceiling on thin steel wires and animated with a help of mechanics, electronics and code."[42] As a good example for a complex system of design, this is a reason to why such designs have not become part of the everyday culture of architecture.



[Figure 23] Kinetic sculpture at the BMW Museum by Joachim Sauter.
http://www.joachimsauter.com/en/work/bmwkinetic.html

D. Unintegrated Actuation

Actuators that act as an added value to a system are considered as non-integrated parts to the material and the overall design. By attaching an external actuator to any material, the material becomes capable of shape change. However, it is not usually based on the inherent shape-changing qualities of a material. As seen in *Sprout I/O,* the SMA wire lifts the cut textile that is not part of the material but rather the design. The material itself becomes a complex system of actuators which makes it heavy, aesthetically unpleasant, and an expensive design. This is while using the original qualities of a material can reduce such obstacles. An example can be the use of stretch materials that is capable of responding to different pressures [Figure 24].



[Figure 24] Soft sculpture by Ernesto Neto.
https://thefantasticthingsifind.wordpress.com/2015/01/31/ernesto-neto/

As a conclusion, this research aimed to fill in such gaps in the design of shape-changing systems by following qualities such as:

- A material with inherent shape-changing qualities.
- An actuation method that could be easily scaled up.
- A cheap and accessible actuation.
- An easy to control, computational actuation.
- And, a life-like shape-changing feedback as an output to sensory input.

In addition to the design considerations listed above, there were additional qualities that were specific to the design to an actuated, interactive floor such as:

- Materials that could be used horizontally.
- A system that could support a sitting person's weight and would not be disabled by bodily interaction on it.
- Compatibility with sensor integration.
- A material that would provide a sense of physical comfort (soft, playful, flexible, and so on) for those who would like to sit on the interactive floor.

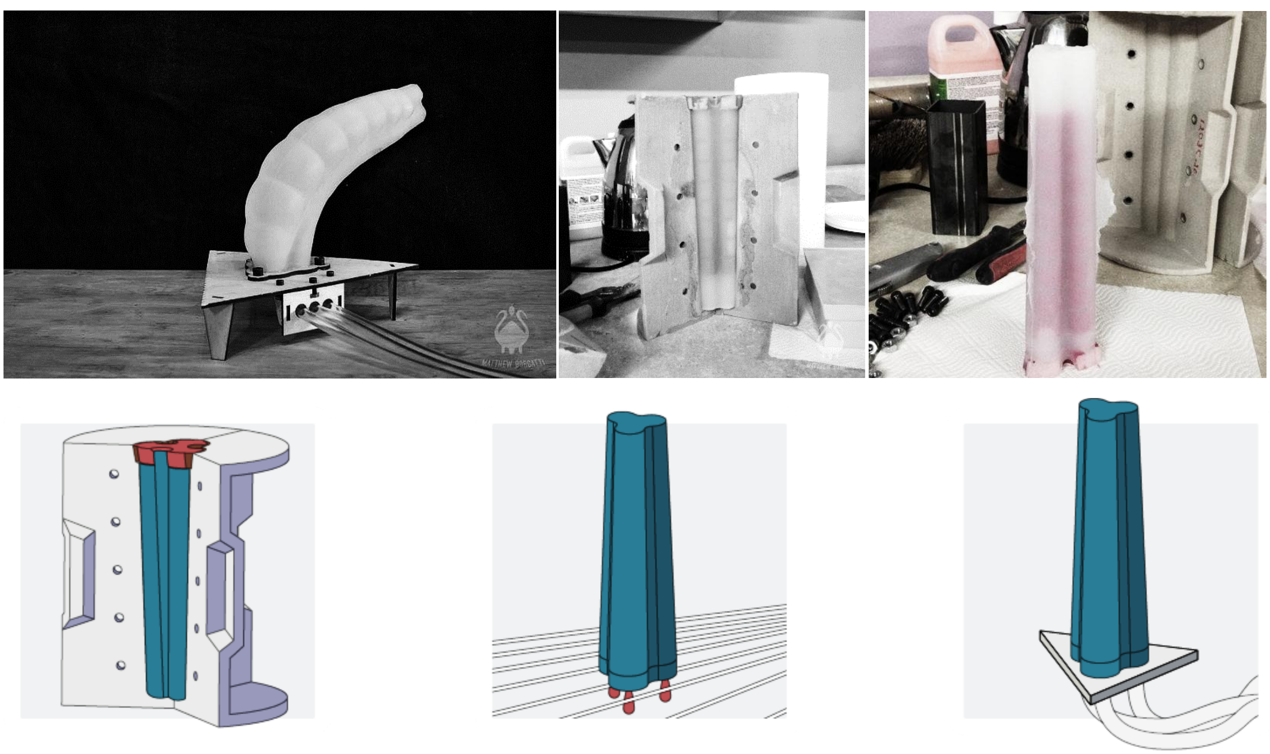
		QUALITIES				
		Raw Use of the Material*	Low Cost	Accessible / Low Tech	Scalable	COMPUTATIONALLY CONTROLLABLE
ACTUATION DEVICES	SHAPE MEMORY ALLOYS	x	x	x	x	✓
	SERVO MOTORS + LEVERS	x	x	✓	x	✓
	INFLATABLES	✓	✓	✓	✓	✓

*Raw use of the material: Some of the shape-changing actuation materials could not be used directly but are rather used as actuation devices, integrated with other materials; etc. SMA can be integrated with soft fabric and change the shape qualities of the fabric in different states.

[TABLE 3] Qualities regarding the decision making for actuation devices. (Author 2017)

3.3. Inflatable Actuators

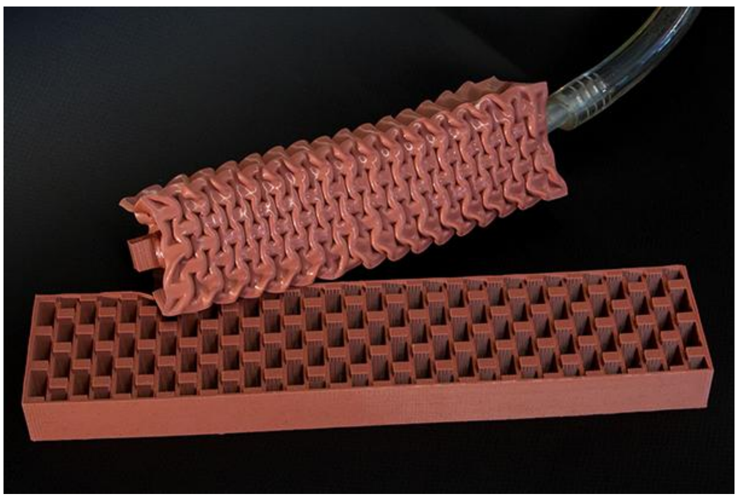
Tentacl[43] by Borgatti is an inflatable "soft" robot that is programmed to change shape by controlling an air valve activated by an Arduino kit. A soft plastic based material such as Silicon is used to make the soft robot that can be actuated through inflation, just as a "soft" robotic muscle [Figure 25]. Silicon, as a material that seamlessly works with its stretchable and inflatable quality, seemed promising for the design of ephemeral, bubble like structures. While such structures were once envisioned by the Soft Cities Lab back in the psychedelic era, they were not as successful. Today, such structures have been used in vast scales and in different shapes. However, using computational methods, inflatables can be used to represent spaces that change shape instantly due to the desire and the will of their users. Pursuing this approach for the design of the DreamSpace prototype, I tried to develop the Silicon inflatable bladder and understand how *Tentacl* functioned computationally.

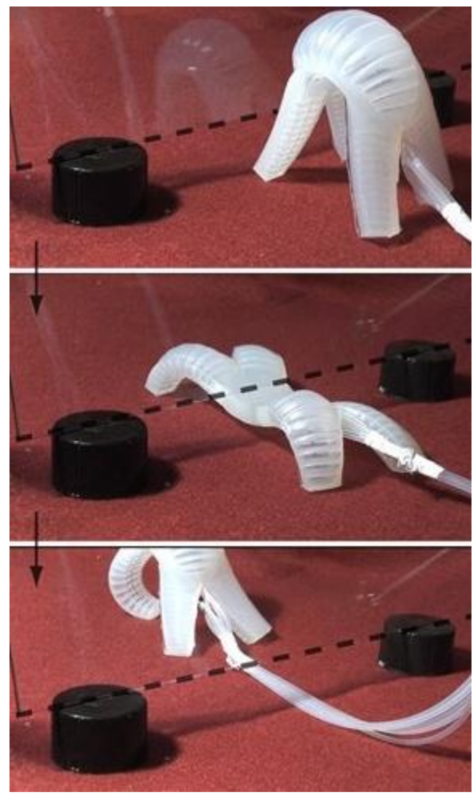


[Figure 25] Robo-Tentacl by Mathew Borgatti. (a) The final Tentacl (Robotic Muscle) (b) Casting the Silicon into 3d printed molds (c) Silicon released from the outer mold. (Adafruit, 2013)
URL: https://www.adafruit.com/

3.3.1. Flat Inflatables

Based on my research into "zero volume actuators" by Harvard's WYSS Institute[44], I was able to start making modified bladders that could only inflate in one direction whis is different than multi-dimensional inflation typical of balloons. Their work on "Actuators inspired by muscles" was published in June 2016, and discusses the use of a double-layer silicon bladder that could be inflated for shape-change and load bearing actuation [Figure 26].



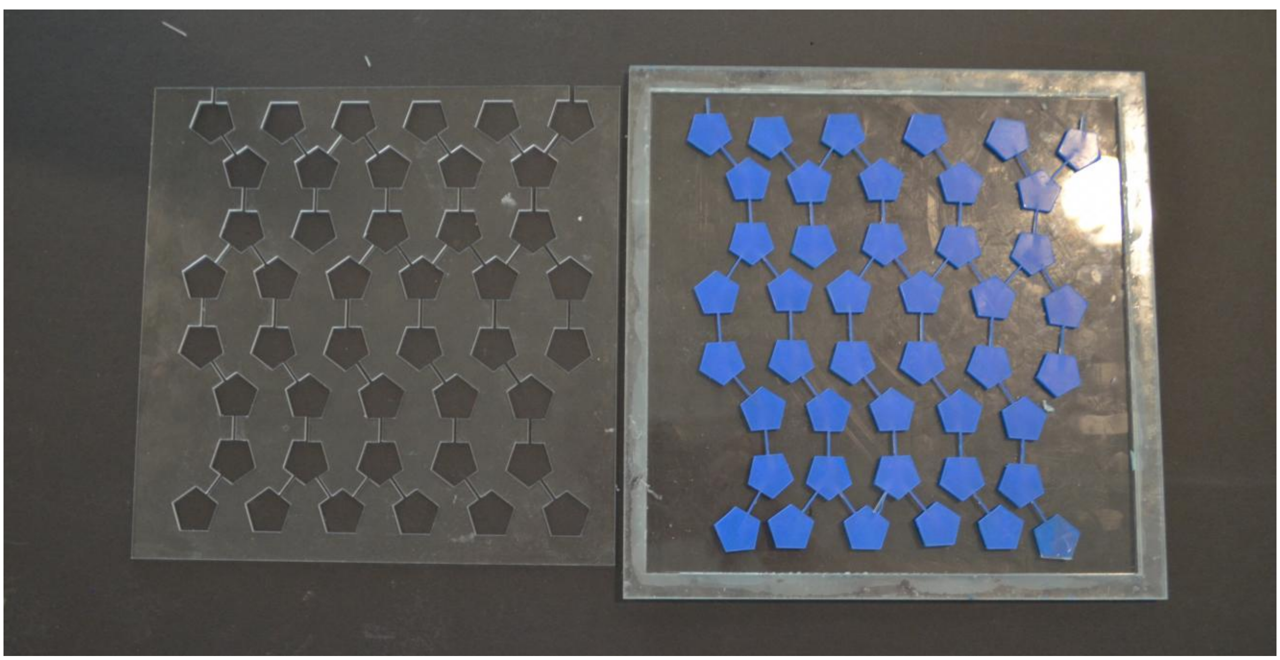


[FIGURE 26A] Top left: Actuators inspired by Muscles.
https://wyss.harvard.edu/actuators-inspired-by-muscle/

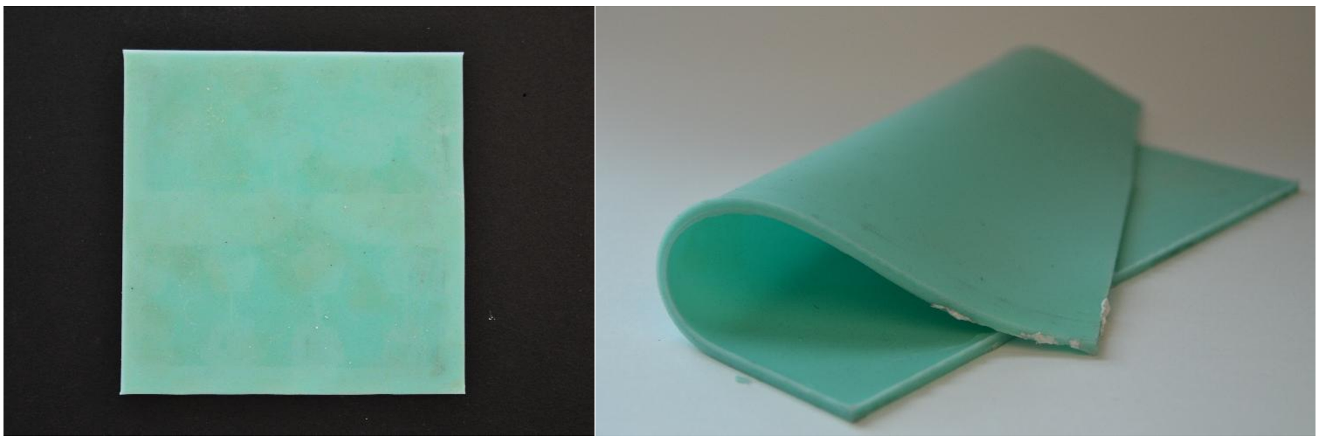
[FIGURE 26B] Right: Soft Robotics.
https://www.forbes.com/sites/jenniferhicks/2012/04/22/soft-robotics-takes-shape/#765b995e4008

Using a technique that was inspired by *XYZAidan*[45], a YouTube video developer, I used the *Smooth-on15* mold making silicon latex rubber and the Mann 200 mold release [Figure 27] to develop the first of my manually inflatable bladders [Figure 28].

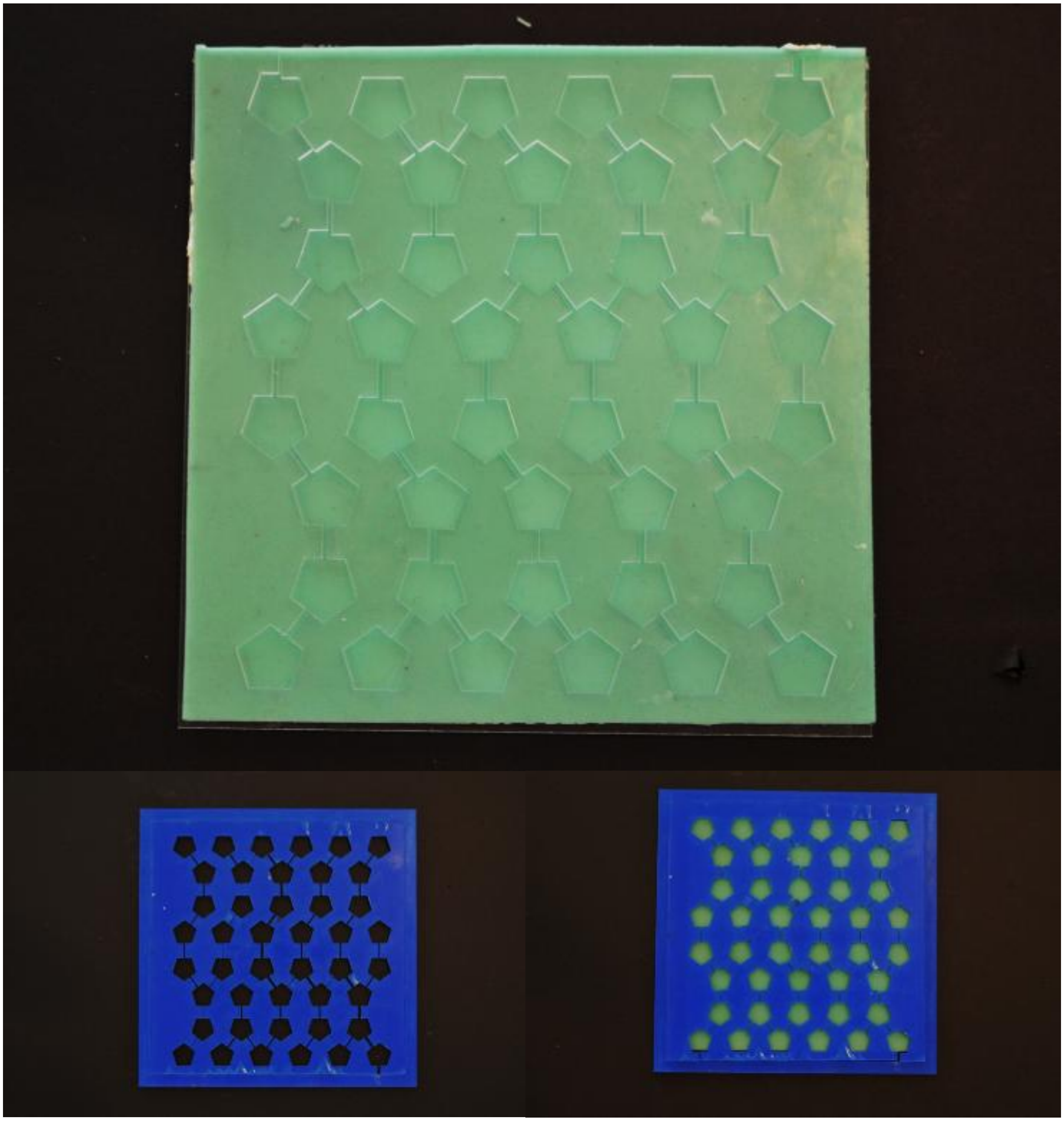
[Figure 27] Smooth-on silicon mix and Mann 200 ease release. (Author, 2017)


[Figure 28] The mold and the hexagonal shaped stencil. (Author, 2017)

First layer of silicon mix is poured in the mold. After four hours the silicon is dry and ready for the second layer of silicon. To prevent the two layers sticking together (as for the bladder) a stencil was used to spray mold release on the preferred areas. After five minutes, the second layer of silicon is poured onto the first. Again, after four hours, that the silicon has dried, the "flat inflatable" is ready to be inflated. Ideally, only the areas of separated silicon would inflate and change shape.



[Figure 29] The first layer of dried silicon is ready to be sprayed. (Author, 2017)



[Figure 30] The final silicon bladder. It is a purely flat and flexible surface. (Author, 2017)

[Figure 31] Inflatable prototype. Left: the double layer silicon that has not yet been inflated. Right: Channeling air through the casted silicon for inflation and shape-change. (Author, 2017)

Problems:

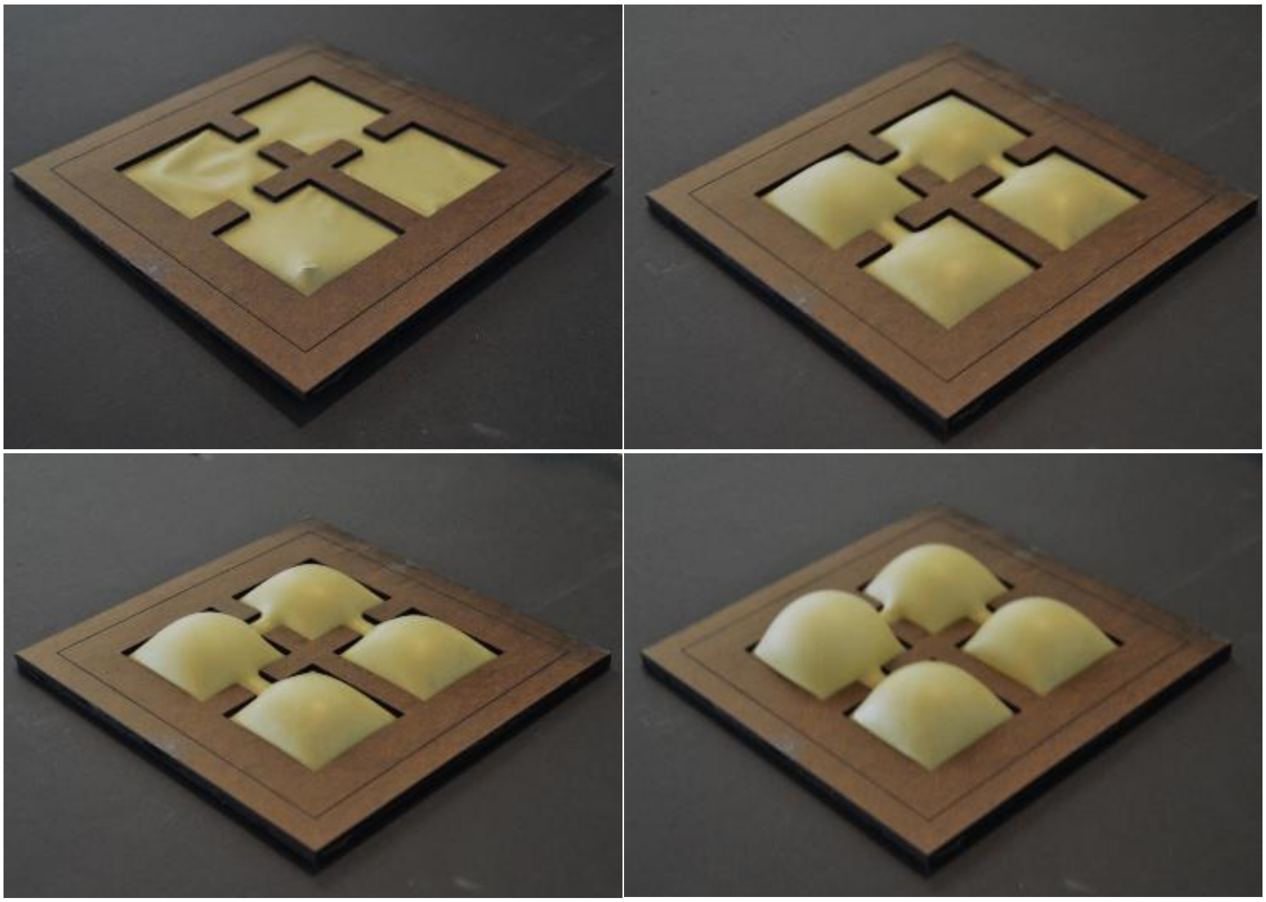
Due to a spray leak while applying the mold release agent, the hexagonal bladders became misshaped and resulted in what could be seen in figure 34. Also the second layer of silicon was too thin, which made it sensitive to the air pressure – holes and rips appeared in the prototype. Even though the results demonstrated the potential of silicon as a soft and moldable material, there were a couple of reasons not to develop the work with silicon. For making a true to scale prototype a large amount of silicon mix would be needed. This meant a high price for making the final prototype in the desired scale. In addition, silicon is not a natural material. The waste that is produced from silicon will not be recycled, nor decomposed in nature. Hence my adviser and I tried to find substitute ways of obtaining similar results, but with other materials that were less expensive, more accessible, less time consuming, and more natural in terms of recyclability.

Latex rubber sheets, which are elastic sheets made out of natural materials, was determined as a substitution for silicon. However, latex rubber bladders (balloons) would have had a sphere shaped inflation that was not aligned with the original idea of flat inflatables. A solution to this problem was to compress the latex rubber sheet between two hard and fairly strong layers to contain the air, and to define different shapes for the inflatable latex rubber on a flat surface. The first prototypes were successful and I decided to pursue this approach for the final version of the DreamSpace Prototype [Figure 32].

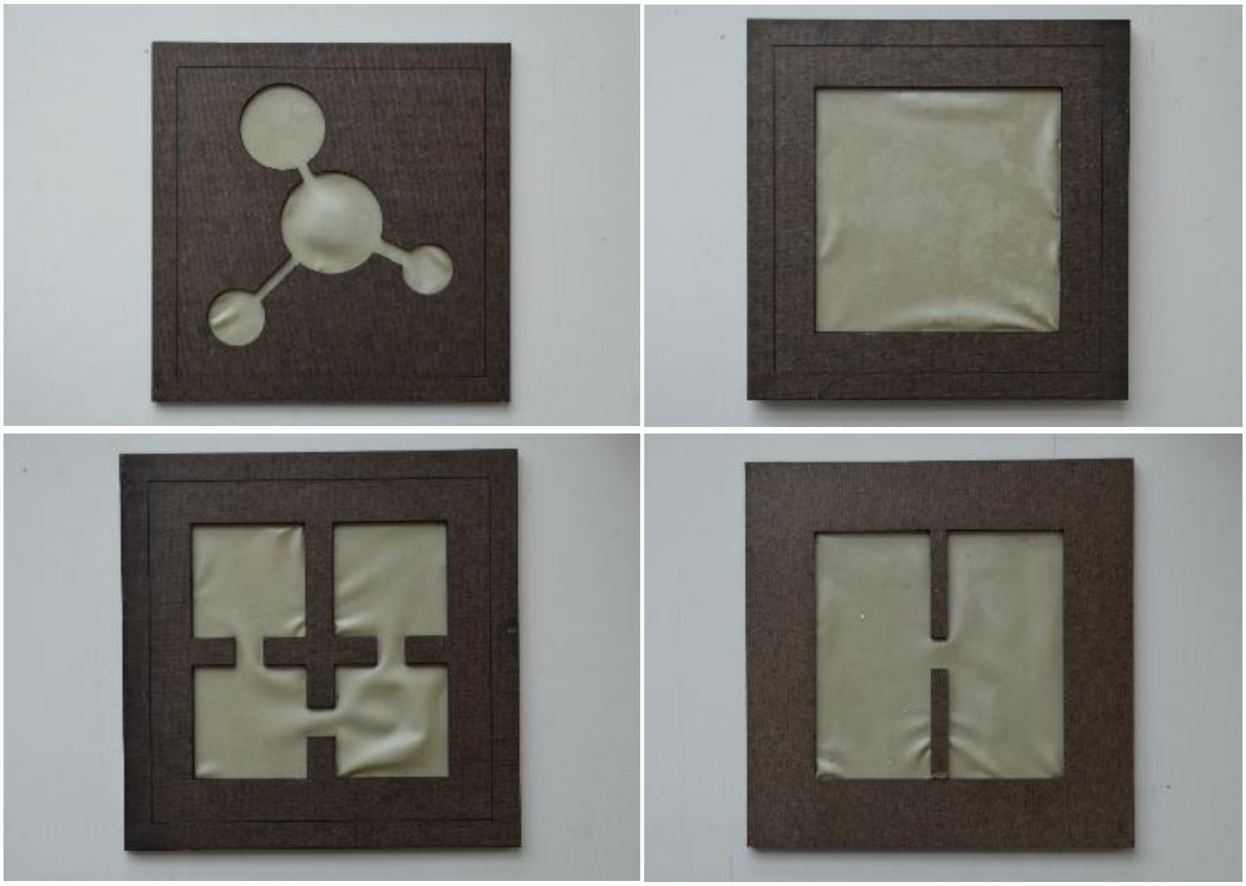
MATRIALS		QUALITIES				
		FLAT INFLATABLE	MOLDABLE	SELF-STANDING BLADDER	COST EFFICIENT	TIME CONSUMING PROCESS FOR MAKING
	Latex Rubber	X	X	X	✓	X
	Silicon	✓	✓	✓	X	✓

MATRIALS		RECYCABLE	ACCESSIBLE
	Latex Rubber	✓	✓
	Silicon	X	X

[TABLE 4] Table indicating differences between Silicon and Latex rubber. (Author, 2017)


[Figure 32] Flat and Inflated version of the latex rubber prototype compressed between two layers of Masonite. 10 PSI air pressure was enough for inflating this prototype unit. (Author, 2017)

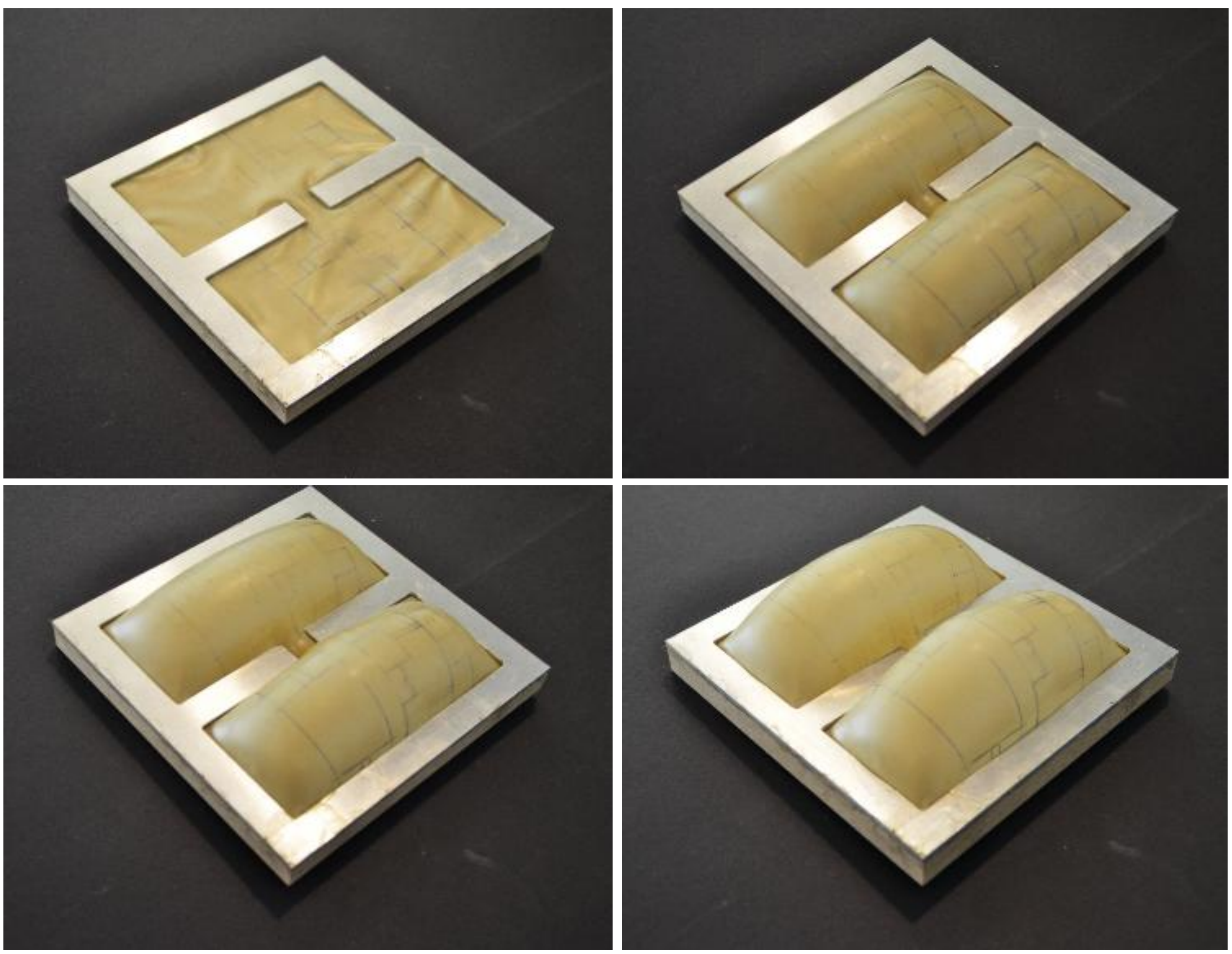
[Figure 33] Flat and Inflated version of the latex rubber prototype compressed between two layers of Masonite. 10 PSI air pressure was enough for inflating this prototype unit. (Author, 2017)



[Figure 34] Some of the shapes that were tested for better inflation feedback. Among them, those with square shape bladders were better inflated and better retained the square shape. 10 PSI air pressure was enough for inflating this prototype unit. (Author, 2017)

In the first design trials using this method, latex rubber is compressed between two layers of Masonite. The Masonite was not strong enough to withstand the air pressure and fractured. Hence in the next prototype, a layer of cut aluminum sheet on top, and a layer of ½" ply wood at the bottom (Under the latex rubber), were used. Both materials were not only strong enough to contain the inflatable latex rubber, but were visually arresting for the final exhibition. The aluminum was cut using a CNC router that resulted a fine cut edge. The three layers of material where compressed together using industrial contact cement, an adhesive that proved to be a very strong, and equally effective at bonding different layers of materials. The ply-wood was also drilled for a 1/8" tubing which supplied compressed air into the bladder.

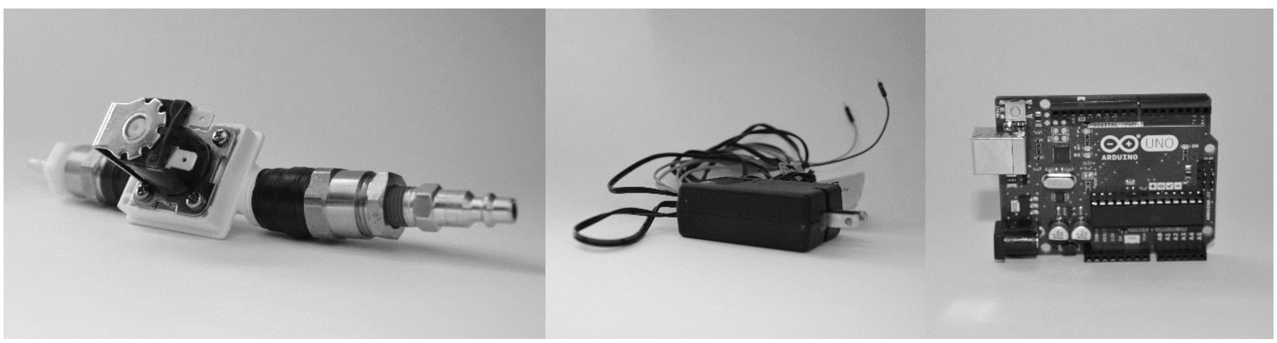
[Figure 35] Aluminum and wood sheets. The Aluminum is cut using a CNC machine. The wood was hand sawn. (Author, 2017)



[Figure 36] Flat and Inflated latex rubber compressed between aluminum and wood sheets. 10 PSI air pressure was enough for inflating this prototype unit. (Author, 2017)

3.3.2. Computational Devices

According to Rasmussen et. al., the *Tentacl* could be categorized as a non-interactive shape-changing output. There were no human inputs involved in controlling how the shape change could happen. However, this project introduced me to the solenoid valve, a device that allowed controlling the air flow through the Arduino electronic kit [Figure 37]. Controlling the inflation through the use of Arduino was vital in a sense that the valve could be connected to any sensory system. The sensors being the input of actuation, and the solenoid valve working as the output. At this point, the decision was to design an inflatable floor that could be used as a playful and interactive space for those who sit on it. The research was then extended in testing the microcontroller, Arduino, solenoid valve, the sensors, and the inflatable materials.



[Figure 37] (a) Solenoid Valve (b) AC Adaptor to power the Solenoid (c) Arduino Uno to control the solenoid. (Author, 2017)

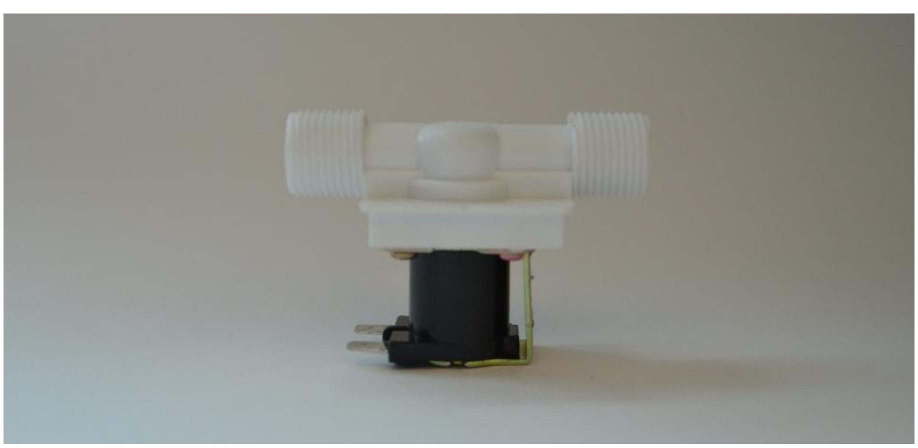
A. ARDUINO

A microcontroller is a small computer in shape of a chip that contains processors, memory, and programmable input/output peripherals[46]. While an Arduino is not a microcontroller, it contains a ATmega238 microcontroller and other electronics to make electronic prototyping easier. By Connecting the Arduino to a computer using a USB cable, it can implement and re-implement a program that has been written in the Arduino open source programming environment. The Arduino language is based on JAVA and similar to Processing that is usually used for programming visual output.

[Figure 38] Arduino Uno. (Author, 2017)

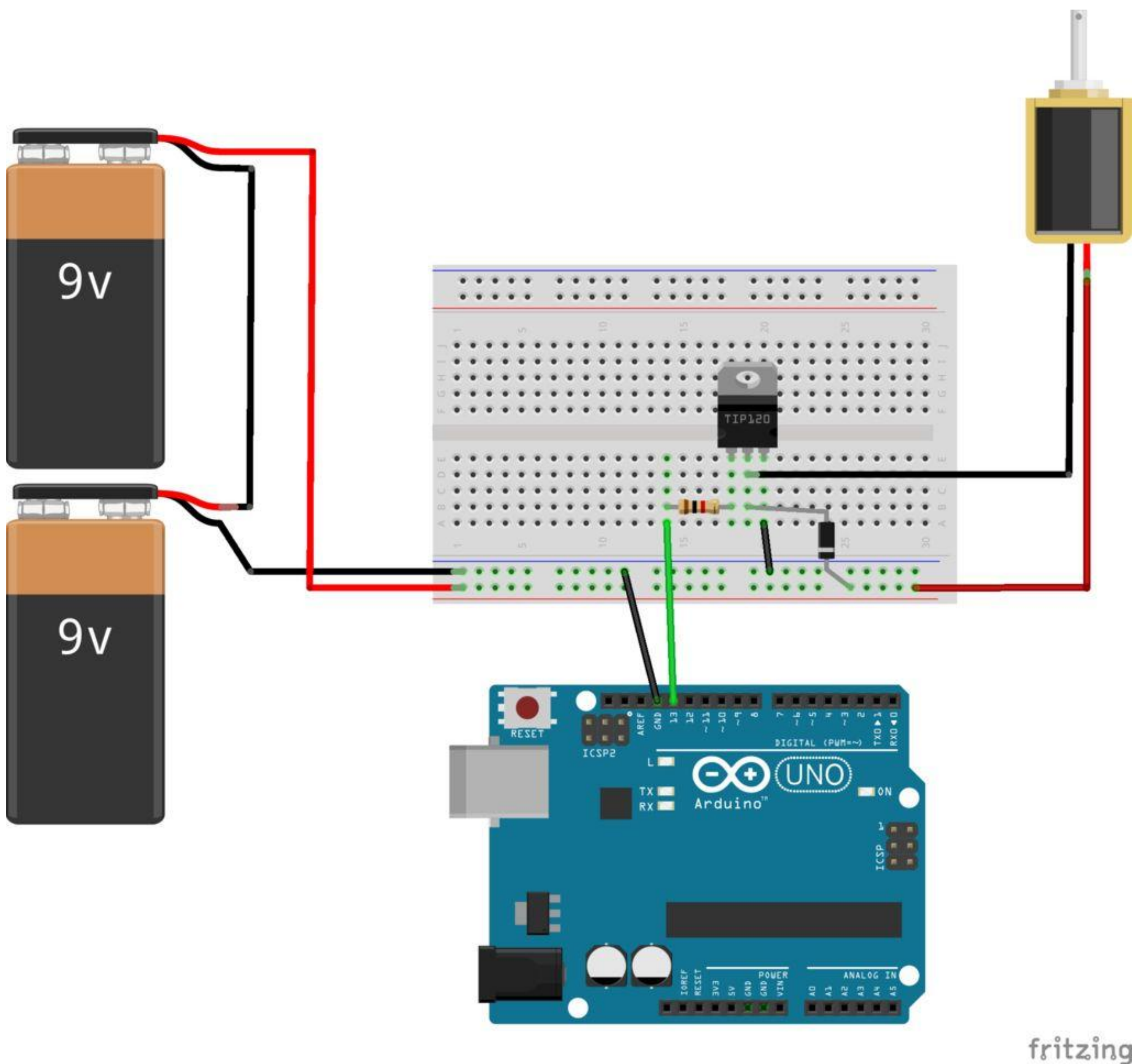
B. Solenoid Valve

The solenoid valve, as for its name, is a valve attached to a solenoid that could open or close the valve through a closed circuit. By connecting the solenoid valve to a 12-volt DC adaptor, the valve opens up, allowing air/fluid to pass through the valve. As soon as the circuit is interrupted, the solenoid closes the valve, interrupting the flow (in this design air). Connecting the solenoid valve to an Arduino made it possible to control how and when the solenoid opened up and closed through Arduino code. The circuit for controlling the solenoid through the Arduino kit was inspired by an Instructable open source tutorial[47] [Figure 43]. The parts to set up the circuit are below:



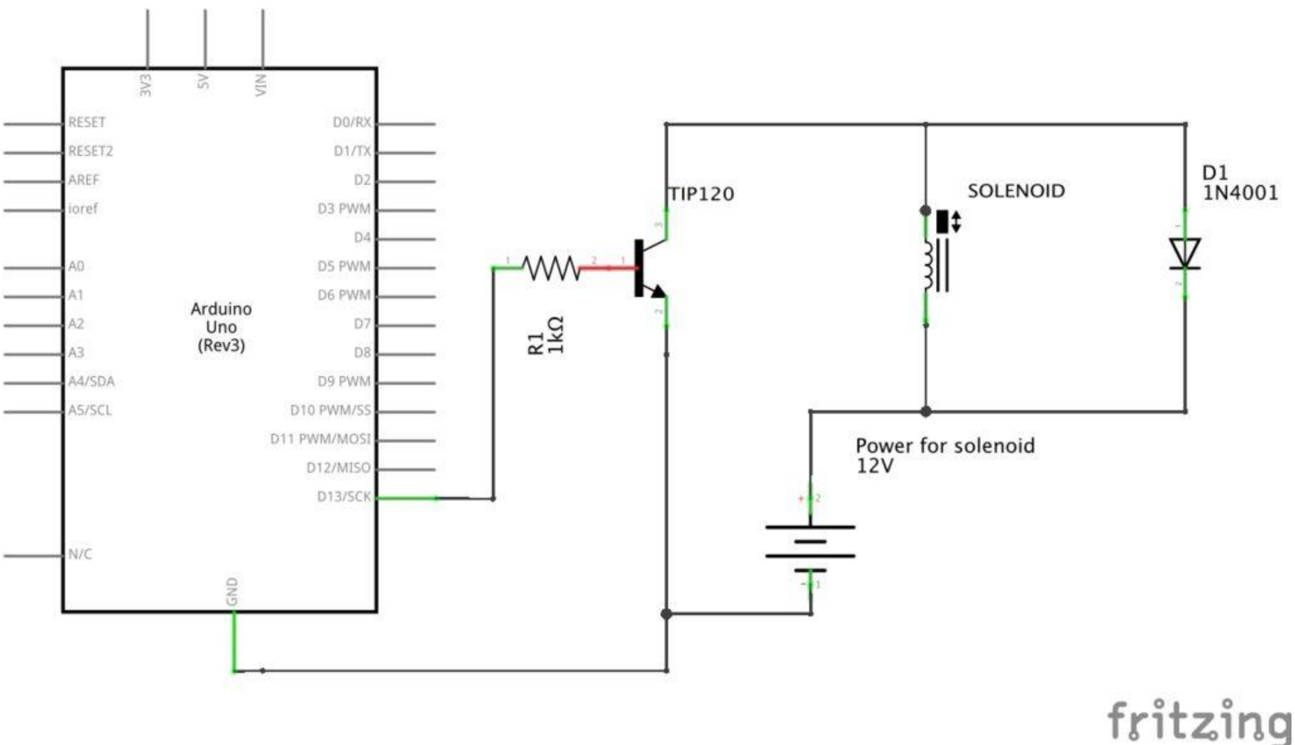
[Figure 39] Solenoid Valve. (Author, 2017)

- Arduino board
- USB cable for programming and powering the Arduino
- Breadboard
- Some jumper cables
- A 1K resistor
- TIP120 transistor (TIP102 will also work fine)
- 1N4004 diode (1N4001 also works)
- 12-volt DC adaptor
- A solenoid with leads to connect to the bread board

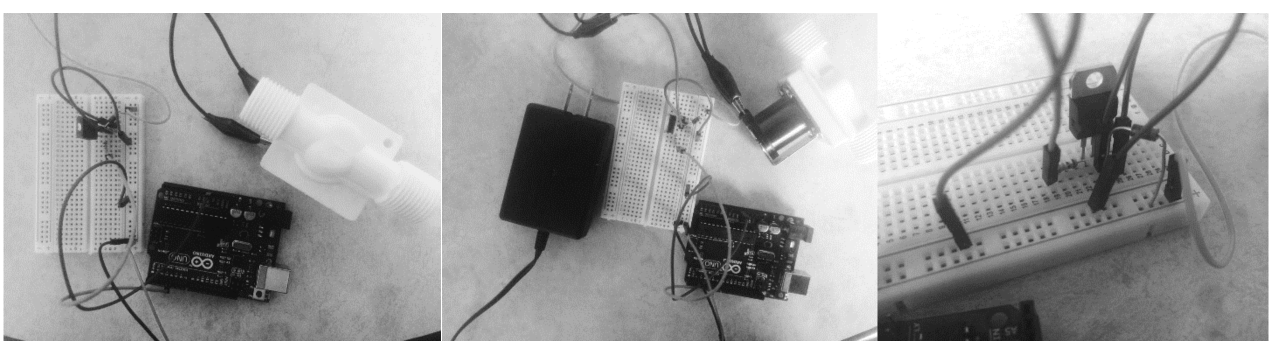


[Figure 40] A Fritzing sketch of the circuit that connects the solenoid valve to and Arduino. (Instructables. 2017) http://www.instructables.com/id/Controlling-solenoids-with-arduino/

In the circuit used for this research, the batteries were replaced with a 12-volt adaptor and the Arduino was powered separately from the computer. The Assembled circuit is shown in figure 42. What powers the solenoid should be different than what powers the Arduino. The details could be found in another work presented in the BC ROBOTICS website.[48]



[Figure 41] Circuit diagram sketch. (Instructables. 2017) http://www.instructables.com/id/Controlling-solenoids-with-arduino/



[**Figure 42]** The circuit connecting the solenoid valve to Arduino. The simplicity of the circuit was an important factor for scaling up the design later on in the design. (Author 2016)

The tutorial explains how and why all the electronic pieces are connected together in the circuit. "When the transistors labeled side is facing up the legs (from left to right) are B, C, E: Base, Collector, Emitter. The output pin of the Arduino will be connected to the Base leg of the transistor through a 1K resistor. The Collector leg of the transistor will be connected to the ground leg of the solenoid valve. And finally, the Emitter leg is connected to the ground channel of the circuit".[49]

Further on, the tutorial explains:

"The ground leg of the solenoid is connected to the collector leg of the transistor. The diode connects the power channel to the solenoid-ground-leg/transistor-collector-leg, preventing the kickback voltage from damaging the circuit. The diode is polarized and should be oriented with white/sliver stripe on the power channel side of the connection."

After making the circuit, a simple trial code was written in Arduino to control the solenoid [Figure 43]. The code simply commands the Arduino to open and close the circuit with certain timing (delays). As the circuit opens and closes, the valve is also opened or closed.

```
int solenoidPin= 4;

void setup() {
  // put your setup code here, to run once:
pinMode(solenoidPin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
digitalWrite(solenoidPin, HIGH);
delay(1000);
digitalWrite(solenoidPin, LOW);
delay(1000);
```

[Figure 43] Arduino code for opening and closing the solenoid valve on timed delays. (Author, 2017)

3.3.3. Pressure Sensor

Bodily interaction on a horizontal surface could be detected in multiple ways. The initial trial was tied to sensing the body pressure on the surface itself as an integrated and seamless approach towards implementing computational devices into everyday objects.

Based on the work published in Kobokant[50], I explored a Matrix Pressure Sensor that not only gives feedback about the location that has been pressed, but also how hard the pressure is. Basically, the pressure sensor is based on the change of resistance of an anti-static material and measuring that resistance. The full detail of how this pressure sensor was made could be found at a tutorial for Matrix Pressure Sensor[51].

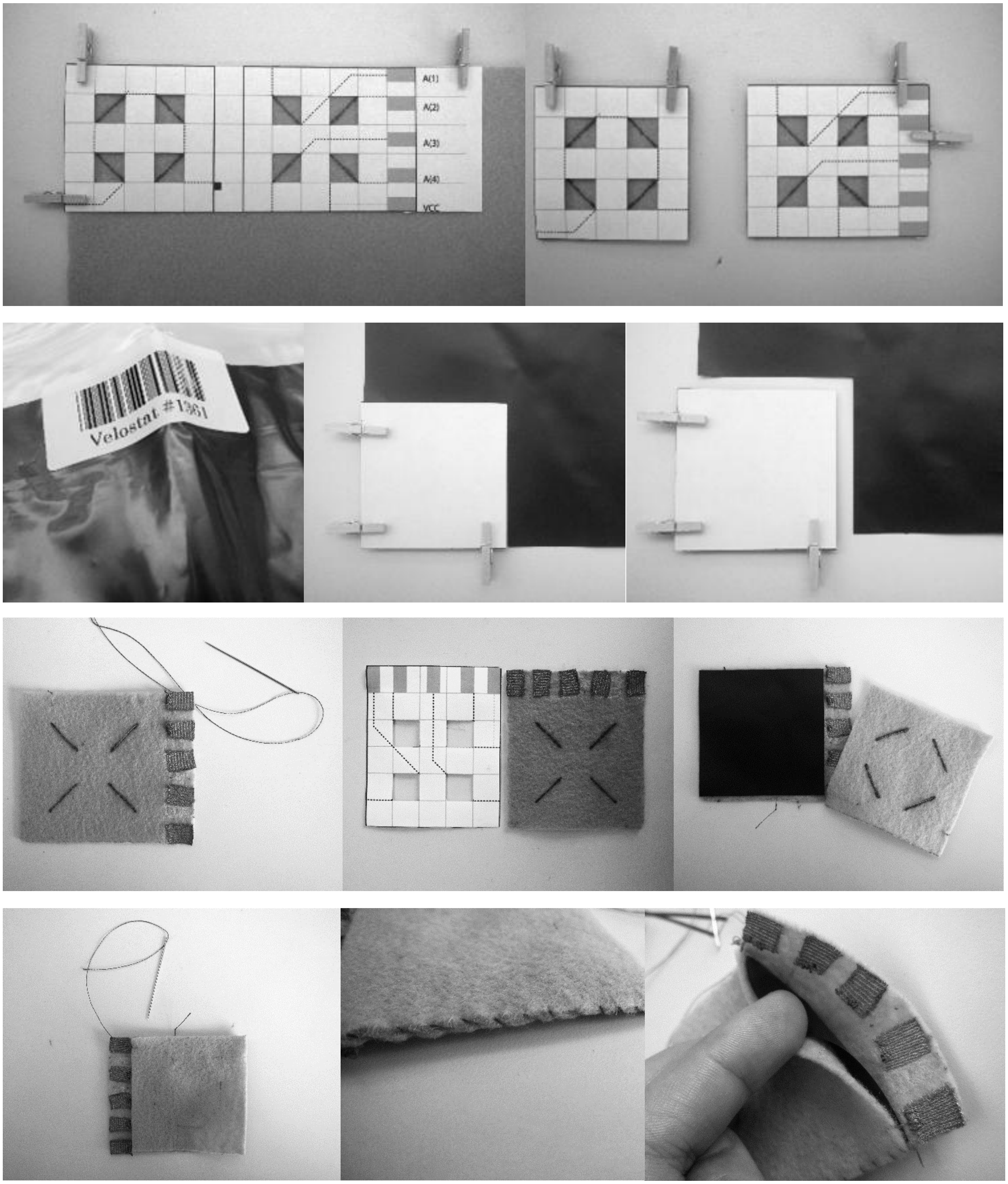
The materials used for making the sensor are[52]:
For sensor:

- Conductive thread
- Neoprene
- Stretch conductive fabric
- Regular thread
- Velostat: Velostat is the brand name for the plastic bags in which sensitive electronic components are packaged in. Also called anti-static, ex-static, carbon based plastic bags.

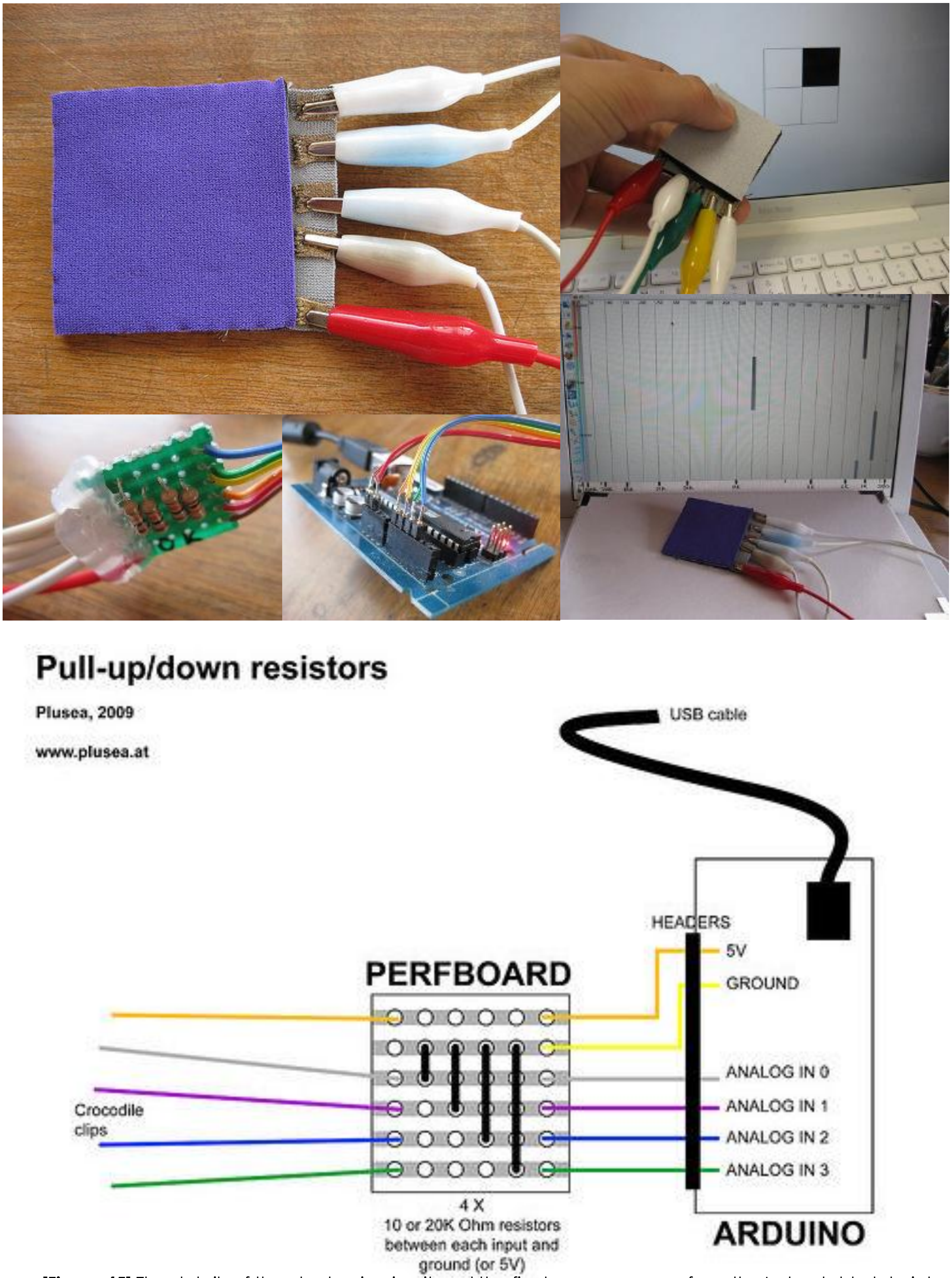
For reading input into the computer and running an application that visualizes the changes in resistance, the pressure sensor was connected to an Arduino. The components for building up the circuit include the following:

A. Arduino software free for download
B. Processing software free for download
C. Arduino USB board
D. Solderable Perfboard with copper line pattern
E. Crocodile clips
F. 4 x 10 or 20K resistors

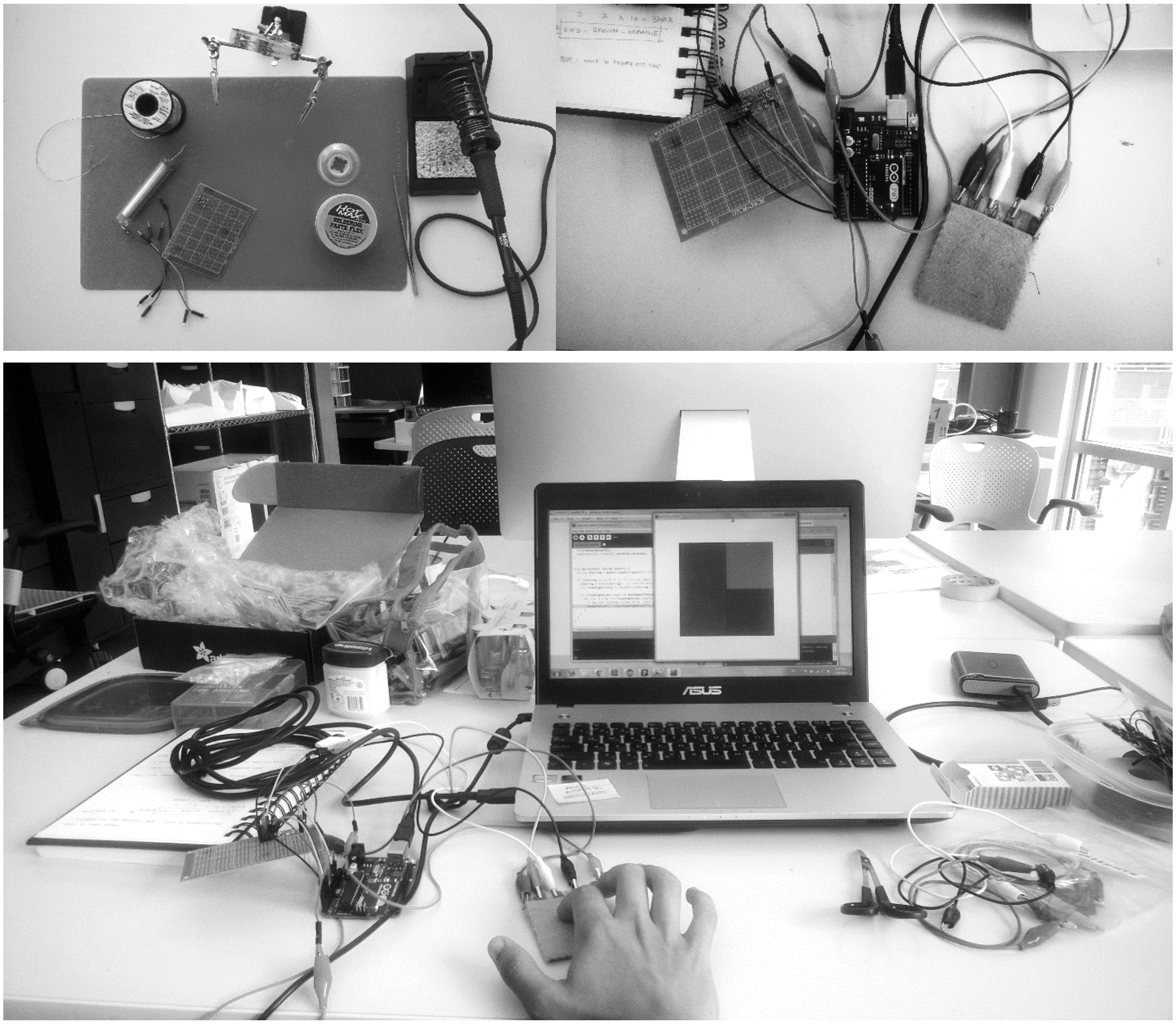
The details on how the pressure sensor was made can be seen in figures 44-46 in steps.



[Figure 44] Making the pressure sensor using conductive fabric, conductive thread, Velostat, and fabric. (Author, 2016)



[Figure 45] The details of the electronic circuit and the final pressure sensor from the Instructable tutorial.
http://www.instructables.com/id/Pressure-Sensor-Matrix/?ALLSTEPS



[Figure 46] Connecting the Matrix Pressure sensor to Arduino and reading the data through processing. The matrix identifies the area that pressure has been applied and represents it in the computer. The grayscale spectrum stands for the amount of pressure. (Author, 2016)

Even Though the sensor was a success, it was not considered as an efficient way for detecting body posture or position. Besides technical problems due to the complexity of the circuit if extended to a matrix bigger than 2*2, the pressure sensor could not distinguish between someone who is only standing on the floor, and the one who is sitting. As long as there is pressure, it is considered as somebody's presence regardless of their posture. To achieve better results, I decided on using the Kinect 360 as a visual sensor that could result in more accurate postural data. Then what was compensated was an integrated sensory system within the floor itself. Using the Kinect meant not having sensors embedded in the object itself; which was a compensation that had to be made for better design feedbacks and outcome.

3.3.4. Kinect – Posture Detection

Using a Microsoft Kinect 360, the gestures of those who are sitting on the piece or interacting with it is detected. This device was originally developed to capture human motions for Xbox360 video game console. The device is able to capture RGB color, depth through VGA camera in the middle and infrared projector and sensors. The principle of this device is same with other computer vision, calculating depth, the distance between the camera and objects, by calculating elapsed time of infrared light, emitted from the projector and return back to the sensor. But the device increases accuracy of recognizing and distinguishing multiple objects from a scene through its own imbedded algorithm. Because of this benefic, it gradually expands its area to interactive design, commercial, and academic fields. For this project three dimensional depth data of Microsoft Kinect for Xbox 360 was used to recognize presence of an object in a certain area. It is programmed in Processing environment with OPENNI library.

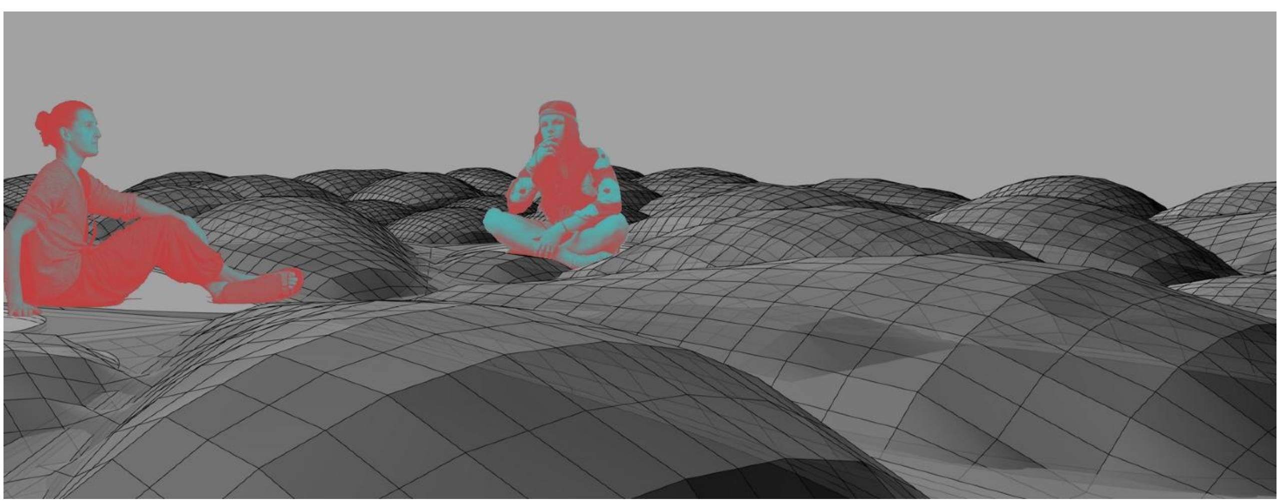
[Figure 47] Microsoft Kinect 360.

4. DESIGN CONCLUSIONS

After exploring design potentials through material possibilities and the availability of computational devices for designers, this section brings a conclusion to the design decision. From finalizing the design concept to representation, simulation, system design, material processes, and finally constructing the true to scale prototype.

As discussed before, the idea of bodily interaction on the ground level was the core concept of the design. Maintaining that concept, it was time to envision what the final prototype would actually look like. As some of the design principles were indicated, it was the aim of this design to construct *an* ***elastic****,* ***acrobatic****,* ***infinite****, and* ***ephemeral*** floor that evoked the users to practice floor living as a novel way for experiencing space in a rested position.

Inflatable materials were considered as the final solution to this design idea. An inflatable floor that corresponds to the position of those who are sitting on it and can act as a living landscape through inflation and deflation. Figures 48 and 49 present an image of what this computationally mediated landscape would look like, depending on the number of people and their position sitting on the floor.



[Figure 48] The Ephemeral Dreamspace in shape of a floor. In this image inflatable landscapes are surrounding those who are sitting on it. (Author, 2017)

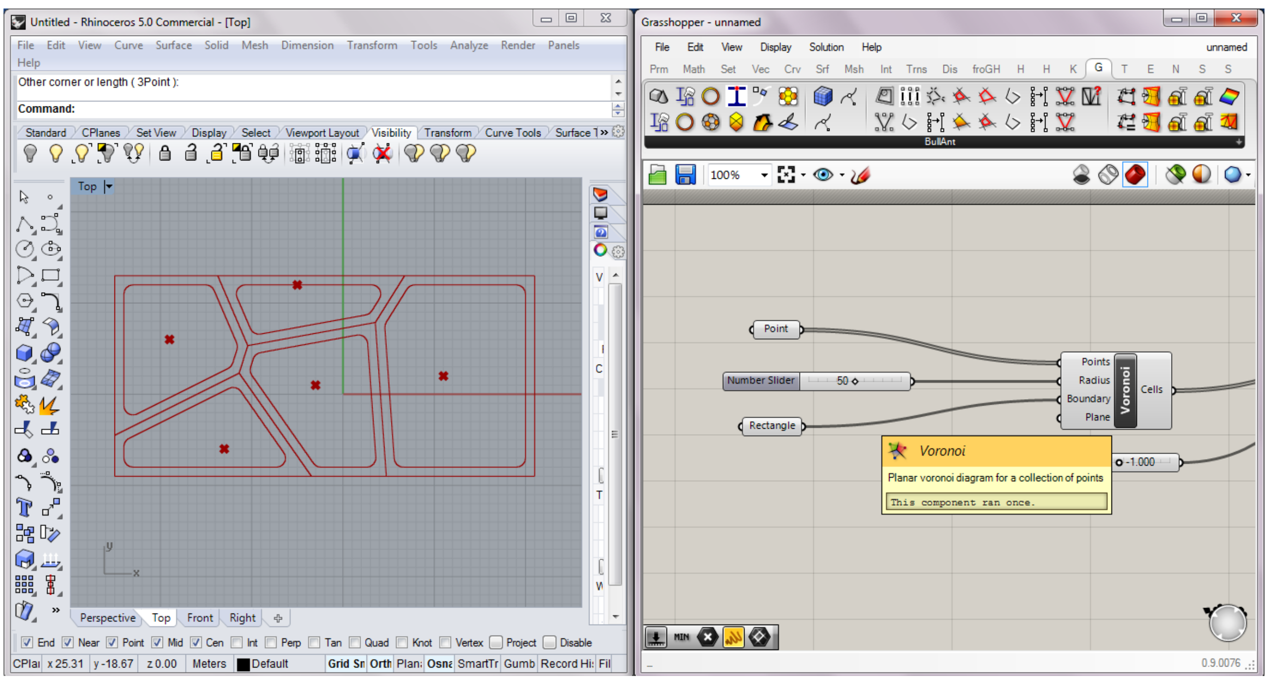
[Figure 49] The Ephemeral Dreamspace in shape of a floor. In this image the number of inflated sections (bladders)have been reduced to provide the space for those who would like to gather around and attend social activities. The bladders can ideally mediate floor living as the user leans on them, works on them, and etc. (Author, 2017)

The design of this dream space is based on parametric modeling that enables infinite but unique iterations. As the design can be expanded infinitely, this landscape provides an evocative context for different social activities on itself. The design is meant to not computationally reconfigurable in the horizontal level as well as to be adjusted based on the functions and bodily positions. The amount of inflation, the type of inflation, and other three dimensional factors can be controlled based on the functional use and desires of those sitting on it. Using Rhino 3d Modeling and the Grasshopper parametric modeling plug-in, these concepts were simulated which will be discussed in details in the next step.

4.1. Parametric Modeling

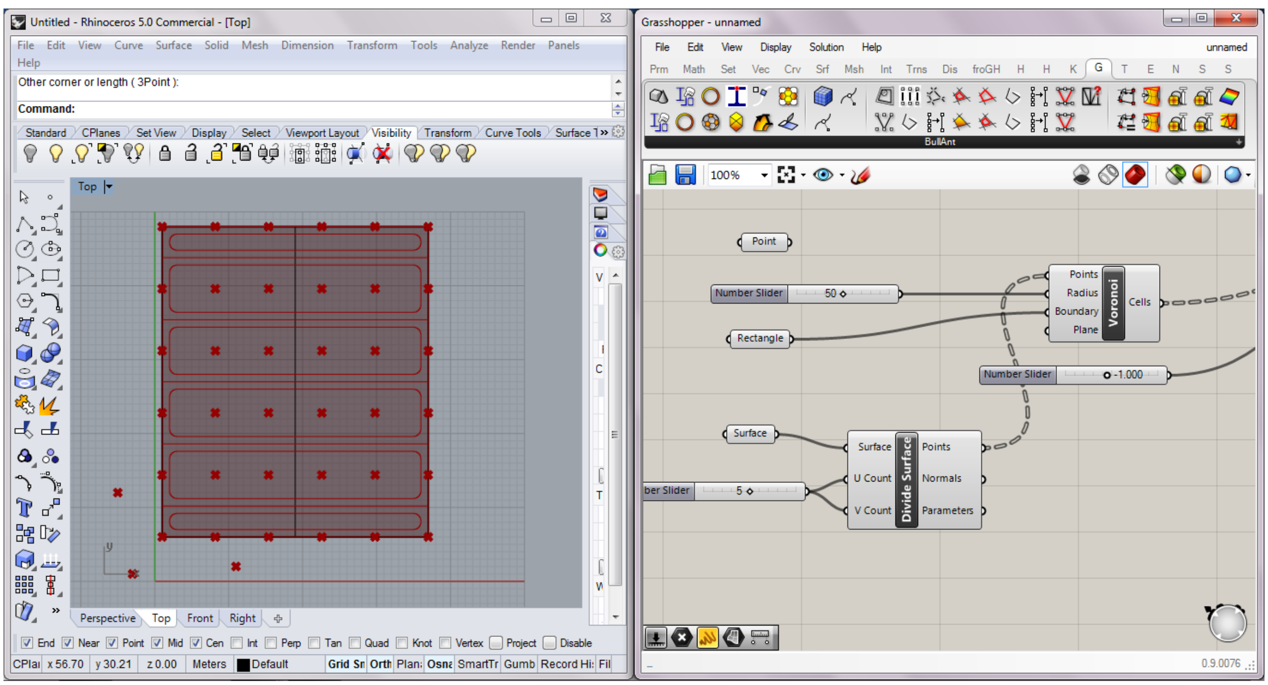
4.1.1. Simulating The Interaction

First step to simulate the floor interaction was to define the floor itself. To simulate the inflatable bladders of the floor, the floor was parametrically designed using a Voronoi algorithm to test the initial simulation. Even though the voronoi model was not selected for the final design, but it was a good example of a modular design that could be expanded and parametrically reconfigured based on point location. To develop the Voronoi geometry, I used the Voronoi component in Grasshopper [Figure 50]. In this iteration, each Voronoi module represented a bladder that could be fitted into a bigger whole. This was an important factor as it enables infinite expansion of the module depending on the design.



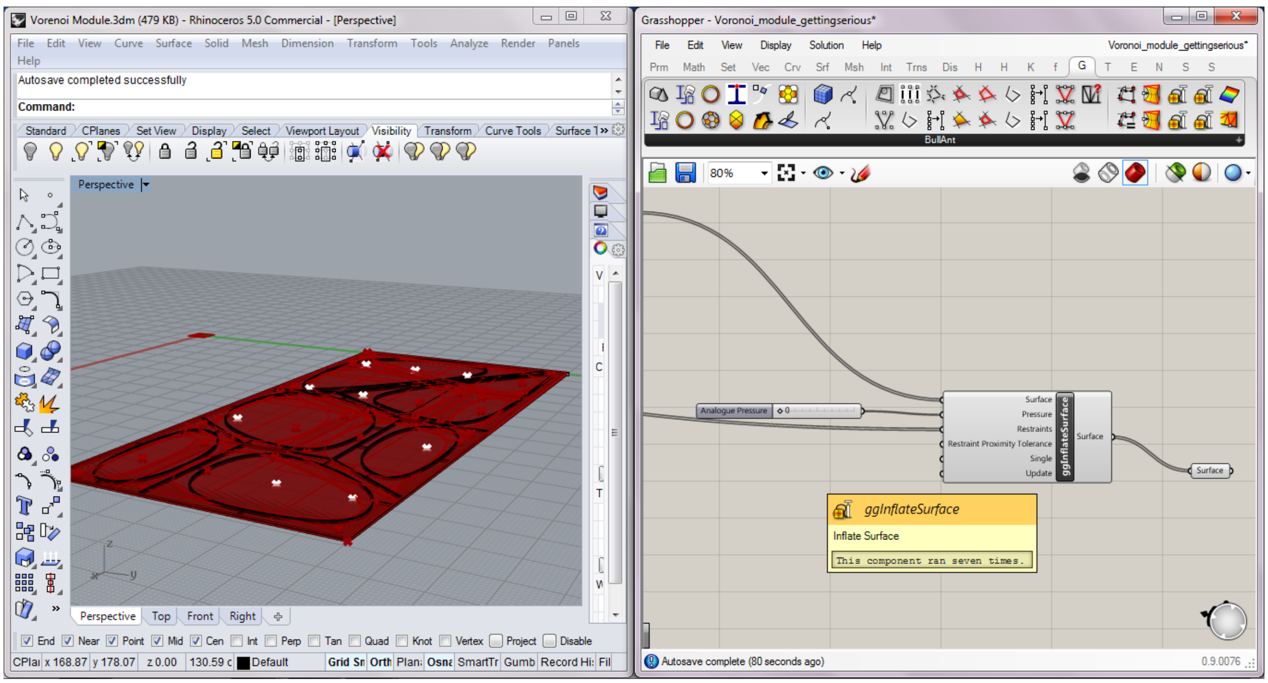
[Figure 50] A sample of 2d Voronoi Model using Voronoi Component in Grasshopper. Each Module Represents a bladder in the real design. (Author, 2017)

The Voronoi modules can be parametrically controlled depending on the location of the center points. However, the component does not allow control over the Voronoi's final shape. This was a downside to this predefined component since each module would have had a unique shape which was out of control. One solution seemed to be defining a point grid instead of feeding random points to the component [Figure 51]. Never the less, the final outcome of the design still would have been out of the designers control. Hence why for the final prototype, the design was customized in opposed to using built in components.



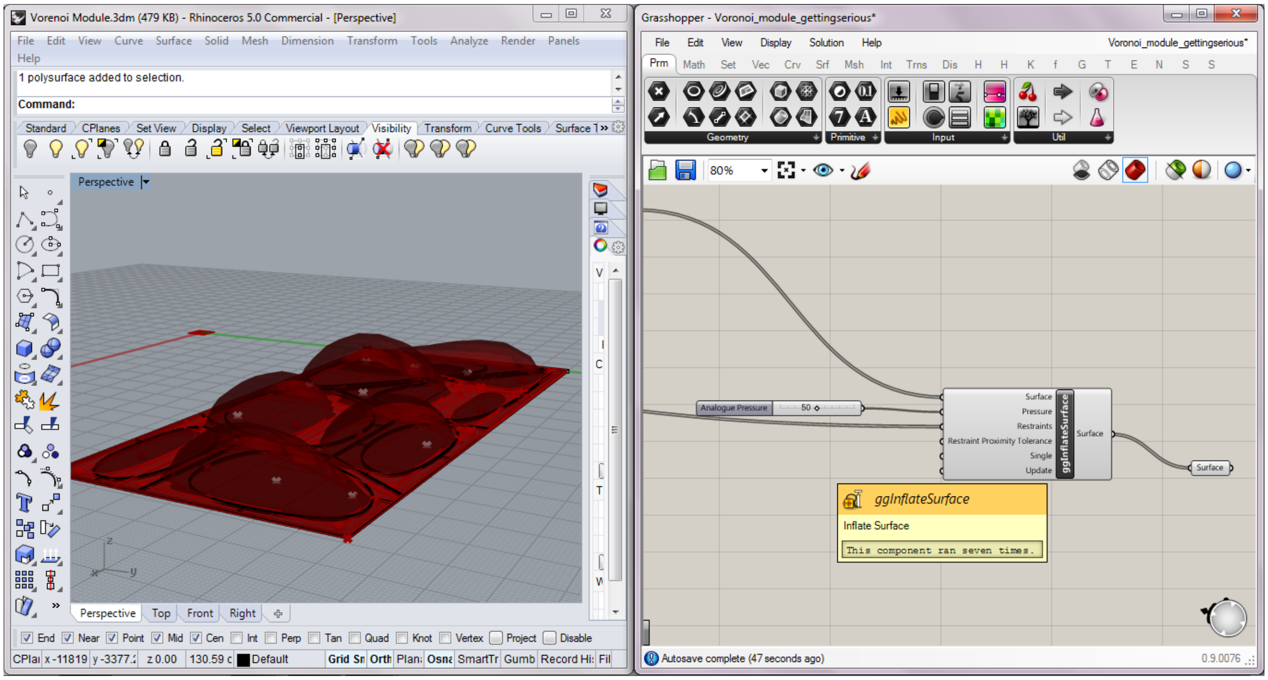
[Figure 51] A sample of 2d Voronoi Model using Voronoi Component in Grasshopper. Hear the center points are defined as grid. This model does not represent variety in design and is rather strictly defined. There is a need to define a design that would work between randomness and control at the same time. (Author, 2017)

Regardless of the pattern, I moved on to defining the inflation of the bladders using a Grasshopper plugin named Geomgym. GeomGym created by Jon Mirtschin[53] enables the designers to simulate inflatables in the 3d environment. Depending on the parameters, the inflation can be controlled in different levels. Figures 52, 53, and 54 show the different levels of inflation. Depending on the interaction scenario, the bladders could then be inflated or deflated. In the scenario chosen for the design of the floor, the area that a user is sitting, would deflate and the landscape would be bounded to the user's surrounding area (as shown in the concepts in the figures 48 and 49).

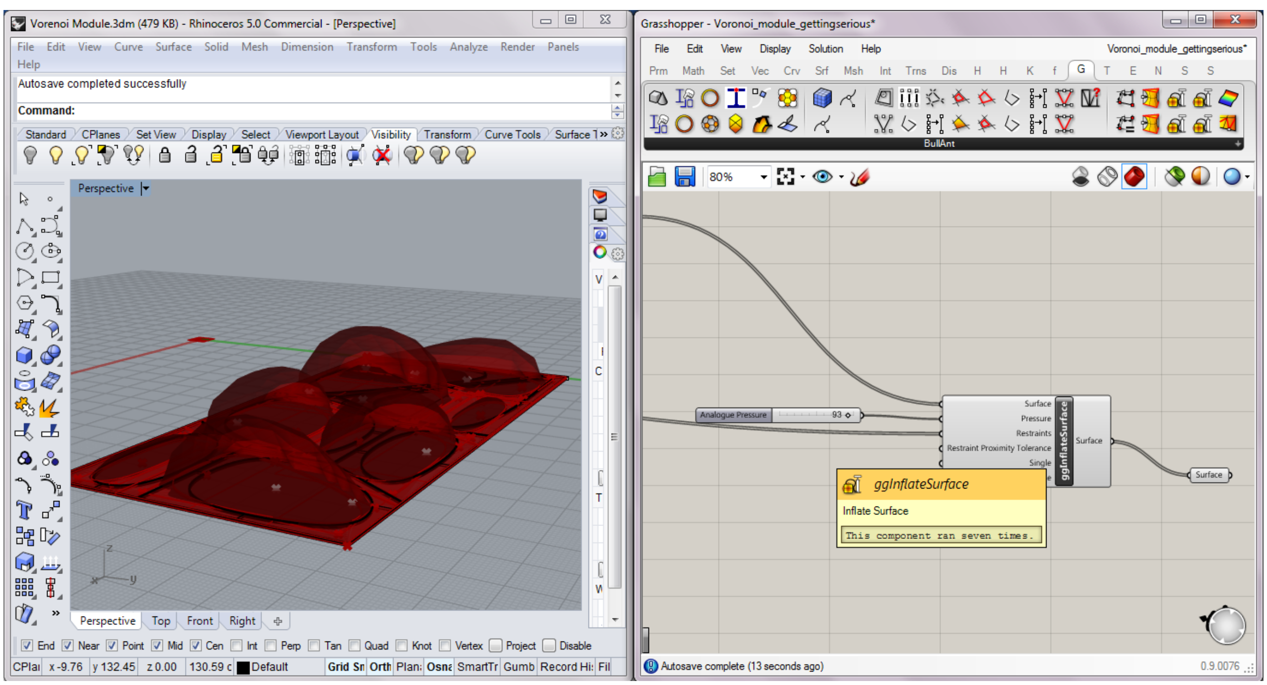


[Figure 52] The GeomGym Inflate-Mesh Component for Grasshopper. The inflation input for this iteration is zero. In the interaction scenario, zero equals to a person being on this area of the floor. In other words:

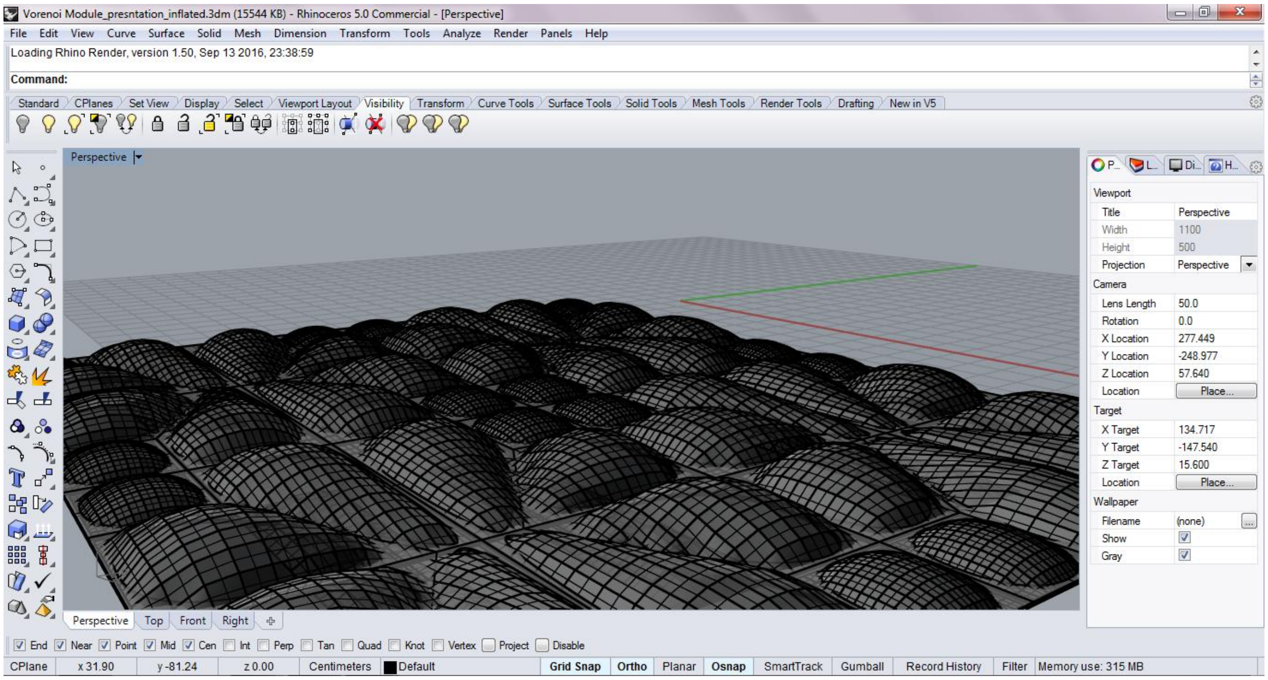
If :a body is detected on this area → (then) deflate. (Author, 2017)



[Figure 53] The GeomGym Inflate-Mesh Component for Grasshopper. The inflation input for this iteration is medium level. In the interaction scenario, a non-zero value equals to the floor being clear and nobody is detected in the area of inflation. In other words: If: no body is detected in this area → (then) inflate. (Author, 2017)



[Figure 54] The GeomGym Inflate-Mesh Component for Grasshopper. The inflation input for this iteration is almost 100 only to show the simulation capabilities and the possible size of inflation. For the design of the floor, depending on the time of inflation, the size of the inflated bladder can be different. (Author, 2017)

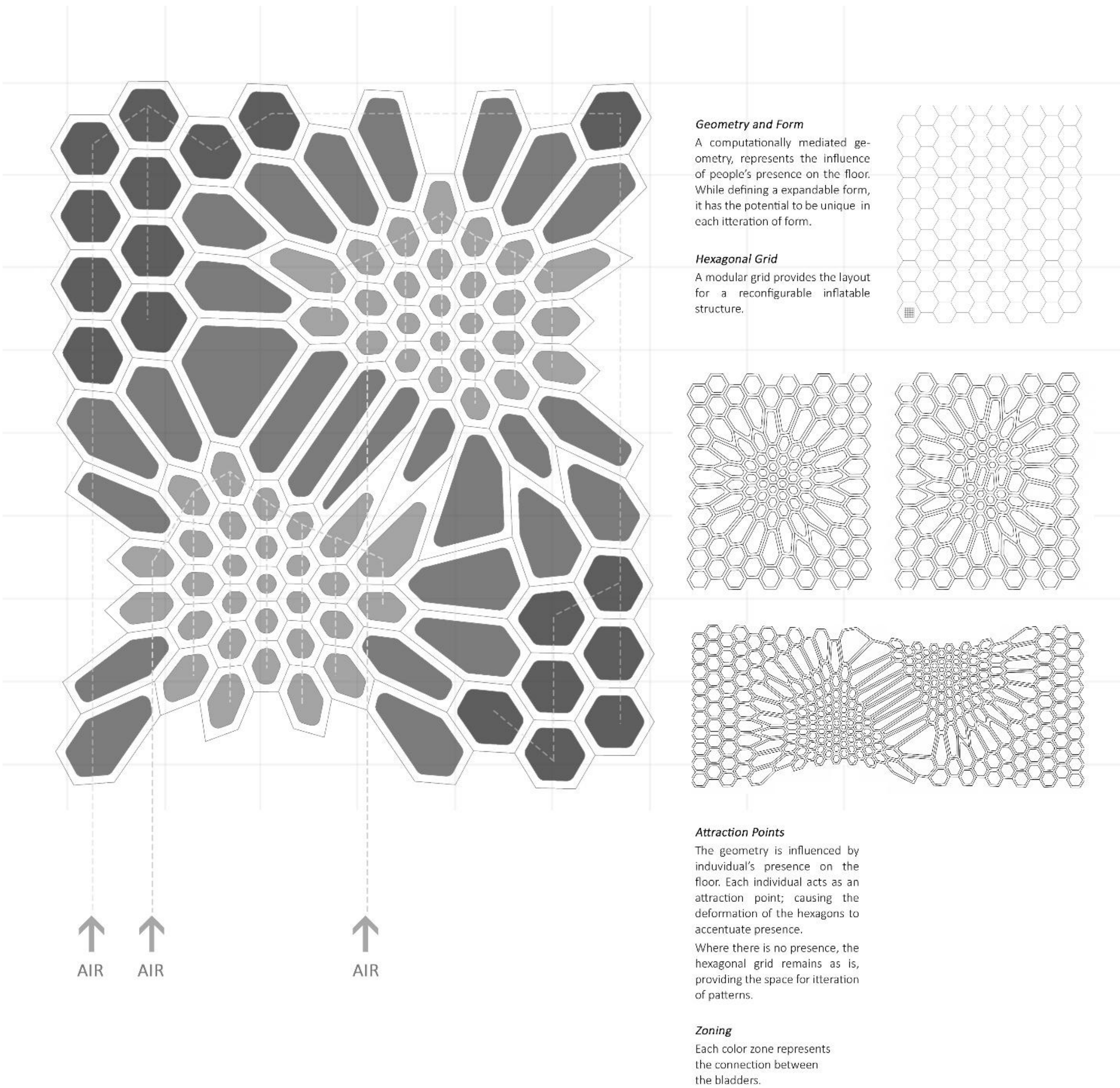


[Figure 55] An image of the floor expanded in multiple iterations. Parametric design allows expanding the floor, as a horizontal space definer, throughout the desired horizon. (Author, 2017)

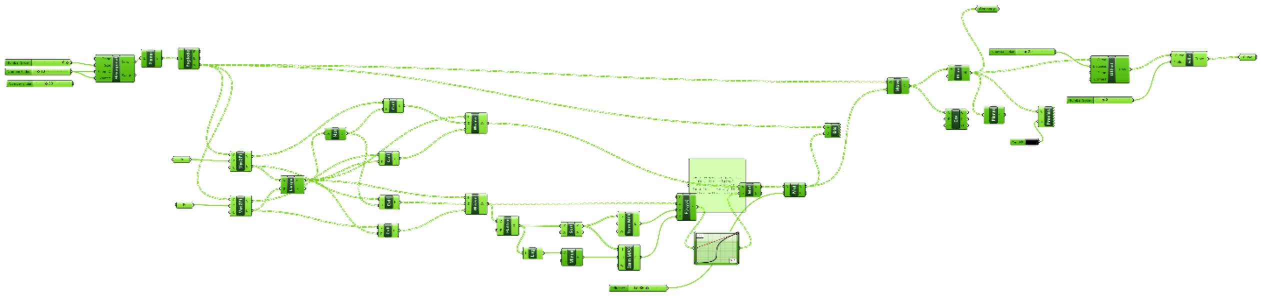
4.2. Grids and Disturbances

In this section I would discuss the design for a gridded system that facilitates the idea of expanding the floor system infinitely. However, in addition to the grid pattern, it was the aim of this design to evoke people to settle down and impact the behavior of the floor as a dream space or a living landscape. For this reason, I propose parametric design as a solution to represent the impact of an individual user on the architectural space. As the gridded system inherently represents homogeneity in design, the disruption of the grid can be said to be a sign of heterogeneity. Hence I define the individual user as a center point who would step into a gridded system and disturbs it in a controlled while non-homogenous way. In other words, parametric design is a solution to design variety within homogeneity.

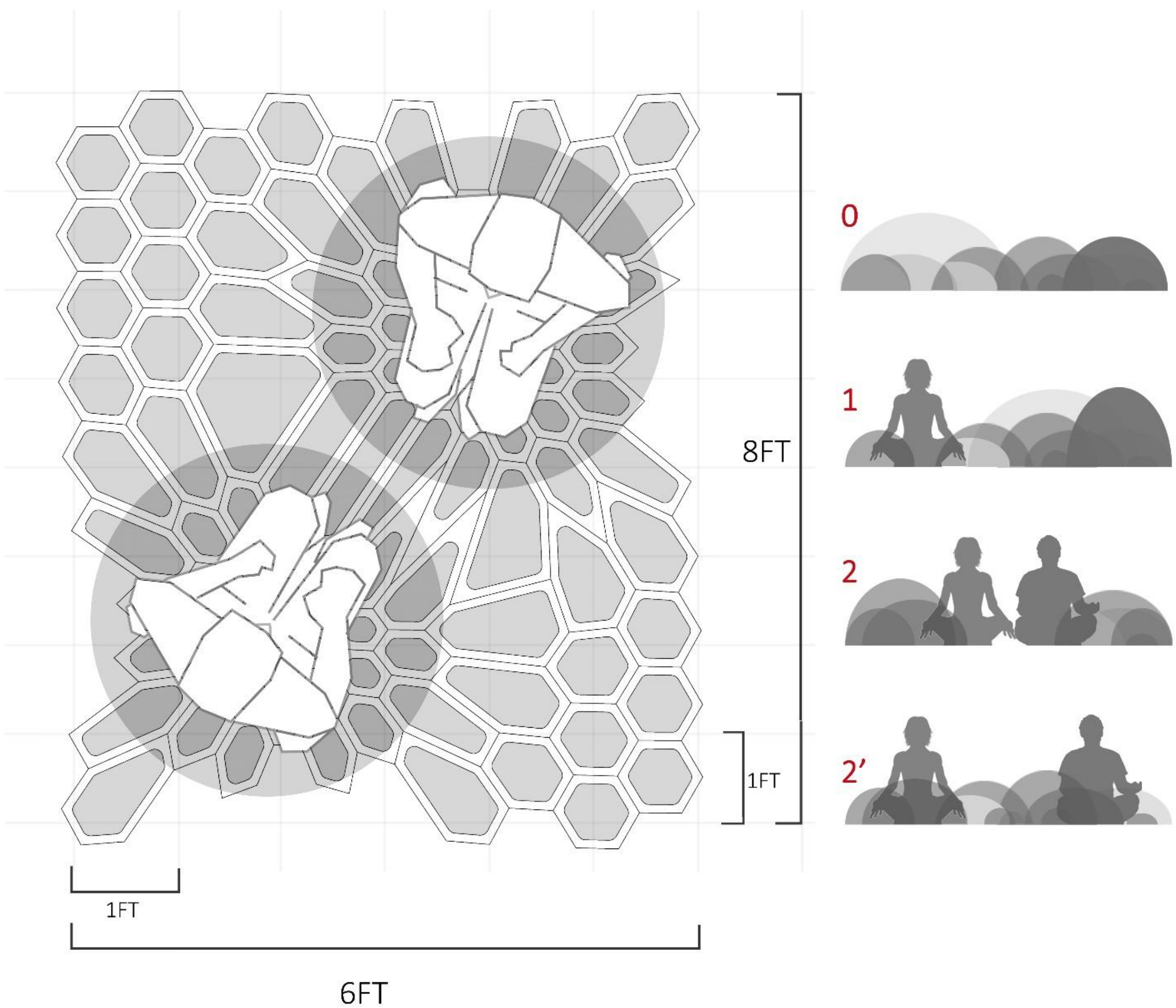
Starting from a homogeneous hexagonal grid, the hexagons provide the freedom for infinite expansion onto the landscape; While each hexagon represent a literal bladder for the design of the floor. In an algorithmic process, using Grasshopper, center points were added to the design reflecting the presence of a human body. As shown in figure 56, the hexagonal grid transforms as one, then two attraction points enter the grid. Depending on the distance between the two points, the geometry of the grid differs. The tension between the two attraction areas can be seen as they fall apart. This distance is solely dependent on people's comfort zones and personal space in a social context [Figure 58]. The behavior of the floor then could be adjusted based on these distances.



[Figure 56] A hexagonal grid represents a modular design that could be expanded infinitely. The impact of the individual user on the grid is then represented in certain areas on the floor. This image represents a selected version where two attraction areas exist for two users; one of many possible iterations in the design. (Author, 2017)



[Figure 57] This image demonstrates the complexity of the Grasshopper code that was developed to represent presence on the hexagonal grid. (Author, 2017)



[Figure 58] The size of the prototype was finalized based on a standard dimension for personal space. The 6*8 feet floor prototype would fit maximum two people on its own and would ideally react according to the positions of those who are sitting on it. (Author, 2017)

4.3. System Design

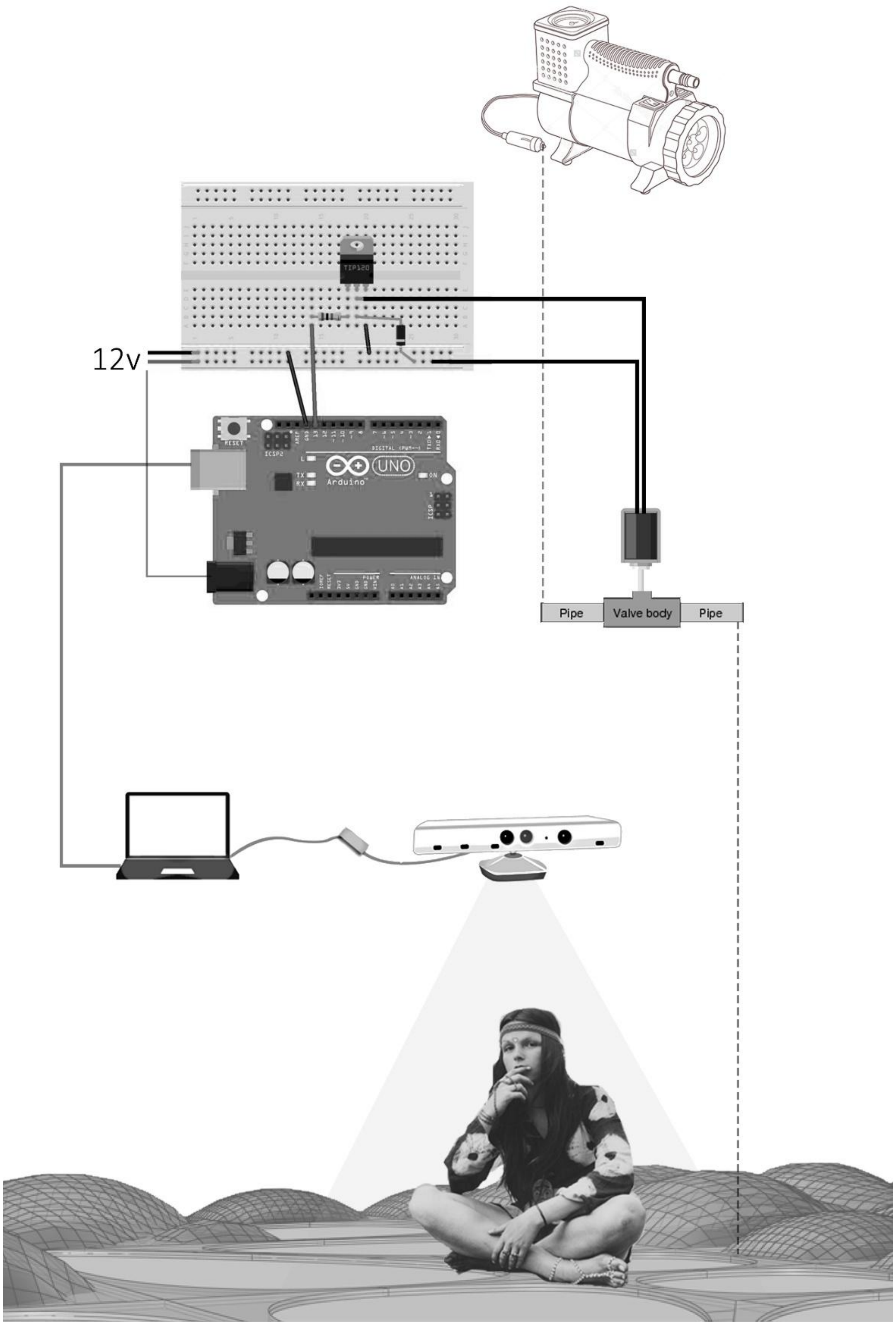
After finalizing the modular design of the floor and the interaction scenario, it is time to explain the technical sensory system of the prototype. With a focus on the exploration conclusions that was achieved in chapter three, the schematic design of the final system is shown in figure 59.

First off, the position of the users will be detected using a Kinect Camera. The Kinect provides 3D data on the X, Y, and Z position of the users. Depending on the user's location, a corresponding area would either inflate or deflate according to the interaction scenario.

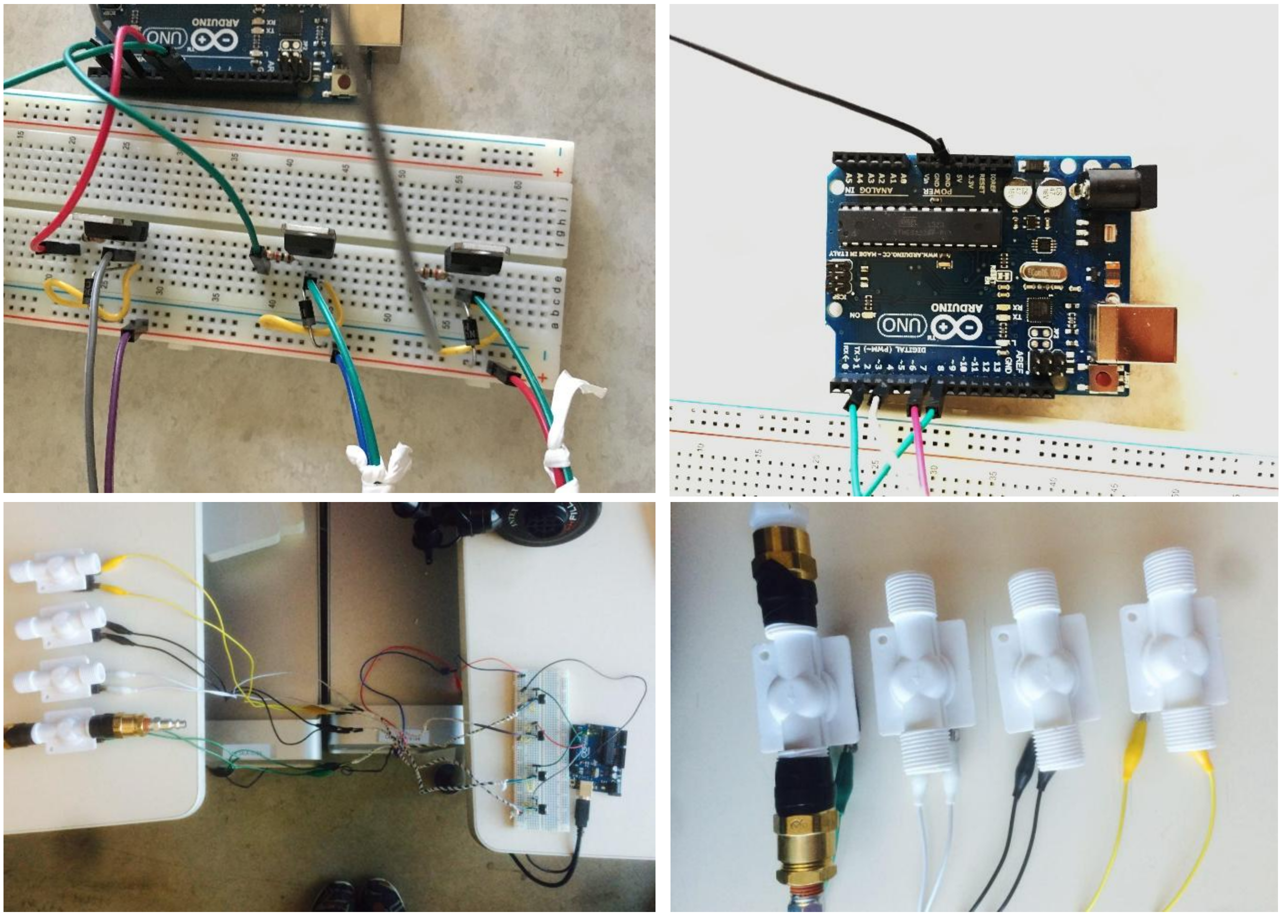
Secondly, the Kinect sends the detected data to the Processing interface where computational code commands the actuators to activate or deactivate. The Arduino electronic kit connect the Kinect sensor to the actuators through the computer.

Consequently, four solenoids are computationally actuated to either open or close the valve; letting the air to stream through the bladders. The solenoid valves are connected to the Arduino on one hand, acting as an output to the system, and to the compressor on the other, that provides the air for the system.

The challenge in this system goes down to the complexities of synchronizing the digital commands with the physical prototype. The barrier that has existed for designers to step out of the virtual and implement digital devices onto the physical matter. From this point beyond in this thesis, I will discuss these challenges.



[Figure 59] System Diagram: A. The air compressor feeds air into the solenoid valves. B. the solenoid valves are computationally controlled through Arduino code. C. The solenoids can be switched on/off to control the air flow. D. The Kinect locates the users on the floor and send triggers the command to the valves. (Author, 2017)



[Figure 60] The details of the circuit. Four Solenoid Valves are connected to the Arduino. (Author, 2017)

4.4. Physical Construction

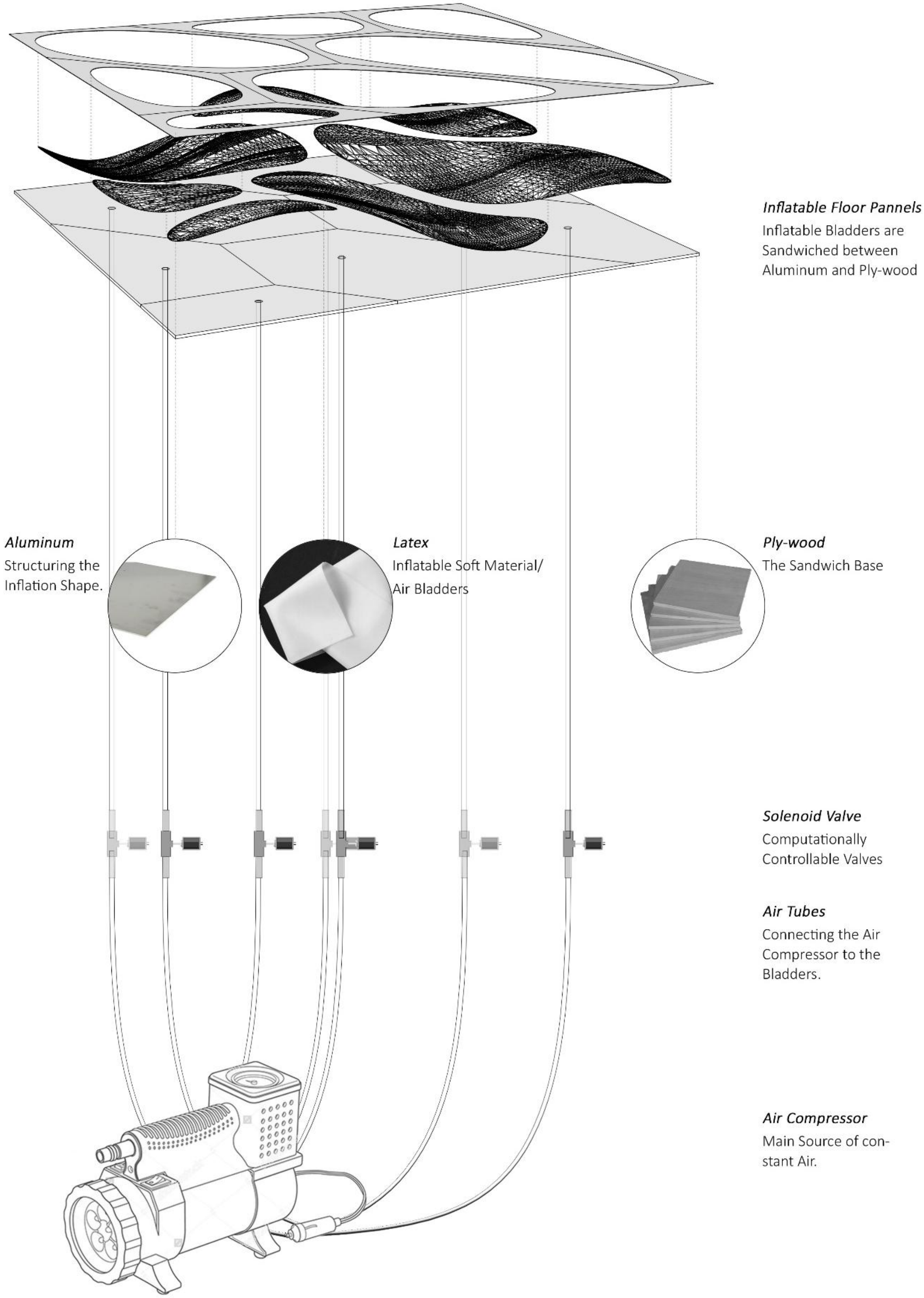
Prior to stepping into the hands on process of physical prototyping, it is important to plan the procedure ahead of time. In order to prepare the materials, think of possible issues on the way, and initiate problem solving before the problem occurs. Figure 61 presents a schema of the materials in whole; from shaping the inflatable bladders, using latex rubber and wood panels, to connecting them to the computational devices and so on. The materials that were used for the final prototype are listed below. The details of the hands on process and the use of each machine will be explained later on in full details.

G. Materials:

- White latex rubber sheet (0.2 mm thick): 3 Yards
- MDF Wood Panels (1/2" thick): 2 pieces of 2*6 Ft
- Plywood (1/8" thick): 2 pieces of 5*5 Ft
- Tubing (1/4" diameter): 10 Ft
- Tubing (1/8" diameter): 100 Ft
- Tube Fittings: per solenoids
- PVC Tubes (1/4" diameter): 4 Ft
- Contact Cement

H. Machinary and Tools

- CNC cutter
- Drill
- Hand Saw
- Compressor



[Figure 61] Integrated diagram of the materials and construction strategies. (Author, 2017)

4.4.1. Hands-On Physical Prototyping

As the design solutions were proposed on paper and in the format of digital drawings and simulations, it was time to step into the real world and start building. Considering the true to scale size of the prototype, the construction of the floor was a time consuming process. In this section, the process of physically making the true to scale prototype has been explained step by step.

A. The platform

The floor design was initially carved out on two pieces of MDF wood. The two pieces then were seamlessly joined to make a joint platform [Figure 62]. Even though the design concept was to make a modular design, it was decided to put all of the modules together on a single sheet only for convenience in building the prototype.

[Figure 62] The two MDF platform are joined together as a base for the inflatable floor. The person in the photo is the author sitting on the platform. (Author, 2017)

B. CNC Router

A Computer Numerical Control machine located in the Stuckeman School of Architecture's woodshop was used to cut the wood platforms [Figure 63]. The CNC machine was used to cut both the MDF and Plywood pieces. However, depending on the type of cut, different drills were used.

[Figure 63] The CNC is cutting through the Ply wood as the frame for the inflatable structure. The CNC Machine is located in the ground floor at the Stuckeman School of Architecture and Landscape Architecture at the Pennsylvania State University. Easy access to the machine made this process much more convenient. (Author, 2017)

C. Plywood Frames

The CNC Machine first cut the outlines of the frames while the brush was installed on the drill. The brush technically disperses the pressure onto the entire wood panel [Figure 64].

[Figure 64] The CNC machine cutting the outlines. (Author, 2017)

After the outlines are done, the brush was removed to cut the inner lines of the frame [Figure 65].

[Figure 65] The CNC Machine is cutting the inner lines of the frames. The brush has been removed for a clearer cut on the smaller pieces. (Author, 2017)

[Figure 66] The final Plywood frames. These frames go on top of the latex rubber sheets to from the inflatable bladders. On the right both the wooden frames and the MDF base are shown. (Author, 2017)

Because of the layers constituting the plywood, the edges of the plywood frames were chipped. It was important to file the edges using a sandpaper in order to prevent damage to the latex rubber sheets later on [Figure 67].

[Figure 67] Filing the edges of the plywood frames. (Author, 2017)

D. Matching Pieces

After cutting the latex rubber sheets using Rotary Cutter (a very time consuming task), it was time to match the puzzle pieces and get them ready to be installed on the MDF platform. 100 Latex rubber pieces were matched with the 100 plywood frames based on the coded design map [Figure 68].

[Figure 68] The Matched Latex Rubber Sheets and the Frames. (Author, 2017)

[Figure 69] All of the pieces were tested on the platform to make sure all pieces are in place. However, unfortunately, I came short on latex rubber and had to fill in some of the bladders. (Author, 2017)

E. Drilling the air holes

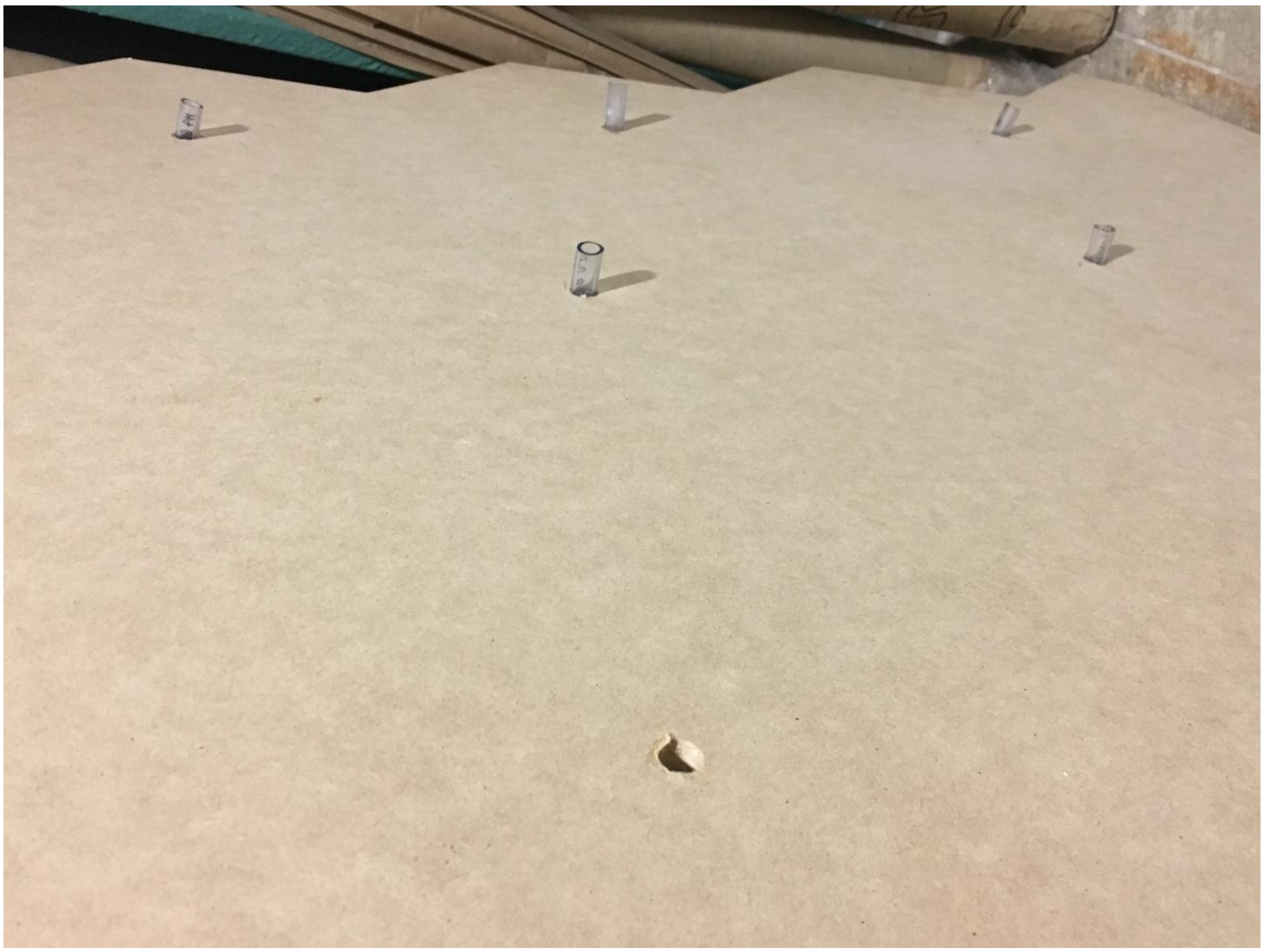
Having the pieces ready, it was time to drill air openings on the MDF platform. Using a simple drill, the holes were done to fit 1/4" tubing [Figure 73]. These tubes would act as permanent joints in the wood that could hold the re-adjustable 1/8" tubing that connect the compressor to the bladders.

[Figure 70] Drilled holes on the MDF platform before cleaning up the wood chips. (Author, 2017)

[Figure 71] The holes are cleared to fit the 1/4" air tubing. (Author, 2017)

F. Tube Joints

The tubing is inserted in a way not to make a bump on the surface where people would sit on. Just a bit longer that 1/2", the tubes are long enough to hold the tubes that go into the bladder. The plastic tubes are glued to the MDF wood panel using Super Glue.



[Figure 72] An image from behind the platform. (Author, 2017)

G. Making the Bladders

In this step, the physical parts of the floor come all together. Sitting on the floor [Figure 73] started here as the pieces are glued together. Contact cement worked as a very good joint between the wood and the latex rubber latex rubber sheets. In a step by step procedure [Figure 74] the jointing process is demostrated and then explained.

[Figure 73] Sitting on the floor prototype, the latex rubber is glued to the MDF and the frame is glued on top of the latex rubber using contact cement. In this Image almost half of the floor is ready to be inflated. (Author, 2017)

[Figure 74] A step by step visual demonstration of how the pieces were joined together. (Author, 2017)

1. The plywood frames are glued once and are left to dry. This insulates the perforated structure of the plywood and helps to glue the two wooden pieces more effectively using contact cement.

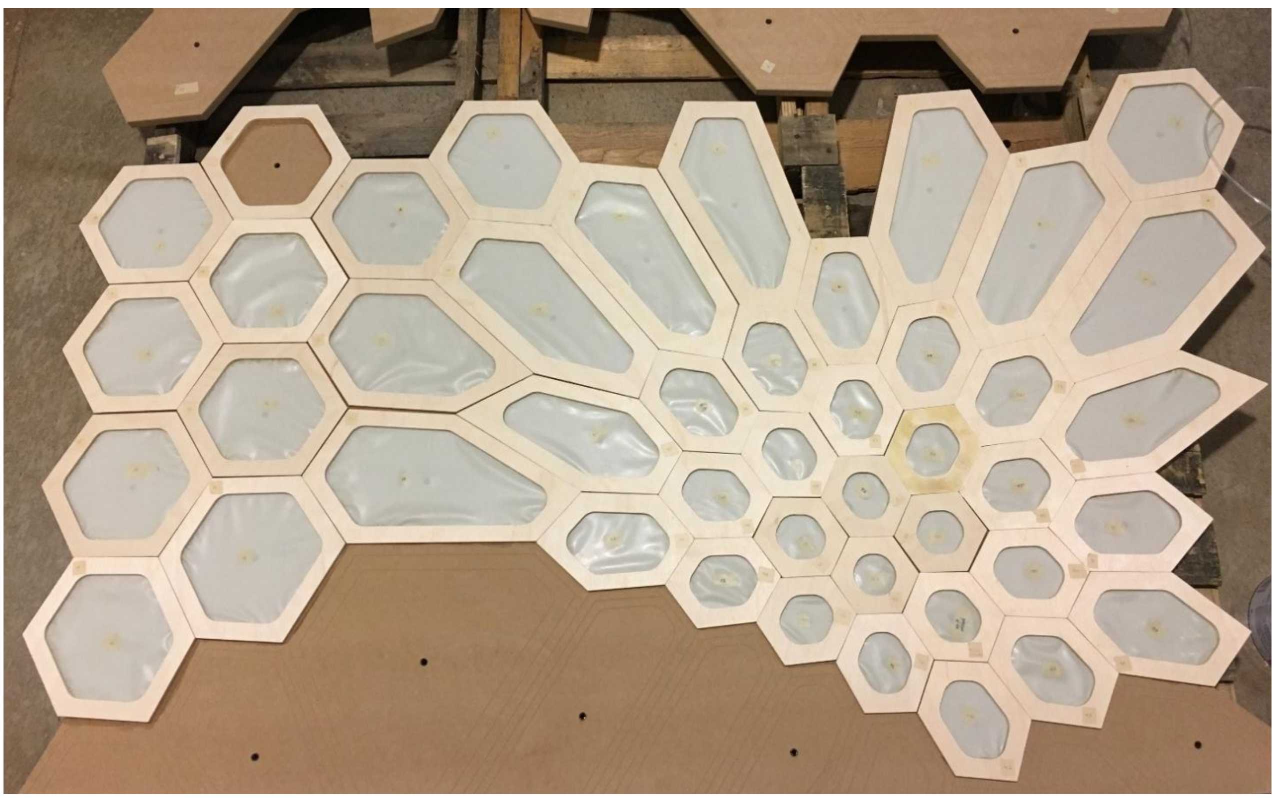
2. As the plywood is left to dry, Contact Cement is brushed on the parameter of each module. This step is to glue the edges of the Latex rubber to the MDF.

3. The latex rubber then goes on top of the MDF. The edges stick to the MDF which shapes the bladder.

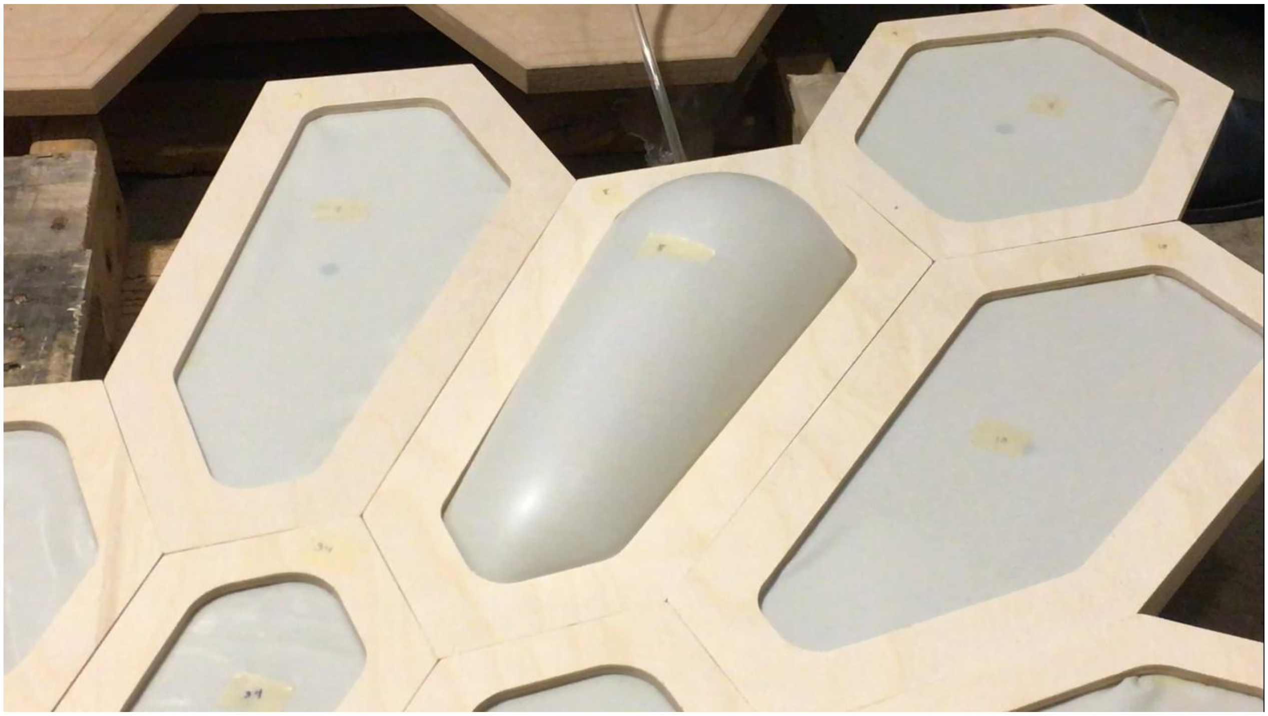
4. The plywood frames then go on top of the latex rubber to prevent any air leaks and keep the shape of the bladder in tack.

5. The Final step is to put enough pressure on top of each frame until the contact cement dries out and all three layers firmly stick. I used a heavy weight for this purpose. Leaving the weight for about 10 minutes usually worked fine [Figure 83].

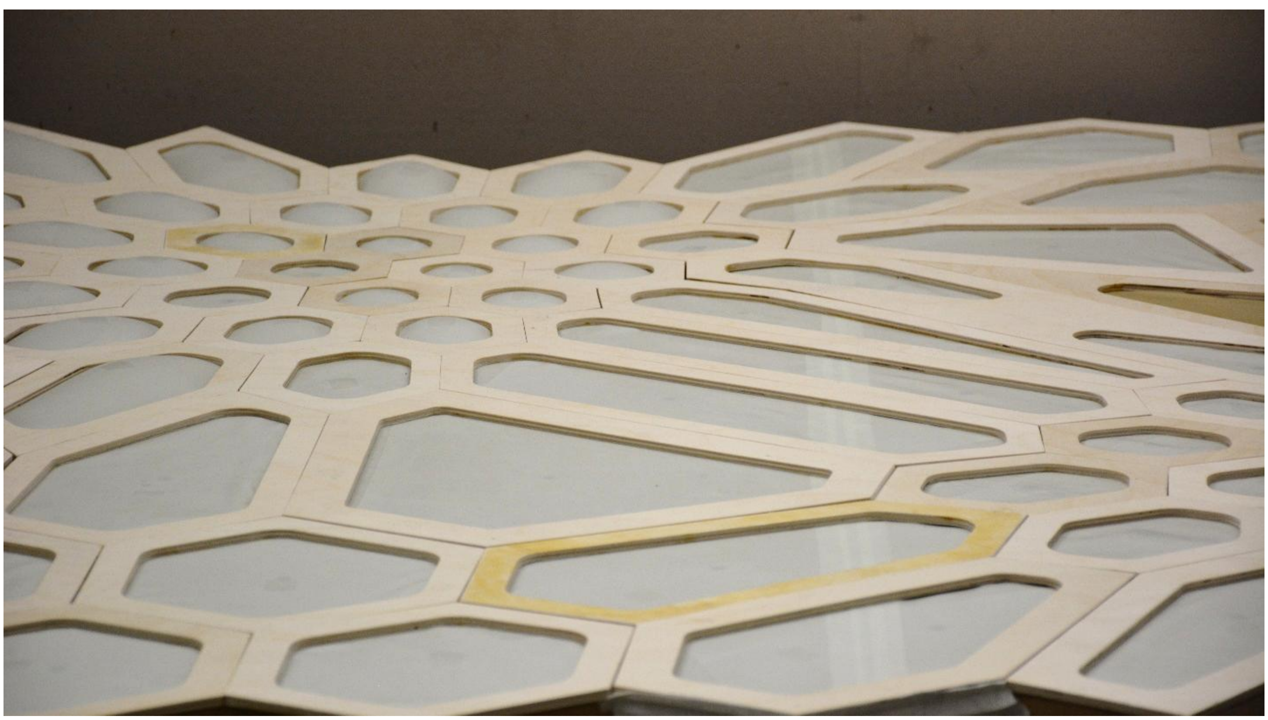
In figure 75 you can see half of the floor done and ready to be inflated. Figure 76 shows one of the bladders in an inflated state. The deflation of ech bladders happens automatically through the tubes that were placed in the MDF platform. As soon as the valve is turned off, the bladders deflate through tube leaks. In figure 77 the all the bladders have been glued onto the platform. The next step was to wire the tubes and the commanding brain of the architeture prototype.



[Figure 75] Half of the floor has been glued together. (Author, 2017)



[Figure 76] The inflated state of a bladder. (Author, 2017)



[Figure 77] The bladders are fully glued to the platform. The upper left corner bladders have been connected to the compressor and are inflated. (Author, 2017)

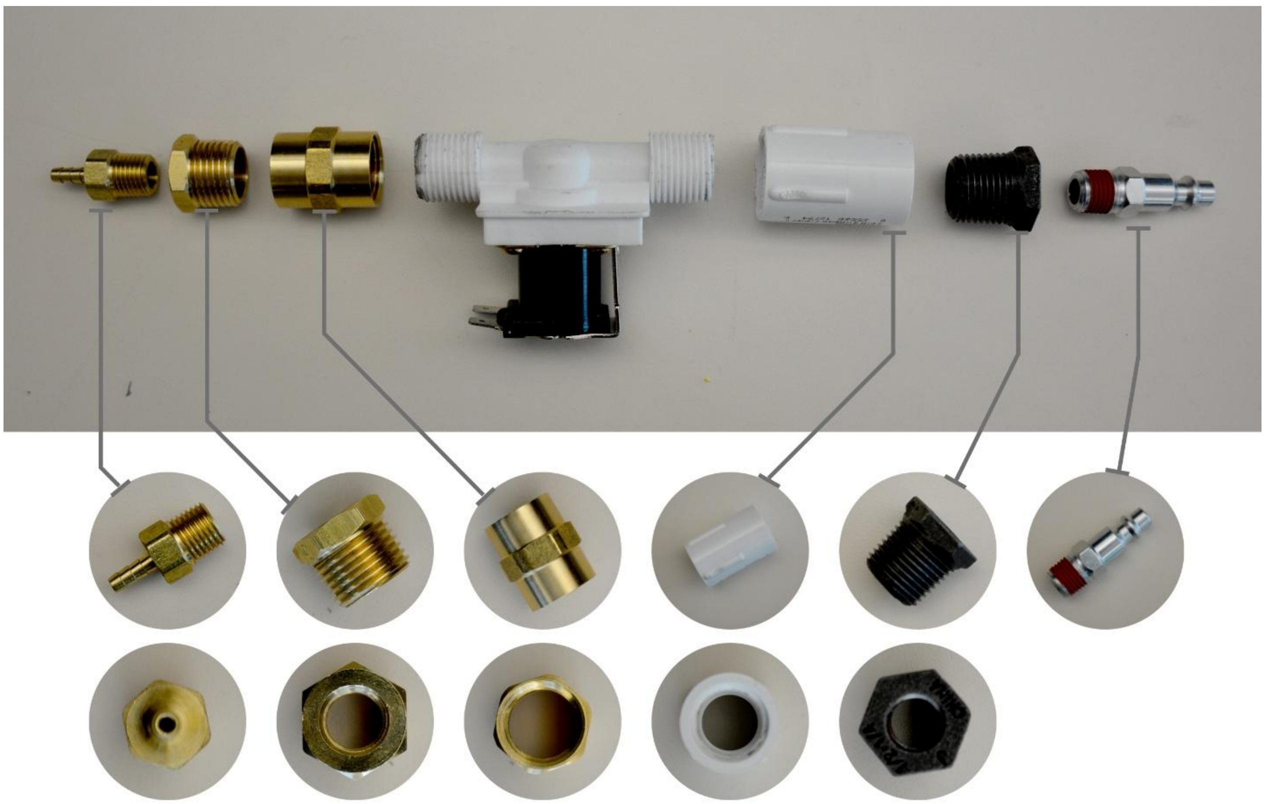
H. Plug-in

Plumming materials and equipments were used to connect the inflatable bladders to the air compressor and the solenoid valves. 300 feet of 1/4 inch tubing were used to connect all the 100 bladders to the four solenoid valves. The solenoids were then connected to the compressor using two ½" air hoses and two compressor branches. Figure78 shows the tubing under the floor. To fit all the tubing and the electronics under the floor, a metal bed was used to raise the floor from the ground. The bed then hid all the tubing beneath itself. The bed raised the floor 2 feet above the ground level.

The challenge facing the tubing was to connect 100 bladders to only four solenoids due to their price and to prevent the complexity of the control mechanism. Hence the bladders were branched in four zones using four manifolds. The manifolds were manually made for the ¼ inch tubing. Figure 79 demonstrates the process for making the manifolds from PVC tubes.

[Figure 78] The tubing under the foor. Each manifold is working for a separate zone and for a single solenoid valve that would be controlled sperately. (Author, 2017)

[Figure 79] The Process for making the manifolds and details for connecting manifolds to the solenoid valves. (Author, 2017)

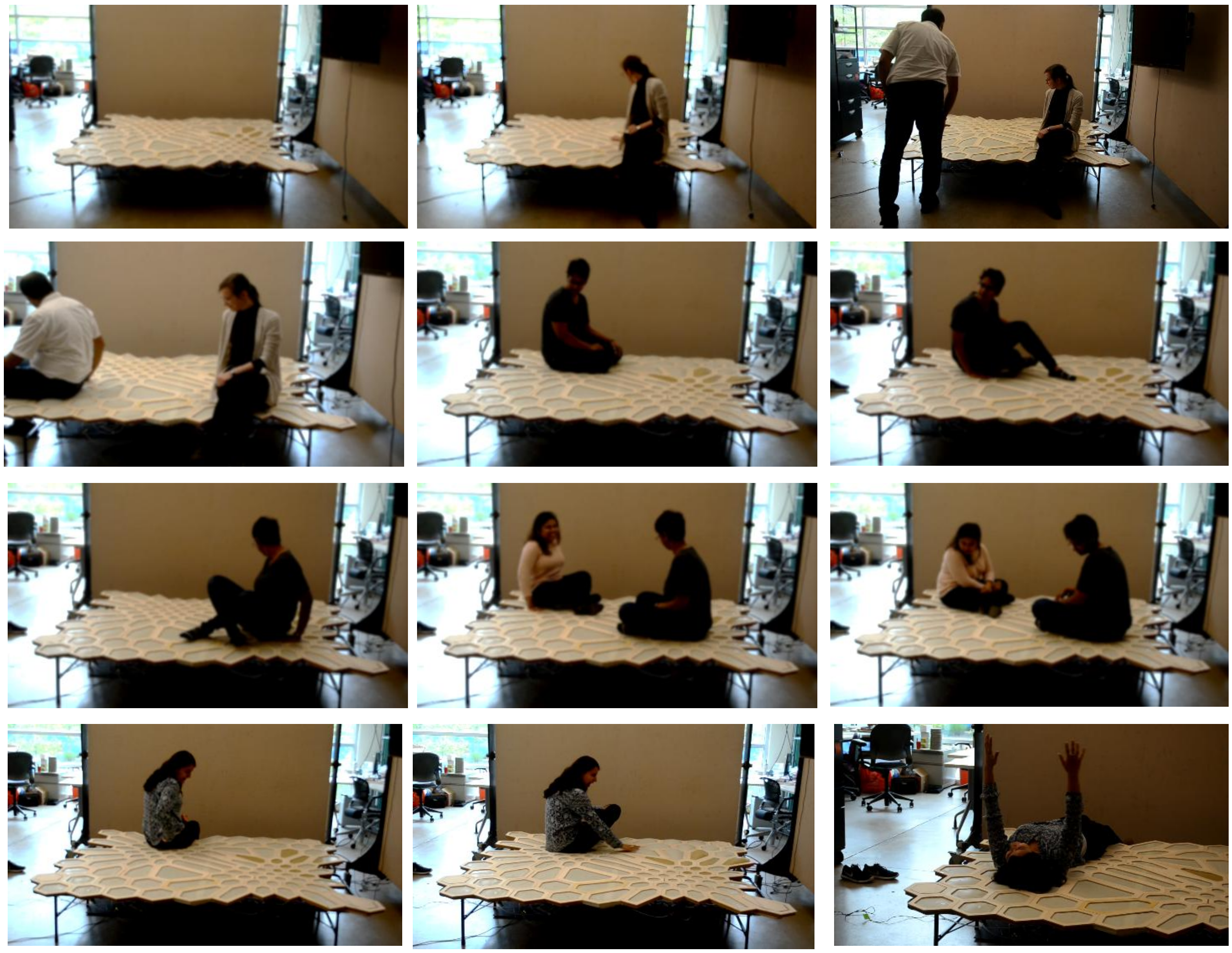


[Figure 80] Each solenoid valve mandated sets of joints to be connected to translate the ½" valve iput/output to ¼" size tubing from one side and to ½" compressor output from another. (Author, 2017)

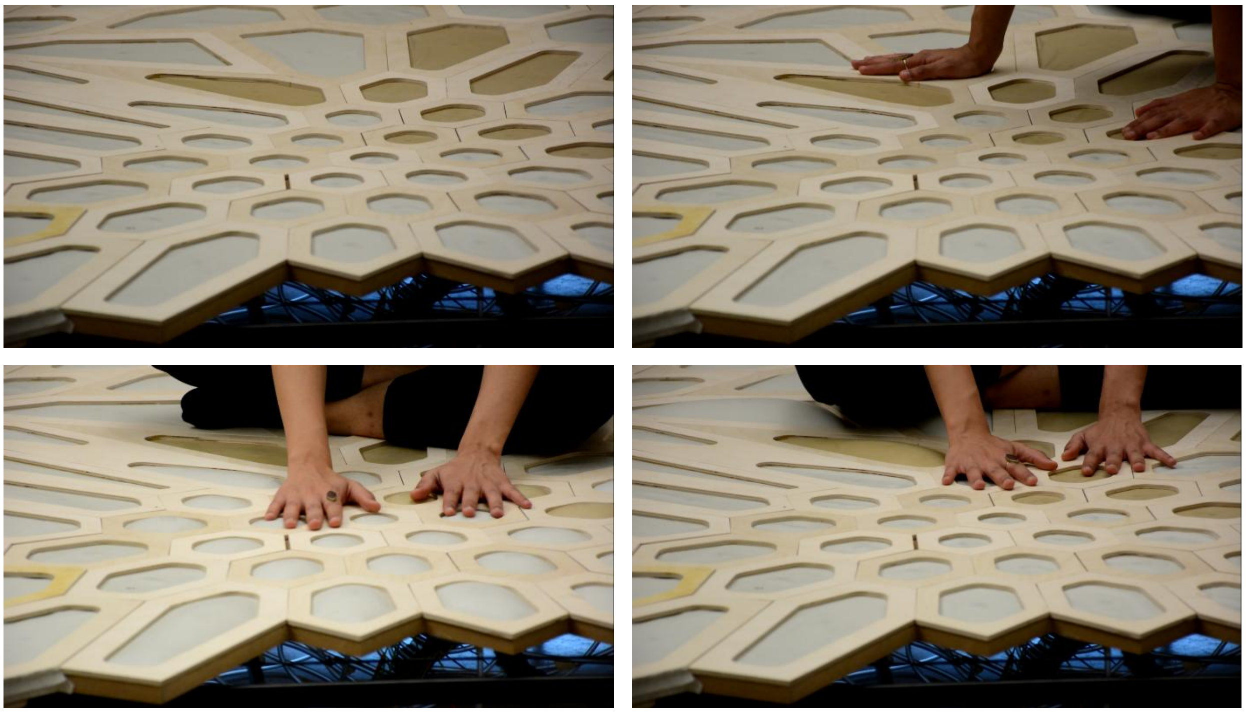
5. THE EPHEMERAL DREAMSPACE: Interaction

The DreamSpace was prototyped in the horizontal plane, in the form of a floor to be seated on by two users. On the day of the thesis defense the floor was installed and tested. Even though the computer code (Processing code) was tested beforehand, some troubles occurred during the installation that caused confusing interaction between the two users and between the users and the prototype. Due to the slow performance of the Kinect and a current unknown code interruption, the Kinect would get confused in detecting two people simultaneously. Future steps would be debugging the code and developing the code into a more complex level for complex interactions. The current code can be found in Appendix C. In this chapter the Processing Depth Image platform, the interaction, and the final installation have been shown. It is hoped that this project will be developed into understanding complex human behaviors in space through bodily movements and postures, as well as responding accordingly to the physical and psychological needs of its users. In a paper by Motalebi et al.[54], the future steps and the future developments have been discussed.

The user's reaction and movements have been demonstrated in figure 81. The prototype acted merely as a proof of concept to the idea of a reactive architecture using inflatable structures. The air pressure used for the final prototype was set to 80 PSI which was split between 100 bladders. This air pressure was not ideal and the reaction would have been more satisfying if the prototype was connected to a stronger air source. For this trial, a portable 5-gallon air compressor with a maximum of 150 PSI was used. Figure 86 depicts how much the bladders inflate in reaction to the user's presence. While the inflate bladders were not big enough to contain any function, they were a physical proof to an idea for a DreamSpace, an ephemeral Virtopia.



[Figure 81] User's behaviors towards the prototype. Some sat alone and some sat together. The prototype was zoned into two sections – right and left – for each user. This was only to make the code simpler and functional. (Author, 2017)



[Figure 82] The prototype's behavior towards user's presence. The bladders inflate and deflate one zone after another in the area where the body is detected. (Author, 2017)

[Figure 83] Single person interaction with the prototype and the prototype's reaction to the user's presence. The different coloring of the latex rubber was only because I was short on latex. (Author, 2017)

6. THE EPHEMERAL DREAMSPACE: Conclusions

This thesis develops a physical space that is influenced and manipulated by user behavior through the use of computational devices. This is motivated by the concept of a DreamSpace – a space in which a collective (in a social sense) architectural element responds to an individuals' movements. The core features of a DreamSpace have been desired by designers and artists in both the 20th and 21st centuries, notably with individuals of the 1960's psychedelic movement. These artists sought to create a new elastic, infinite space, influenced in part through experiences with hallucinogenic drugs. However, technological constraints of the time period and limitations of the physical world prevented the reproduction of their perceived space into reality. Attempts in the 21st century have been made to embody a DreamSpace, but are either limited to the virtual world or do not operate with social context considerations.

The fundamental hypothesis of this thesis is that techniques, technology, and resources have evolved to the point where a DreamSpace can be achieved in reality. More specifically, advanced computational devices, such as controllable valves, shape memory alloys, and servo motors, are now available to alter environments which a greater degree of freedom as compared to the past. In order to test this hypothesis, an ephemeral DreamSpace, one that allows an individual user to manipulate space through the user's behavior in a social context, is required. To embody this, an interactive floor is selected as a suitable candidate. Alternative materials, actuators, sensors, and solution methods are considered and evaluated in order to select the most suitable for the interactive floor's design.

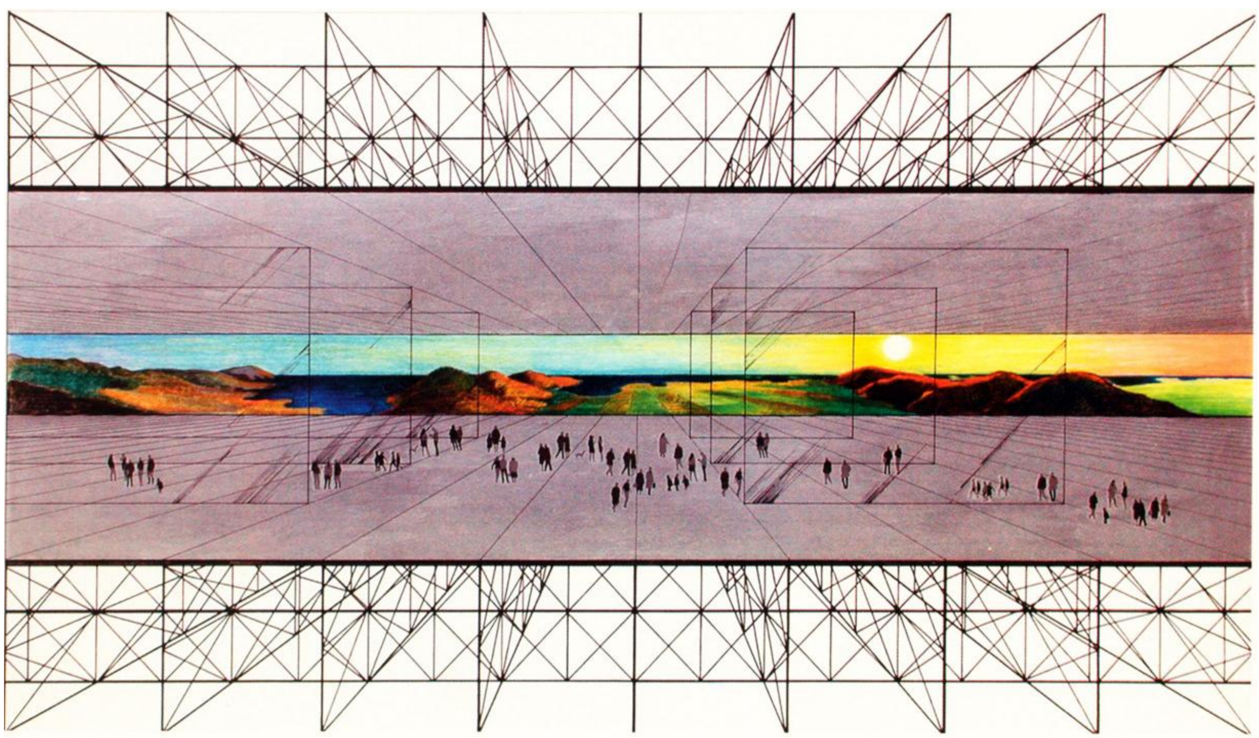
Once final design decisions are made, the step-by-step construction process of a prototype is outlined and the floor is built. The floor consists of modular, inflatable bladders and a design to reflect the positions of two potential users through existing attraction points. A Kinect sensor is used to detect the position of both users, and solenoid valves and a compressor are manipulated, using a microprocessor for control, to inflate or deflate different sections of the floor based on these positions.

The final interactive floor prototype proves the validity of the aforementioned hypothesis. The prototype exists as a proof-of-concept of a DreamSpace in reality through the use of computational devices. The designed space is able to adjust itself based on the detected positions of the users interacting in the environment, rather than a user manually inputting commands to have the space meet their needs. This adjustment is based on a collective behavior of users, rather than just a single individual, meeting the criteria of a DreamSpace.

APPENDIX A: No-Stop City, SuperSurface, New Babylon, The Psychedelics, Crashpads, Phantasy Landscape

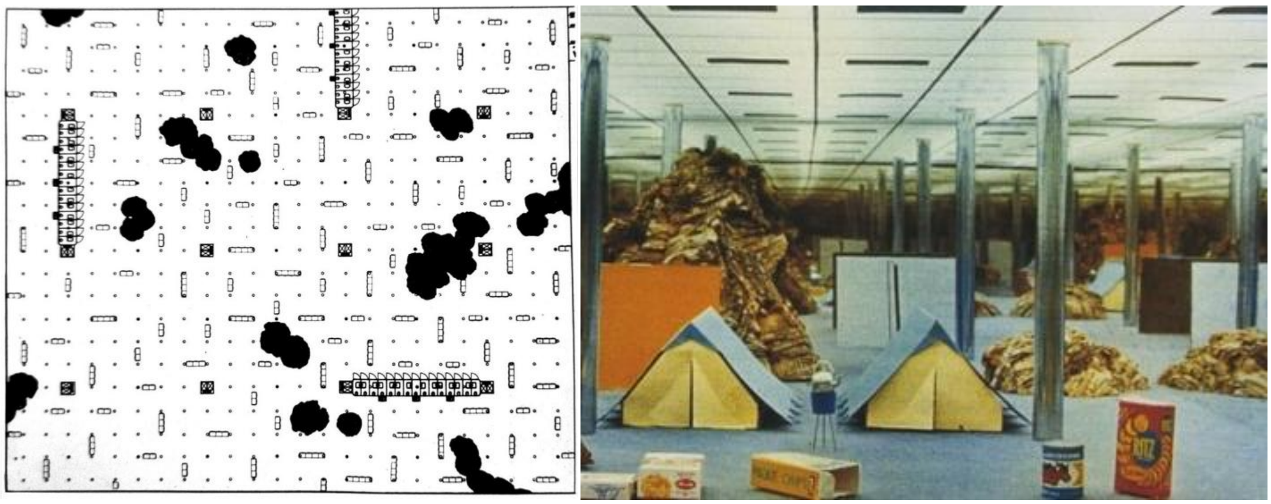
The No-Stop City

The Italy of the 1960s gave birth to the radical architects who were idealizing the breakdown of trivialized functions in architectural space. In 1964, Andrea Barnzi and a few other Italian architects formed the Archizoom group. As they conceptualized what ideal architectural space should look like, they proposed the No-Stop City as a Utopia that frees its inhabitants from the greyscale industrial architecture. The No-Stop City was blank, featureless, and anti-decorative architecture. The City was to allow people to be anyone and anywhere along the infinite space that stretched towards the horizon.



[Figure 84] No-Stop City by Archizoom. https://architizer.com/blog/archizoom-retrospective/

The No-stop City is an unbuilt project. The city in an infinite grid that is subdivided by lines as the walls and are interrupted by natural elements such as mountains. The city as an artificial world is then filled with pieces of nature. People, now living like nomads, can reside anywhere among these natural elements.

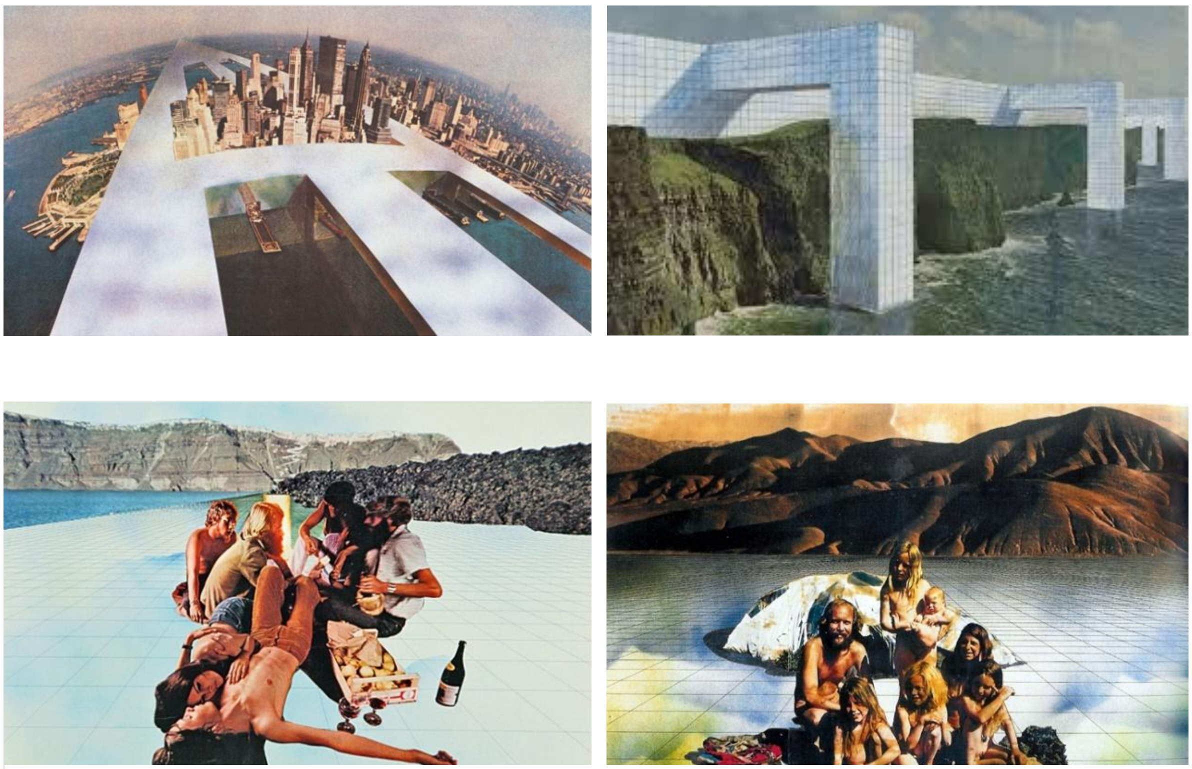


[Figure 85] No-Stop City Interior Landscape by Archizoom, 1969. https://architizer.com/blog/archizoom-retrospective/

SuperSurface

In 1966, Adolfo Natalini and Cristiano Toraldo di Francia founded an Architectural firm called Superstudio. As part of the radical movement of the 1960s in Italy, the Superstudio presented multiple models for “Total Urbanization” where they used anti-architectural motifs to criticize the blank architecture of the modern era. They attacked the mass production of architecture and architectural elements whereas, consumerism that has imposed a monotonous lifestyle upon individuals.

By conceptualizing a negative Utopia that overtakes the modern cities, SuperStudio tried to warn against homogeneity. In oppose to the modern Utopia, they tried to stand up for individual expressions in space. In their works “Continuous Monument” and “Supersurface” they suggest that “everything could be replaced by a continuous global grid” that represents the modern era. Superstudio then manifests architecture based on the freedom of Vita (life); “human beings are the only creators of their own choices. They can free themselves from induced needs and behaviors, pick their own place, everywhere on the earth's Supersurface"[55]. The Supersurface then acts as a "potential device"[55] instead of a placeholder for objects. It is a world without objects, "an alternative model of life on earth"[55].



[Figure 86] Supersurface by SuperStudio 1966. http://www.architectureplayer.com/clips/supersurface-an-alternative-model-for-life-on-the-earth

New Babylon

New Babylon was designed by Constant Nieuwenhuys, an artist who worked with the Situationists as their architect back in the 1960s. The Situationists were a group of intellectual activists who tried to revolutionize the social constructs of the Modern capitalism of the time through the use of the capitalist system itself. Guy Debord, the founding member of the Situationists, invented strategies such as the *derive*, *psychogeography*, and *detournment* as radical ways of living in the city as an impost structure. While all the three are mostly concerned with disorientation in space, personal and individual perception of space, and random encounters, Constant tried to manifest these characteristics in a spatial design proposal.

The New Babylon was designed to be an unpredictable structure that could be endlessly woven to provide new and ever-changing personal experiences in space. The structure was to be liberating and self-inventing in response to the endless desires of individuals of space. The Babylon was then developed as a massive steel structure that acted in a monumental scale. The steel represented the modern technology that was available to be exploited and the Situationists, such as constant, used it to attack Modern capitalist ideologies.

[Figure 87] New Babylon by Constant Nieuwenhuys.
https://lebbeuswoods.wordpress.com/2009/10/19/constant-vision/

The emergence of Post-Modernism however, marginalized the utopian city introduced by the Situationists. The early 60s was the end of the beautiful machines and industrial technologies as Post-Modernism retrograded towards the values of culture and imagery over structure. Therefore, the presentation of technology as a window to Marxism and liberation from authoritarian systems was disturbed. Leaving what was once envisioned unresolved, the solution then shifted towards understanding the origins of life and practicing primitive lifestyles, experience spirituality, and develop their individual perception of life through a recess of their inner space. Such practices got in line with the psychedelic movements that perception was shifted due to the popularity of LSD and hallucinogen drugs. As Timothy Leary and others advocated for the consumption of LSD, they opened up a window of hope among the young generation for liberation from the conventions of the time.

The Psychedelics

In 1943, after his first bicycle ride home under a small doze of a psychoactive substance, Albert Hofmann knew he had made a significant discovery. The "Bicycle Day", known as the first LSD trip, was the beginning of an extraordinary experience for the hippie youths back in the 1960s. Even though hallucinogens were used in other tribal cultures for spiritual practices, it was only recently discovered in the West. In 1963, Timothy Leary, an American psychologist advocated for the use of LSDs. As LSDs were tested among the members of the general public for their psychiatric purposes, they became known to the youth as substances that could shift their consciousness and perception of space. By 1966, the use of LSDs became legal in California. As LSDs were known for their psychedelic, kaleidoscopic experience, they became the ticket to a spiritual journey and away from the materialistic world.

The mid 60s was the peak of the psychedelic movement. Not only it had shifted perceptual awareness of those who were influenced by psychedelic drugs, but it had made a cultural turn that was aligned with the Post-Modern, anti-authoritarian movement. The psychedelic movement then was known as a counterculture lifestyle that had influenced art, design, clothing, and so on. The feeling of liberation was extended to the living space and now houses were not necessarily homes to the hippies.

The psychedelics envisioned spaces that blurred the boundaries between reality and illusion. Their extraordinary experiences from LSD – perhaps another type of virtual world – demanded and inspired extra-ordinary settings. The people of the psychedelics were hypnotized by the idea of re-living their dreams in the material world. Alastair Gordon talks of the 1960's psychedelics whose drug-altered minds were driven to imagining liquefied simulations of the material world, spaces that were flowing; non-hierarchical environments in a constant state of mutation. This shared dream led the psychedelics towards shaping a community that would be empathetic towards unconventional spaces.

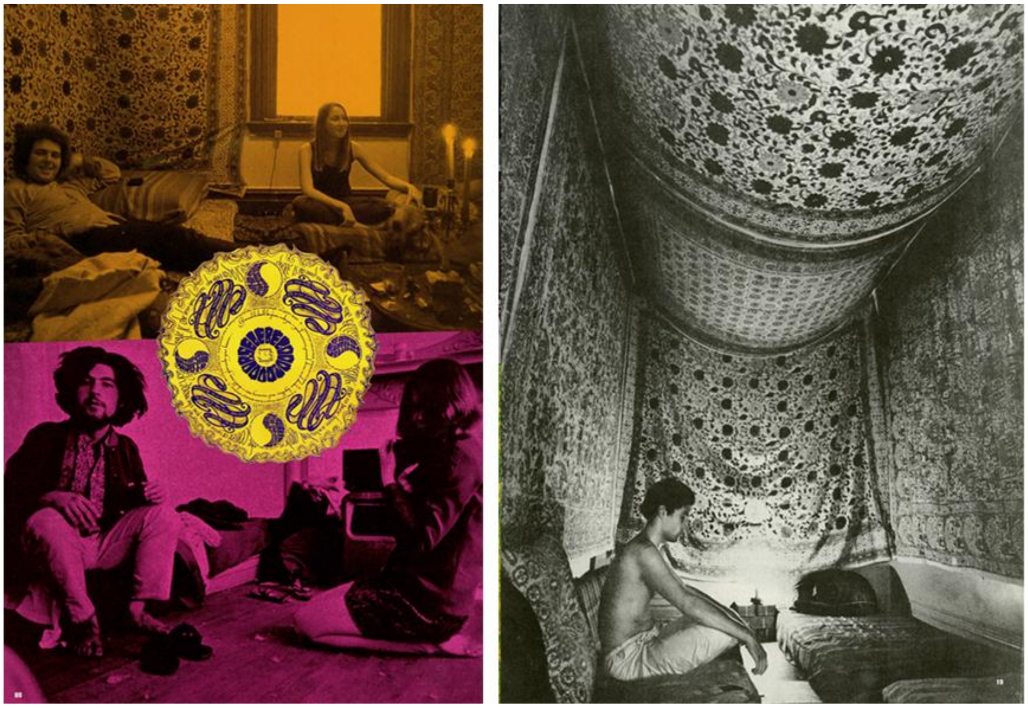
Unconventional, in the 1960's, stemming from the anti-authoritarian Marxism[56], meant redefining Marxism back to its roots as a way against Capitalism at the time. As the Situationist International[57] made their philosophical criticism through writing and city planning, the psychedelics spread their words through arts, design and lifestyle. Rooting back to Europe's Existentialism, unconventional was to give back the power to the human individual who experiences life beyond a thinking subject but as a human subject that acts, feels, and lives. Trying to reflect the individual perception of space and personal experiences from the "Inner Space" onto the physical world, the psychedelics started a liberated lifestyle. Focusing on what Superstudio called Vita or life, the psychedelic youths around New York and California created the communes where communal life happened in houses with "no objects" or furniture but bed matts. These mattresses were then characterized as "CrashPads" where, life, love, sleep and so on all happened in the same place.

Crash Pads

A revolution was underway. In 1966, it was estimated that one million Americans had tried LSD.[58] The hippies who believed in free love and communal families had become a stereotype that limited their lifestyle in fear of a soul destroying world. At this time, a new relationship to space was defined which was allegorical, embracing, and non-hierarchical.

For the psychedelics, space or the idea of space was to be elastic, acrobatic, infinite, and ephemeral; the old measurements of space were ridiculed. They wanted spaces that could be anything and spaces that could fit any function within itself. Crash Pads were early examples of space definers that embraced different activities onto itself. Due to the high expenses of New York City, Crash Pads became the cheap and simple substitutions for everyday communal home that was in line with their ideologies of space.

[Figure 88] CrashPads. Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008).



[Figure89] Hippie counterculture. Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008).

Such communal spaces plus the psychedelic experiences of space, impacted how designers approached architecture and defined spatial experience. It seemed as if designers tried to give life to a spatial manifestation that was shared among the psychedelics. The followers of the anti-authoritarian movements wanted to re-build the physical world so to relive their hallucinations in a communal sense, share their experience with others, and empathize with their community for the dream of life, space, and architecture.

Phantasy Landscape

Phantasy Landscape was a materialization of a recurring dream of a mother's womb.[59] As a symbol to pre-birth life, Verner Panton's idea was to break the conventional architecture into a spectacle of bizarre scenery. His aim was to create curvilinear lines, altered scales, disorientation, illusion, and most importantly, repurposing the ground plane for an infantile relationship with the universe. Walls, ceilings and floors started to merge into a single space where all activities could be done anywhere on anything. The Phantasy Landscape was a projection of Panton's fantasies that other psychedelics could crawl into and live the dream without any hallucinogens.

[Figure 90] Phantasy Landscape by Verner Panton. http://www.verner-panton.com/spaces/archive/121/

Panton's work visualized an extraordinary experience of color, curvilinear shapes, and life style; spaces were shaped based on their proximity to the body itself. Based on a psychological study by Dazkir et al., "curvilinear settings elicited higher amounts of pleasant-unarousing emotions (such as feeling relaxed, peaceful, and calm) compared to rectilinear settings."[60] The sense of pleasure in the Phantasy Landscape was to provide a relaxing sensation to the people who were frustrated by the social constructs of the time. They would sit, crawl, sleep and play within the space as if it was one with the body; architecture as the body's second skin.[61] As Witold Rybczynski[62] reminds us that there is nothing natural about sitting on a chair, the psychedelics tried to free themselves from the conventional lifestyles by reaching back to the roots of position the body in relation to the horizontal plane and the floor.

APPENDIX B: Floor Living

The archetypical postures of floor living, those that the body sits down and gets close to the ground, liberates human activity from the conventional ways of settlement in space in the horizontal (ground) level. The culture of floor living, considered as an archaic behavior, is known as a way to promote social activities and communal life where each individual is free to act and behave as they desire. In this regard, this thesis develops a computationally mediated floor as a design concept for the Dreamspace that is infinitely expandable on the horizontal level, reacts appropriately to the individual behavior on it, is an evocative and playful platform for shared activities, and is socially significant to the architectural space. Further on, the concept of floor living and its cultural significance has been elaborated in order to specify the design influence for this project.

"At rest we assume natural archetypical postures"[63] in a sense that the postures of resting and sitting on the floor are our birthright. The modern society has mostly neglected this right by developing furniture that does not support such natural postures. Even though such furniture has helped the body to erect from the sitting position, the loses of not sitting on the ground should be noticed. Based on a study by Gordon Hewes, there are more than one hundred common sitting positions around the world. "A fourth of mankind habitually squats in a fashion very similar to the squatting position of chimpanzee, and the rest of us might squat this way too if we were not trained to use other postures beyond infancy."[64] Even though the tone of this statement implies lack of postural education for those who sit on the floor, Rybzynski confirms that it is not a matter of class, nor technology that lead societies towards sitting on chairs.[65] While it is still unknown why some societies develop onto chair sitting, "after 5000 years of chair sitting history, there is nothing natural about sitting on chairs."[66]

[Figure 91] An adaptation of world distribution of Postural habits by Hewes in 1955. This diagram shows the existing postures of sitting on the ground and how they are spread in the globe.

In the book "Now I Sit Me Down", Ryczynski[67] reiterates that there are so many different reasons to sit, hence there are different types of chairs to fulfill those reasons. If so, what are the cultural and social reasons behind sitting on the floor? Is there a difference in how people communicate and interact in the floor living cultures than those who do not? And finally, does floor living promote social communication and interaction in such countries? It is needless to say that each morphology (type) of rest affects other aspects of one's life, from clothing to architecture, and architectural elements (windows, knobs, and so on) [Figure 92]. But most importantly for this design proposal is the social impacts of floor living on communication in an architectural setting.

[Figure 92] "An Indian woman making bread. Her shelves are at low height, which is common in floor-sitting cultures." (Mansi Thapliyal / Reuters)
https://www.theatlantic.com/international/archive/2016/08/chairs-history-witold-rybczynski/497657/

Rybczynski also talks of the act of sitting on the ground as a way to express anger and resistance towards unpleasant situations. An example of such behavior was the 99% movement when people sat down and occupied Wall Street for days against social and economic inequality [Figure 93]. As a means to non-verbal communication, sitting on the ground has become a coded behavior that implies different meanings in different settings. In a culture that most people sit in an upper level from the ground, sitting on the floor would be seen as an act against the conventions and cultural rules of the society. In such societies, sitting on the floor could become an instigator for social bonding and communication among those who have mutual interests. Hence, movements such as the Post-Modern Avant Grades, used floor living as a counterculture against capitalism and spiritual decay.

[Figure 93] Protesters sit in Freedom Plaza, forming a human "99%" shape during Occupy DC in Washington, on Thursday, October 6, 2011. (AP Photo/Jacquelyn Martin)

From the article "Occupy Wall Street spreads beyond NYC", Accessed February 2017. URL: https://www.theatlantic.com/photo/2011/10/occupy-wall-street-spreads-beyond-nyc/100165/

Respectively, floor living as an unconventional way of settlement and rest in the United States was chosen as the paradigm for the design of the actuated architecture. A floor that reacts to people's way of sitting as to initiate an evocative experience in space through its design and social encounters on itself. The next is to define the design details of such a floor and material exploration that lead to the design of an ephemeral Dreamspace.

APPENDIX C: Center of Mass Kinect/Processing Code

```
import SimpleOpenNI.*;
SimpleOpenNI context;

import processing.serial.*;

import cc.arduino.*;
Arduino arduino;

int counter = 0 ;
long prevNumUsers = 0;
long currentNumUsers = 0;
float [] posX = new float[2];
float [] posY = new float[2];
float [] posZ = new float[2];

int solenoidPin1 = 2;
int solenoidPin2 = 4;
int solenoidPin3 = 7;
int solenoidPin4 = 8;


//set 1
int A = 4000; //the time it takes to inflate a bladder
int A2 = A + 8000;
int B = A2 + 4000;
int B2 = B + 8000;
int C = B2 + 4000; //time to deflate
int C2 = C + 8000;
int D = C2 + 4000;
int D2 = D + 8000;

float currentTime = 0;
float previousTime= millis();
float t = 0;


void setup() {
  size(1600, 900);

  println(Arduino.list());

  arduino = new Arduino(this, Arduino.list()[0], 57600);
  arduino.pinMode(2, Arduino.OUTPUT);
  arduino.pinMode(4, Arduino.OUTPUT);
  arduino.pinMode(7, Arduino.OUTPUT);
  arduino.pinMode(8, Arduino.OUTPUT);


  arduino.digitalWrite(solenoidPin1, Arduino.LOW);
  arduino.digitalWrite(solenoidPin2, Arduino.LOW);
  arduino.digitalWrite(solenoidPin3, Arduino.LOW);
```

```
  arduino.digitalWrite(solenoidPin4, Arduino.LOW);
  delay(5000);


  context = new SimpleOpenNI(this);
  if (context.isInit() == false) {
    println("Can't initialize SimpleOpenNI, camera not connected properly.");
    exit();
    return;
  }

  context.enableDepth();
  context.enableUser();
  context.setMirror(false);
  background(0);
  stroke(0, 0, 255);
  strokeWeight(3);
  smooth();
}


void draw() {

  currentTime = millis ();
  t = currentTime - previousTime;

  context.update();
  scale (2);

  image(context.depthImage(), 0, 0);

  IntVector userList = new IntVector();

  context.getUsers(userList);

    for (int i=0; i<userList.size (); i++) {
      int userId = userList.get(i);
      PVector position = new PVector();

      context.getCoM(userId, position);
      context.convertRealWorldToProjective(position, position);
      fill(255, 0, 0);

      textSize(40);
      text(userId, position.x, position.y);
      text(position.x, position.x, 100);
      text(position.y, position.x, 150);
      text(position.z, position.x, 200);

      posX[i] = position.x;
      posY[i] = position.y;
      posZ[i] = position.z;
```

```
  //println( userId, position.x, position.y, position.z);
}


currentNumUsers = userList.size(); // The current # of users.


if (currentNumUsers != prevNumUsers) {
  // deflate completely.
  arduino.digitalWrite(solenoidPin1, Arduino.LOW);
  arduino.digitalWrite(solenoidPin2, Arduino.LOW);
  arduino.digitalWrite(solenoidPin3, Arduino.LOW);
  arduino.digitalWrite(solenoidPin4, Arduino.LOW);
  delay (100); // delay for time it takes to completely deflate.
  previousTime = currentTime;
}
prevNumUsers = currentNumUsers; // set previous # of users = to current # of users.

if (currentNumUsers == 0) {

  //println(currentNumUsers,millis(),"default movement"); // This is for no users. Do the
default movement.

  if ( t < A) {
    arduino.digitalWrite(solenoidPin1, Arduino.HIGH);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 1 Inflate");
  }

  if (t >= A && t < A2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 1 Deflate");
  }

  if (t >= A2 && t < B) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.HIGH);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 2 Inflate");
  }

  if (t >= B && t < B2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 2 Deflate");
```

```
  }

  if (t >= B2 && t < C) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.HIGH);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 3 Inflate");
  }

  if (t >= C && t < C2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 3 Deflate");
  }

  if (t >= C2 && t < D) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.HIGH);
    println(currentNumUsers, millis(), "Sec. 4 Inflate");
  }

  if (t >= D && t < D2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 4 Deflate");
  }

  if (t >= D2 ) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    previousTime = currentTime;
    println(currentNumUsers, millis(), "Reset default movement");
  }
} else if (currentNumUsers == 1) {
  if (posX[0] < 280 && posY[0] > 170) {
    // half one (sec 2 and 4 ) moves

    if ( t < A) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.HIGH);
      arduino.digitalWrite(solenoidPin3, Arduino.LOW);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 2 Inflate");
    }
```

```
  if (t >= A && t < A2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 2 Deflate");
  }

  if (t >= A2 && t < B) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.HIGH);
    println(currentNumUsers, millis(), "Sec. 4 Inflate");
  }

  if (t >= B && t < B2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 4 Deflate");
  }

  if (t >= B2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    previousTime = currentTime;
    println(currentNumUsers, millis(), "RESET");
  }
} else if (posX[0] > 280 && posY[0] > 170 ) {
  // half 2 (sec 1 & 3) moves

  if ( t < A) {
    arduino.digitalWrite(solenoidPin1, Arduino.HIGH);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 1 Inflate");
  }

  if (t >= A && t < A2) {
    arduino.digitalWrite(solenoidPin1, Arduino.LOW);
    arduino.digitalWrite(solenoidPin2, Arduino.LOW);
    arduino.digitalWrite(solenoidPin3, Arduino.LOW);
    arduino.digitalWrite(solenoidPin4, Arduino.LOW);
    println(currentNumUsers, millis(), "Sec. 1 Deflate");
  }

  if (t >= A2 && t < B) {
```

```
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.LOW);
      arduino.digitalWrite(solenoidPin3, Arduino.HIGH);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 3 Inflate");
    }

    if (t >= B && t < B2) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.LOW);
      arduino.digitalWrite(solenoidPin3, Arduino.LOW);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 3 Deflate");
    }

    if (t >= B2) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.LOW);
      arduino.digitalWrite(solenoidPin3, Arduino.LOW);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "RESET");
      previousTime = currentTime;
    }
  }
} else if (currentNumUsers >= 2) {

  if ( posY[0] > 170 && posY[1] > 170) {
    // section 2&3 moves
    if ( t < A) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.HIGH);
      arduino.digitalWrite(solenoidPin3, Arduino.LOW);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 1 Inflate");
    }

    if (t >= A && t < A2) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.LOW);
      arduino.digitalWrite(solenoidPin3, Arduino.LOW);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 1 Deflate");
    }

    if (t >= A2 && t < B) {
      arduino.digitalWrite(solenoidPin1, Arduino.LOW);
      arduino.digitalWrite(solenoidPin2, Arduino.LOW);
      arduino.digitalWrite(solenoidPin3, Arduino.HIGH);
      arduino.digitalWrite(solenoidPin4, Arduino.LOW);
      println(currentNumUsers, millis(), "Sec. 4 Inflate");
    }

    if (t >= B && t < B2) {
```

```
        arduino.digitalWrite(solenoidPin1, Arduino.LOW);
        arduino.digitalWrite(solenoidPin2, Arduino.LOW);
        arduino.digitalWrite(solenoidPin3, Arduino.LOW);
        arduino.digitalWrite(solenoidPin4, Arduino.LOW);
        println(currentNumUsers, millis(), "Sec. 4 Deflate");
      }

      if (t >= B2) {
        arduino.digitalWrite(solenoidPin1, Arduino.LOW);
        arduino.digitalWrite(solenoidPin2, Arduino.LOW);
        arduino.digitalWrite(solenoidPin3, Arduino.LOW);
        arduino.digitalWrite(solenoidPin4, Arduino.LOW);
        println(currentNumUsers, millis(), "RESET");
        previousTime = currentTime;
      }
    }
  }
}
```

APPENDIX D: Arduino "Standardfatirama" code for connecting processing commands to Arduino output

```
/*
  Firmata is a generic protocol for communicating with microcontrollers
  from software on a host computer. It is intended to work with
  any host computer software package.

  To download a host software package, please clink on the following link
  to open the download page in your default browser.

  https://github.com/firmata/arduino#firmata-client-libraries

  Copyright (C) 2006-2008 Hans-Christoph Steiner.  All rights reserved.
  Copyright (C) 2010-2011 Paul Stoffregen.  All rights reserved.
  Copyright (C) 2009 Shigeru Kobayashi.  All rights reserved.
  Copyright (C) 2009-2015 Jeff Hoefs.  All rights reserved.

  This library is free software; you can redistribute it and/or
  modify it under the terms of the GNU Lesser General Public
  License as published by the Free Software Foundation; either
  version 2.1 of the License, or (at your option) any later version.

  See file LICENSE.txt for further informations on licensing terms.

  Last updated by Jeff Hoefs: April 11, 2015
*/

#include <Servo.h>
#include <Wire.h>
#include <Firmata.h>

#define I2C_WRITE                   B00000000
#define I2C_READ                    B00001000
#define I2C_READ_CONTINUOUSLY       B00010000
#define I2C_STOP_READING            B00011000
#define I2C_READ_WRITE_MODE_MASK    B00011000
#define I2C_10BIT_ADDRESS_MODE_MASK B00100000
#define MAX_QUERIES                 8
#define REGISTER_NOT_SPECIFIED      -1

// the minimum interval for sampling analog input
#define MINIMUM_SAMPLING_INTERVAL 10


/*==============================================================================
 * GLOBAL VARIABLES
 *============================================================================*/
```

```
/* analog inputs */
int analogInputsToReport = 0; // bitwise array to store pin reporting

/* digital input ports */
byte reportPINs[TOTAL_PORTS];       // 1 = report this port, 0 = silence
byte previousPINs[TOTAL_PORTS];     // previous 8 bits sent

/* pins configuration */
byte pinConfig[TOTAL_PINS];         // configuration of every pin
byte portConfigInputs[TOTAL_PORTS]; // each bit: 1 = pin in INPUT, 0 = anything else
int pinState[TOTAL_PINS];           // any value that has been written

/* timer variables */
unsigned long currentMillis;        // store the current value from millis()
unsigned long previousMillis;       // for comparison with currentMillis
unsigned int samplingInterval = 19; // how often to run the main loop (in ms)

/* i2c data */
struct i2c_device_info {
  byte addr;
  int reg;
  byte bytes;
};

/* for i2c read continuous more */
i2c_device_info query[MAX_QUERIES];

byte i2cRxData[32];
boolean isI2CEnabled = false;
signed char queryIndex = -1;
// default delay time between i2c read request and Wire.requestFrom()
unsigned int i2cReadDelayTime = 0;

Servo servos[MAX_SERVOS];
byte servoPinMap[TOTAL_PINS];
byte detachedServos[MAX_SERVOS];
byte detachedServoCount = 0;
byte servoCount = 0;

boolean isResetting = false;

/* utility functions */
void wireWrite(byte data)
{
#if ARDUINO >= 100
  Wire.write((byte)data);
#else
  Wire.send(data);
#endif
}

byte wireRead(void)
```

```
{
#if ARDUINO >= 100
  return Wire.read();
#else
  return Wire.receive();
#endif
}

/*==============================================================================
===========
 * FUNCTIONS
 
*==============================================================================
==*/

void attachServo(byte pin, int minPulse, int maxPulse)
{
  if (servoCount < MAX_SERVOS) {
    // reuse indexes of detached servos until all have been reallocated
    if (detachedServoCount > 0) {
      servoPinMap[pin] = detachedServos[detachedServoCount - 1];
      if (detachedServoCount > 0) detachedServoCount--;
    } else {
      servoPinMap[pin] = servoCount;
      servoCount++;
    }
    if (minPulse > 0 && maxPulse > 0) {
      servos[servoPinMap[pin]].attach(PIN_TO_DIGITAL(pin), minPulse, maxPulse);
    } else {
      servos[servoPinMap[pin]].attach(PIN_TO_DIGITAL(pin));
    }
  } else {
    Firmata.sendString("Max servos attached");
  }
}

void detachServo(byte pin)
{
  servos[servoPinMap[pin]].detach();
  // if we're detaching the last servo, decrement the count
  // otherwise store the index of the detached servo
  if (servoPinMap[pin] == servoCount && servoCount > 0) {
    servoCount--;
  } else if (servoCount > 0) {
    // keep track of detached servos because we want to reuse their indexes
    // before incrementing the count of attached servos
    detachedServoCount++;
    detachedServos[detachedServoCount - 1] = servoPinMap[pin];
  }

  servoPinMap[pin] = 255;
}
```

```
void readAndReportData(byte address, int theRegister, byte numBytes) {
  // allow I2C requests that don't require a register read
  // for example, some devices using an interrupt pin to signify new data available
  // do not always require the register read so upon interrupt you call Wire.requestFrom()
  if (theRegister != REGISTER_NOT_SPECIFIED) {
    Wire.beginTransmission(address);
    wireWrite((byte)theRegister);
    Wire.endTransmission();
    // do not set a value of 0
    if (i2cReadDelayTime > 0) {
      // delay is necessary for some devices such as WiiNunchuck
      delayMicroseconds(i2cReadDelayTime);
    }
  } else {
    theRegister = 0;  // fill the register with a dummy value
  }

  Wire.requestFrom(address, numBytes);  // all bytes are returned in requestFrom

  // check to be sure correct number of bytes were returned by slave
  if (numBytes < Wire.available()) {
    Firmata.sendString("I2C: Too many bytes received");
  } else if (numBytes > Wire.available()) {
    Firmata.sendString("I2C: Too few bytes received");
  }

  i2cRxData[0] = address;
  i2cRxData[1] = theRegister;

  for (int i = 0; i < numBytes && Wire.available(); i++) {
    i2cRxData[2 + i] = wireRead();
  }

  // send slave address, register and received bytes
  Firmata.sendSysex(SYSEX_I2C_REPLY, numBytes + 2, i2cRxData);
}

void outputPort(byte portNumber, byte portValue, byte forceSend)
{
  // pins not configured as INPUT are cleared to zeros
  portValue = portValue & portConfigInputs[portNumber];
  // only send if the value is different than previously sent
  if (forceSend || previousPINs[portNumber] != portValue) {
    Firmata.sendDigitalPort(portNumber, portValue);
    previousPINs[portNumber] = portValue;
  }
}

/* ------------------------------------------------------------------------------
 * check all the active digital inputs for change of state, then add any events
 * to the Serial output queue using Serial.print() */
void checkDigitalInputs(void)
{
```

```
  /* Using non-looping code allows constants to be given to readPort().
   * The compiler will apply substantial optimizations if the inputs
   * to readPort() are compile-time constants. */
  if (TOTAL_PORTS > 0 && reportPINs[0]) outputPort(0, readPort(0, portConfigInputs[0]), false);
  if (TOTAL_PORTS > 1 && reportPINs[1]) outputPort(1, readPort(1, portConfigInputs[1]), false);
  if (TOTAL_PORTS > 2 && reportPINs[2]) outputPort(2, readPort(2, portConfigInputs[2]), false);
  if (TOTAL_PORTS > 3 && reportPINs[3]) outputPort(3, readPort(3, portConfigInputs[3]), false);
  if (TOTAL_PORTS > 4 && reportPINs[4]) outputPort(4, readPort(4, portConfigInputs[4]), false);
  if (TOTAL_PORTS > 5 && reportPINs[5]) outputPort(5, readPort(5, portConfigInputs[5]), false);
  if (TOTAL_PORTS > 6 && reportPINs[6]) outputPort(6, readPort(6, portConfigInputs[6]), false);
  if (TOTAL_PORTS > 7 && reportPINs[7]) outputPort(7, readPort(7, portConfigInputs[7]), false);
  if (TOTAL_PORTS > 8 && reportPINs[8]) outputPort(8, readPort(8, portConfigInputs[8]), false);
  if (TOTAL_PORTS > 9 && reportPINs[9]) outputPort(9, readPort(9, portConfigInputs[9]), false);
  if (TOTAL_PORTS > 10 && reportPINs[10]) outputPort(10, readPort(10, portConfigInputs[10]), false);
  if (TOTAL_PORTS > 11 && reportPINs[11]) outputPort(11, readPort(11, portConfigInputs[11]), false);
  if (TOTAL_PORTS > 12 && reportPINs[12]) outputPort(12, readPort(12, portConfigInputs[12]), false);
  if (TOTAL_PORTS > 13 && reportPINs[13]) outputPort(13, readPort(13, portConfigInputs[13]), false);
  if (TOTAL_PORTS > 14 && reportPINs[14]) outputPort(14, readPort(14, portConfigInputs[14]), false);
  if (TOTAL_PORTS > 15 && reportPINs[15]) outputPort(15, readPort(15, portConfigInputs[15]), false);
}

// -----------------------------------------------------------------------------
/* sets the pin mode to the correct state and sets the relevant bits in the
 * two bit-arrays that track Digital I/O and PWM status
 */
void setPinModeCallback(byte pin, int mode)
{
  if (pinConfig[pin] == IGNORE)
    return;

  if (pinConfig[pin] == I2C && isI2CEnabled && mode != I2C) {
    // disable i2c so pins can be used for other functions
    // the following if statements should reconfigure the pins properly
    disableI2CPins();
  }
  if (IS_PIN_DIGITAL(pin) && mode != SERVO) {
    if (servoPinMap[pin] < MAX_SERVOS && servos[servoPinMap[pin]].attached()) {
```

```
          detachServo(pin);
        }
      }
      if (IS_PIN_ANALOG(pin)) {
        reportAnalogCallback(PIN_TO_ANALOG(pin), mode == ANALOG ? 1 : 0); // turn
on/off reporting
      }
      if (IS_PIN_DIGITAL(pin)) {
        if (mode == INPUT) {
          portConfigInputs[pin / 8] |= (1 << (pin & 7));
        } else {
          portConfigInputs[pin / 8] &= ~(1 << (pin & 7));
        }
      }
      pinState[pin] = 0;
      switch (mode) {
        case ANALOG:
          if (IS_PIN_ANALOG(pin)) {
            if (IS_PIN_DIGITAL(pin)) {
              pinMode(PIN_TO_DIGITAL(pin), INPUT);    // disable output driver
              digitalWrite(PIN_TO_DIGITAL(pin), LOW); // disable internal pull-ups
            }
            pinConfig[pin] = ANALOG;
          }
          break;
        case INPUT:
          if (IS_PIN_DIGITAL(pin)) {
            pinMode(PIN_TO_DIGITAL(pin), INPUT);    // disable output driver
            digitalWrite(PIN_TO_DIGITAL(pin), LOW); // disable internal pull-ups
            pinConfig[pin] = INPUT;
          }
          break;
        case OUTPUT:
          if (IS_PIN_DIGITAL(pin)) {
            digitalWrite(PIN_TO_DIGITAL(pin), LOW); // disable PWM
            pinMode(PIN_TO_DIGITAL(pin), OUTPUT);
            pinConfig[pin] = OUTPUT;
          }
          break;
        case PWM:
          if (IS_PIN_PWM(pin)) {
            pinMode(PIN_TO_PWM(pin), OUTPUT);
            analogWrite(PIN_TO_PWM(pin), 0);
            pinConfig[pin] = PWM;
          }
          break;
        case SERVO:
          if (IS_PIN_DIGITAL(pin)) {
            pinConfig[pin] = SERVO;
            if (servoPinMap[pin] == 255 || !servos[servoPinMap[pin]].attached()) {
              // pass -1 for min and max pulse values to use default values set
              // by Servo library
              attachServo(pin, -1, -1);
```

```
      }
    }
    break;
  case I2C:
    if (IS_PIN_I2C(pin)) {
      // mark the pin as i2c
      // the user must call I2C_CONFIG to enable I2C for a device
      pinConfig[pin] = I2C;
    }
    break;
  default:
    Firmata.sendString("Unknown pin mode"); // TODO: put error msgs in EEPROM
 }
 // TODO: save status to EEPROM here, if changed
}

void analogWriteCallback(byte pin, int value)
{
 if (pin < TOTAL_PINS) {
   switch (pinConfig[pin]) {
     case SERVO:
       if (IS_PIN_DIGITAL(pin))
         servos[servoPinMap[pin]].write(value);
       pinState[pin] = value;
       break;
     case PWM:
       if (IS_PIN_PWM(pin))
         analogWrite(PIN_TO_PWM(pin), value);
       pinState[pin] = value;
       break;
   }
 }
}

void digitalWriteCallback(byte port, int value)
{
 byte pin, lastPin, mask = 1, pinWriteMask = 0;

 if (port < TOTAL_PORTS) {
   // create a mask of the pins on this port that are writable.
   lastPin = port * 8 + 8;
   if (lastPin > TOTAL_PINS) lastPin = TOTAL_PINS;
   for (pin = port * 8; pin < lastPin; pin++) {
     // do not disturb non-digital pins (eg, Rx & Tx)
     if (IS_PIN_DIGITAL(pin)) {
       // only write to OUTPUT and INPUT (enables pullup)
       // do not touch pins in PWM, ANALOG, SERVO or other modes
       if (pinConfig[pin] == OUTPUT || pinConfig[pin] == INPUT) {
         pinWriteMask |= mask;
         pinState[pin] = ((byte)value & mask) ? 1 : 0;
       }
     }
     mask = mask << 1;
```

```
    }
    writePort(port, (byte)value, pinWriteMask);
  }
}


// -----------------------------------------------------------------------------
/* sets bits in a bit array (int) to toggle the reporting of the analogIns
 */
//void FirmataClass::setAnalogPinReporting(byte pin, byte state) {
//}
void reportAnalogCallback(byte analogPin, int value)
{
  if (analogPin < TOTAL_ANALOG_PINS) {
    if (value == 0) {
      analogInputsToReport = analogInputsToReport & ~ (1 << analogPin);
    } else {
      analogInputsToReport = analogInputsToReport | (1 << analogPin);
      // prevent during system reset or all analog pin values will be reported
      // which may report noise for unconnected analog pins
      if (!isResetting) {
        // Send pin value immediately. This is helpful when connected via
        // ethernet, wi-fi or bluetooth so pin states can be known upon
        // reconnecting.
        Firmata.sendAnalog(analogPin, analogRead(analogPin));
      }
    }
  }
  // TODO: save status to EEPROM here, if changed
}

void reportDigitalCallback(byte port, int value)
{
  if (port < TOTAL_PORTS) {
    reportPINs[port] = (byte)value;
    // Send port value immediately. This is helpful when connected via
    // ethernet, wi-fi or bluetooth so pin states can be known upon
    // reconnecting.
    if (value) outputPort(port, readPort(port, portConfigInputs[port]), true);
  }
  // do not disable analog reporting on these 8 pins, to allow some
  // pins used for digital, others analog.  Instead, allow both types
  // of reporting to be enabled, but check if the pin is configured
  // as analog when sampling the analog inputs.  Likewise, while
  // scanning digital pins, portConfigInputs will mask off values from any
  // pins configured as analog
}

/*==============================================================================
 * SYSEX-BASED commands
 *============================================================================*/
```

```
==*/

          void sysexCallback(byte command, byte argc, byte *argv)
          {
            byte mode;
            byte slaveAddress;
            byte data;
            int slaveRegister;
            unsigned int delayTime;

            switch (command) {
              case I2C_REQUEST:
                mode = argv[1] & I2C_READ_WRITE_MODE_MASK;
                if (argv[1] & I2C_10BIT_ADDRESS_MODE_MASK) {
                  Firmata.sendString("10-bit addressing not supported");
                  return;
                }
                else {
                  slaveAddress = argv[0];
                }

                switch (mode) {
                  case I2C_WRITE:
                    Wire.beginTransmission(slaveAddress);
                    for (byte i = 2; i < argc; i += 2) {
                      data = argv[i] + (argv[i + 1] << 7);
                      wireWrite(data);
                    }
                    Wire.endTransmission();
                    delayMicroseconds(70);
                    break;
                  case I2C_READ:
                    if (argc == 6) {
                      // a slave register is specified
                      slaveRegister = argv[2] + (argv[3] << 7);
                      data = argv[4] + (argv[5] << 7);  // bytes to read
                    }
                    else {
                      // a slave register is NOT specified
                      slaveRegister = REGISTER_NOT_SPECIFIED;
                      data = argv[2] + (argv[3] << 7);  // bytes to read
                    }
                    readAndReportData(slaveAddress, (int)slaveRegister, data);
                    break;
                  case I2C_READ_CONTINUOUSLY:
                    if ((queryIndex + 1) >= MAX_QUERIES) {
                      // too many queries, just ignore
                      Firmata.sendString("too many queries");
                      break;
                    }
                    if (argc == 6) {
                      // a slave register is specified
                      slaveRegister = argv[2] + (argv[3] << 7);
```

```
      data = argv[4] + (argv[5] << 7);  // bytes to read
    }
    else {
      // a slave register is NOT specified
      slaveRegister = (int)REGISTER_NOT_SPECIFIED;
      data = argv[2] + (argv[3] << 7);  // bytes to read
    }
    queryIndex++;
    query[queryIndex].addr = slaveAddress;
    query[queryIndex].reg = slaveRegister;
    query[queryIndex].bytes = data;
    break;
  case I2C_STOP_READING:
    byte queryIndexToSkip;
    // if read continuous mode is enabled for only 1 i2c device, disable
    // read continuous reporting for that device
    if (queryIndex <= 0) {
      queryIndex = -1;
    } else {
      // if read continuous mode is enabled for multiple devices,
      // determine which device to stop reading and remove it's data from
      // the array, shifiting other array data to fill the space
      for (byte i = 0; i < queryIndex + 1; i++) {
        if (query[i].addr == slaveAddress) {
          queryIndexToSkip = i;
          break;
        }
      }

      for (byte i = queryIndexToSkip; i < queryIndex + 1; i++) {
        if (i < MAX_QUERIES) {
          query[i].addr = query[i + 1].addr;
          query[i].reg = query[i + 1].reg;
          query[i].bytes = query[i + 1].bytes;
        }
      }
      queryIndex--;
    }
    break;
  default:
    break;
  }
  break;
case I2C_CONFIG:
  delayTime = (argv[0] + (argv[1] << 7));

  if (delayTime > 0) {
    i2cReadDelayTime = delayTime;
  }

  if (!isI2CEnabled) {
    enableI2CPins();
  }
```

```
  break;
case SERVO_CONFIG:
  if (argc > 4) {
    // these vars are here for clarity, they'll optimized away by the compiler
    byte pin = argv[0];
    int minPulse = argv[1] + (argv[2] << 7);
    int maxPulse = argv[3] + (argv[4] << 7);

    if (IS_PIN_DIGITAL(pin)) {
      if (servoPinMap[pin] < MAX_SERVOS && servos[servoPinMap[pin]].attached()) {
        detachServo(pin);
      }
      attachServo(pin, minPulse, maxPulse);
      setPinModeCallback(pin, SERVO);
    }
  }
  break;
case SAMPLING_INTERVAL:
  if (argc > 1) {
    samplingInterval = argv[0] + (argv[1] << 7);
    if (samplingInterval < MINIMUM_SAMPLING_INTERVAL) {
      samplingInterval = MINIMUM_SAMPLING_INTERVAL;
    }
  } else {
    //Firmata.sendString("Not enough data");
  }
  break;
case EXTENDED_ANALOG:
  if (argc > 1) {
    int val = argv[1];
    if (argc > 2) val |= (argv[2] << 7);
    if (argc > 3) val |= (argv[3] << 14);
    analogWriteCallback(argv[0], val);
  }
  break;
case CAPABILITY_QUERY:
  Firmata.write(START_SYSEX);
  Firmata.write(CAPABILITY_RESPONSE);
  for (byte pin = 0; pin < TOTAL_PINS; pin++) {
    if (IS_PIN_DIGITAL(pin)) {
      Firmata.write((byte)INPUT);
      Firmata.write(1);
      Firmata.write((byte)OUTPUT);
      Firmata.write(1);
    }
    if (IS_PIN_ANALOG(pin)) {
      Firmata.write(ANALOG);
      Firmata.write(10); // 10 = 10-bit resolution
    }
    if (IS_PIN_PWM(pin)) {
      Firmata.write(PWM);
      Firmata.write(8); // 8 = 8-bit resolution
```

```
      }
      if (IS_PIN_DIGITAL(pin)) {
        Firmata.write(SERVO);
        Firmata.write(14);
      }
      if (IS_PIN_I2C(pin)) {
        Firmata.write(I2C);
        Firmata.write(1);  // TODO: could assign a number to map to SCL or SDA
      }
      Firmata.write(127);
    }
    Firmata.write(END_SYSEX);
    break;
  case PIN_STATE_QUERY:
    if (argc > 0) {
      byte pin = argv[0];
      Firmata.write(START_SYSEX);
      Firmata.write(PIN_STATE_RESPONSE);
      Firmata.write(pin);
      if (pin < TOTAL_PINS) {
        Firmata.write((byte)pinConfig[pin]);
        Firmata.write((byte)pinState[pin] & 0x7F);
        if (pinState[pin] & 0xFF80) Firmata.write((byte)(pinState[pin] >> 7) & 0x7F);
        if (pinState[pin] & 0xC000) Firmata.write((byte)(pinState[pin] >> 14) & 0x7F);
      }
      Firmata.write(END_SYSEX);
    }
    break;
  case ANALOG_MAPPING_QUERY:
    Firmata.write(START_SYSEX);
    Firmata.write(ANALOG_MAPPING_RESPONSE);
    for (byte pin = 0; pin < TOTAL_PINS; pin++) {
      Firmata.write(IS_PIN_ANALOG(pin) ? PIN_TO_ANALOG(pin) : 127);
    }
    Firmata.write(END_SYSEX);
    break;
  }
}

void enableI2CPins()
{
  byte i;
  // is there a faster way to do this? would probaby require importing
  // Arduino.h to get SCL and SDA pins
  for (i = 0; i < TOTAL_PINS; i++) {
    if (IS_PIN_I2C(i)) {
      // mark pins as i2c so they are ignore in non i2c data requests
      setPinModeCallback(i, I2C);
    }
  }

  isI2CEnabled = true;
```

```
  Wire.begin();
}

/* disable the i2c pins so they can be used for other functions */
void disableI2CPins() {
  isI2CEnabled = false;
  // disable read continuous mode for all devices
  queryIndex = -1;
}

/*==============================================================
===========
 * SETUP()

*==============================================================
==*/

void systemResetCallback()
{
  isResetting = true;

  // initialize a defalt state
  // TODO: option to load config from EEPROM instead of default

  if (isI2CEnabled) {
    disableI2CPins();
  }

  for (byte i = 0; i < TOTAL_PORTS; i++) {
    reportPINs[i] = false;    // by default, reporting off
    portConfigInputs[i] = 0;  // until activated
    previousPINs[i] = 0;
  }

  for (byte i = 0; i < TOTAL_PINS; i++) {
    // pins with analog capability default to analog input
    // otherwise, pins default to digital output
    if (IS_PIN_ANALOG(i)) {
      // turns off pullup, configures everything
      setPinModeCallback(i, ANALOG);
    } else {
      // sets the output to 0, configures portConfigInputs
      setPinModeCallback(i, OUTPUT);
    }

    servoPinMap[i] = 255;
  }
  // by default, do not report any analog inputs
  analogInputsToReport = 0;

  detachedServoCount = 0;
  servoCount = 0;
```

```
  /* send digital inputs to set the initial state on the host computer,
   * since once in the loop(), this firmware will only send on change */
  /*
  TODO: this can never execute, since no pins default to digital input
        but it will be needed when/if we support EEPROM stored config
  for (byte i=0; i < TOTAL_PORTS; i++) {
    outputPort(i, readPort(i, portConfigInputs[i]), true);
  }
  */
  isResetting = false;
}

void setup()
{
  Firmata.setFirmwareVersion(FIRMATA_MAJOR_VERSION, FIRMATA_MINOR_VERSION);

  Firmata.attach(ANALOG_MESSAGE, analogWriteCallback);
  Firmata.attach(DIGITAL_MESSAGE, digitalWriteCallback);
  Firmata.attach(REPORT_ANALOG, reportAnalogCallback);
  Firmata.attach(REPORT_DIGITAL, reportDigitalCallback);
  Firmata.attach(SET_PIN_MODE, setPinModeCallback);
  Firmata.attach(START_SYSEX, sysexCallback);
  Firmata.attach(SYSTEM_RESET, systemResetCallback);

  Firmata.begin(57600);
  systemResetCallback();  // reset to default config
}

/*==============================================================
===========
 * LOOP()

*===============================================================
==*/
void loop()
{
  byte pin, analogPin;

  /* DIGITALREAD - as fast as possible, check for changes and output them to the
   * FTDI buffer using Serial.print()  */
  checkDigitalInputs();

  /* STREAMREAD - processing incoming messagse as soon as possible, while still
   * checking digital inputs.  */
  while (Firmata.available())
    Firmata.processInput();

  // TODO - ensure that Stream buffer doesn't go over 60 bytes

  currentMillis = millis();
  if (currentMillis - previousMillis > samplingInterval) {
    previousMillis += samplingInterval;
    /* ANALOGREAD - do all analogReads() at the configured sampling interval */
```

```
      for (pin = 0; pin < TOTAL_PINS; pin++) {
        if (IS_PIN_ANALOG(pin) && pinConfig[pin] == ANALOG) {
          analogPin = PIN_TO_ANALOG(pin);
          if (analogInputsToReport & (1 << analogPin)) {
            Firmata.sendAnalog(analogPin, analogRead(analogPin));
          }
        }
      }
      // report i2c data for all device with read continuous mode enabled
      if (queryIndex > -1) {
        for (byte i = 0; i < queryIndex + 1; i++) {
          readAndReportData(query[i].addr, query[i].reg, query[i].bytes);
        }
      }
    }
}
```

BIBLIOGRAPHY

Althusser, Louis, *Lenin and Philosophy and other Essays.* New York: Monthly Review Press, 2001.

Ballard, J.G, *Vermilion Sands*. London: Vintage, 1971.

Banham, Reyner, "A Home is Not a House," Art in Architecture 2 (1965): 70-79.

Beach Phillip, *Muscles and Merdians: The Manipulation of Shape*. Edinburgh: Churchil Livingstone, 2010.

Beach, Phillip, *Muscles and Meridians: The Manipulation of Shape.* Edinburgh: Churchill Livingstone, 2010.

Beach, Phillp, *Archetypical Postures*. Accessed February 2017. URL: http://www.phillipbeach.com/archetypal_postures.html Accessed February 2017.

Benavides, X., Amores, J., Maes, P., "Psychicvr: Increasing Mindfulness by using Virtual Reality and Brain Computer Interfaces". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/425

Branzi, Andrea, *No-Stop City: Archizoom Associati*, Orléans: HYX, 2006.

Brukman, Amy, "*Identity Workshop: Emergent Social and Psychological Phenomena in Text-Based Virtual Reality*." Unpublished Paper, Accessed February 2017. URL: http://www.cc.gatech.edu/fac/Amy.Bruckman/papers.html.

Caballero, Rosario, *Re-viewing Space: Figurative Language in Architects' Assessment of Built Space*, Berlin; New York, NY: Mouton de Gruyter, 2006.

Chappel, Brian D, "*Ephemeral Architecture: Towards Definition*", Unpublished Document. Wordpress. Accessed May 2017. URL: https://evrosoriou.files.wordpress.com/2012/05/ephemeral-architecture1.pdf

Cheng, Anlin A., Baker, Joseph, *Second Skin: Josephine Baker & the Modern Surface*. New York: Oxford University Press, 2011.

Coelho, M., Maes, P., "Sprout I/O:A texturally rich interface.", Proceedings of the 2nd international conference on Tangible and embedded interaction. ACM, 2008, Pages 221-222.

Coelho, M., Maes, P., "Shutters: a permeable surface for environmental control and communication.", Proceedings of the 3rd international conference on Tangible and embedded interaction. ACM, 2009, Pages 13-18.

Dazkir, Sibel S., Read, Marilyn A., Furniture forms and their influence on our Emotional Responses Toward Interior Environments, Environment and Behavior 44, No. 5 (2012): 722-732 de Souza, Renato Cesar Ferreira, "*Ephemeral Spaces*", M.S.C Thesis, Federal University of Minas, 1998.

Fox, Michael, Kemp, Miles, *Interactive Architecture*. New York: Princeton Architectural Press, 2009.

Gibson, William, *Neuromancer*. New York: Ace Books, 2004.

Gordon, Alastair, *Spaced Out*. New York: Rizzoli, 2008.

Hewes, Goedon W., "World Distribution of Certain Postural Habits." American Anthropologist 57, No. 2 (1955): 231-44. doi:10.1525/aa.1955.57.2.02a00040.

Liu, X., Qian, Y., Maes, P., "*Treesense: Eliciting Body Ownership Illusion in Virtual Reality with Electrical Muscle Stimulation*". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/465

Negroponte, Nicholas, *Soft Architecture Machines*, Cambridge, MIT Press, 1975.

Oh, Seoug, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014. URL: https://etda.libraries.psu.edu/catalog/23660

Rasmussen, M.K., Pedersen, E.W., Petersen, M.G., Hornbaek, K., "Shape-changing interfaces: a review of the design space and open research questions", Proceedings of the SIGCHI Conference on Human Factors in Computer Systems, CHI 2012, Pages 735-744.

Rybczynski, Witold, "*Now I Sit Me Down: From Klismos to Plastic Chair: A Natural History*, New York, Farrar, Straus and Giroux, 2016.

Streeck, J., Goodwin, C., LeBaron, C., *Embodied Interaction: language and body in the material world*, Cambridge, U Press, 2011.

Turckle, Sherry, *Life on a Screen*, New York, Touchstone, 1995.

Ugur, Secil, Wearing Embodied Emotions: A Practice Based Design Research on Wearable Technology, Milan, Springer, 2013.

Wigley, Mark, *Contant's New Babylon: the hyper-architecture of desire*, Rotterdam, 010 Publishers, 1998.

Yang, D., Verma, M.S., So, J.H., Mosadegh, B., Keplinger, C., Lee, B., Khashai, F., Lossner, E., Suo, Z. and Whitesides, G.M., "*Buckling Pneumatic Linear Actuators Inspired by Muscle*", Advanced Materials Technologies 1, No.3, 2016.

ENDNOTES

[1] Virtopia: A portmanteau of Virtual and Utopia that defines a virtual utopia for users of the digital world.

[2] Burrowed from the categories defined by Sylvie Kruger, *Textile Architecture*, (Berlin, Jovis, 2009). In Textile Architecture, horizontal space definers have been limited to the roof level, here I use it to define a space definer in the ground level where it is used for settlement.

[3] Rosario Caballero, *Re-viewing Space: Figurative Language in Architects' Assessment of Built Space*, (Berlin; New York, NY: Mouton de Gruyter, 2006.), 21.

[4] Secil Ugur, Wearing Embodied Emotions: A Practice Based Design Research on Wearable Technology, (Milan, Springer, 2013).

[5] Lysergic Acid Diethylamide, also known as acid, is a psychedelic drug known for its psychological effects.

[6] "How was one expected to return to the routines of the everyday life after a journey to the deepest recesses of inner space?" Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008), 20.

[7] Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008), 61.

[8] Andrea Branzi, *No-Stop City: Archizoom Associati*, (Orléans: HYX, 2006).

[9] Mark Wigley, *Contant's New Babylon: the hyper-architecture of desire*, (Rotterdam, 010 Publishers, 1998).

[10] Verner Panton, Visionia, Official Reference Portal, Accessed May 2017. URL: http://www.verner-panton.com/spaces/archive/121/

[11] Environment Bubble, Reyner Banham, "A Home is Not a House," Art in Architecture 2 (1965): 70-79.

[12] "Cyber Space", environments in which communication happens over a network of computer systems. William Gibson, Neuromancer, (New York, Ace Books, 2004).

[13] Amy Bruckman, "*Identity Workshop: Emergent Social and Psychological Phenomena in Text-Based Virtual Reality*." Unpublished Paper, Accessed February 2017. URL: http://www.cc.gatech.edu/fac/Amy.Bruckman/papers.html.

[14] Sherry Turckle, Life on a Screen, (New York, Touchstone, 1995).

[15] Xin Liu, Yedan Qian, Pattie Maes, "*Treesense: Eliciting Body Ownership Illusion in Virtual Reality with Electrical Muscle Stimulation*". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/465

[16] Xin Liu, Yedan Qian, Pattie Maes, "*Treesense: Eliciting Body Ownership Illusion in Virtual Reality with Electrical Muscle Stimulation*". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/465

[17] Xavier Benavides, Judith Amores, Pattie Maes, "Psychicvr: Increasing Mindfulness by using Virtual Reality and Brain Computer Interfaces". MIT Media Lab Projects, Accessed May 2017. URL: http://fluid.media.mit.edu/node/425

[18] Microsoft Hololens, Accessed May 2017. URL: https://www.microsoft.com/en-us/hololens

[19] Theories in Embodied Interaction implies that human psychology could be studied through human movement in space.
Jurgen Streeck, Charles Goodwin, Curris LeBaron, *Embodied Interaction: language and body in the material world*, (Cambridge, U Press, 2011).

[20] Nicholas Negroponte, *Soft Architecture Machines*, (Cambridge, MIT Press, 1975), 131–132.

[21] A collection of work has been represented at : Michael Fox, *Interactive Architecture*, (New York: Princeton Architectural Press, 2009).

[22] Michael Fox, *Interactive Architecture*, (New York: Princeton Architectural Press, 2009).

[23] Seoug Oh, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014. URL: https://etda.libraries.psu.edu/catalog/23660

[24] Michael Fox, *Interactive Architecture*, (New York: Princeton Architectural Press, 2009), 30.

[25] Seoug Oh, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014, 22. URL: https://etda.libraries.psu.edu/catalog/23660

[26] Seoug Oh, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014, 22. URL: https://etda.libraries.psu.edu/catalog/23660

[27] Michael Fox, *Interactive Architecture*, (New York: Princeton Architectural Press, 2009).

[28] Seoug Oh, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014, 23. URL: https://etda.libraries.psu.edu/catalog/23660

[29] Future Cities Lab, Data Grove, "Seeking Silicon Valley" 2012 Biennial, Accessed May 2017. URL: http://www.future-cities-lab.net/datagrove
[30] Haque design + research, Marling, November 2012, Accessed May 2017. URL: http://www.haque.co.uk/marling.php

[31] Haque design + research, Marling, November 2012, Accessed May 2017. URL: http://www.haque.co.uk/marling.php

[32] Seoug Oh, "*A step Towards a Topology of Responsive Architecture: Actors, Interaction, Transformations*", Masters Thesis, The Pennsylvania State University, 2014, 25. URL: https://etda.libraries.psu.edu/catalog/23660

[33] Haque design + research, DIY City 0.01a, Accessed May 2017, URL: http://www.haque.co.uk/diycity.php

[34] Scott Snibbe, Boundary Function, 1998, Accessed May 2017. URL: https://www.snibbe.com/projects/interactive/boundaryfunctions/

[35] Scott Snibbe, Boundary Function, 1998, Accessed May 2017. URL: https://www.snibbe.com/projects/interactive/boundaryfunctions/

[36] Renato Cesar Ferreira de Souza, "*Ephemeral Spaces*", M.S.C Thesis, Federal University of Minas, 1998.

[37] Brian D Chappel, "*Ephemeral Architecture: Towards Definition*", Unpublished Document. Wordpress. Accessed May 2017. URL: https://evrosoriou.files.wordpress.com/2012/05/ephemeral-architecture1.pdf

[38] Majken K. Rasmussen, Esben W. Pedersen, MarianneG. Petersen, Kasper Hornbaek, "Shape-changing interfaces: a review of the design space and open research questions", Proceedings of the SIGCHI Conference on Human Factors in Computer Systems, CHI 2012, Pages 735-744.

[39] Future Cities Lab, Data Grove, "Seeking Silicon Valley" 2012 Biennial, Accessed May 2017. URL: http://www.future-cities-lab.net/datagrove

[40] Marcelo Coelho, Pattie Maes, "Sprout I/O:A texturally rich interface.", Proceedings of the 2nd international conference on Tangible and embedded interaction. ACM, 2008, Pages 221-222.

[41] Marcelo Coelho, Pattie Maes, "Shutters: a permeable surface for environmental control and communication.", Proceedings of the 3rd international conference on Tangible and embedded interaction. ACM, 2009, Pages 13-18.

[42] Joachim Sauter, Kinetic Sculpture BMW, 2008. Accessed May 2017. URL: Joachim Sauter. http://www.joachimsauter.com/en/work/bmwkinetic.html

[43] Adafruit Website, Accessed February 2017. URL: https://www.adafruit.com/

[44] Dian Yang, Mohit .S. Verma, Ju-Hee So, Bobak Mosadegh, Christoph Keplinger, Benjamin Lee, Fatemeh Khashai, Elton Lossner, Zhigang Suo, and George M. Whitesides "*Buckling Pneumatic Linear Actuators Inspired by Muscle*", Advanced Materials Technologies 1, No.3, 2016.

[45] XYZAidan, Flat Pnumatic Actuators, Youtube Tutorial. Accessed May2017, URL: https://www.youtube.com/watch?v=LlxUZABcah0

[46] Microcontroller Definition, Wikipedia. Accessed May 2017. URL: https://en.wikipedia.org/wiki/Microcontroller

[47] Controlling Solenoids with Arduino, Instructables. Accessed May 2017. URL: http://www.instructables.com/id/Controlling-solenoids-with-arduino/

[48] BC Robotics, Accessed May 2017. URL: https://www.bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/

[49] Controlling Solenoids with Arduino, Instructables. Accessed May 2017. URL: http://www.instructables.com/id/Controlling-solenoids-with-arduino/

[50] Neoprene Matrix Pressure Sensor by Kobokant, How to Get What you Want, Accessed May

2017. URL: http://www.kobakant.at/DIY/?p=213

[51] Pressure Sensor Matrix, Instructables, Accessed May 2017. URL: http://www.instructables.com/id/Pressure-Sensor-Matrix/

[52] Material processes for the pressure sensor all have been used from the Pressure Sensor Matrix Tutorial, Accessed May 2017. URL: http://www.instructables.com/id/Pressure-Sensor-Matrix/

[53] GeomGym for Grasshopper and Rhino, Created by Jon Mitschn. URL: http://www.grasshopper3d.com/group/geometrygym

[54] Nasim Motalebi, J. P. Duarte, "A Shape Grammar of Emotional Postures: An approach towards encoding the analogue qualities of bodily expressions of emotions", eCAADe 2017, Rome.

[55] Super Studio," Supersurface - An alternative model for life on the Earth", 1972, Italy. URL: http://www.architectureplayer.com/clips/supersurface-an-alternative-model-for-life-on-the-earth. Accessed July 2017.

[56] Situationist International Wikipedia, URL: https://en.wikipedia.org/wiki/Situationist_International, Accessed July 2017.

[57] "The Situationist International (SI) was an international organization of social revolutionaries made up of avant-garde artists, intellectuals, and political theorists, prominent in Europe from its formation in 1957 to its dissolution in 1972". Sadie Plant, *The most Radical Gesture*, (NewYork, Routledge, 1992).

[58] Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008).

[59] Alastair Gordon, *Spaced Out*, (New York, Rizzoli, 2008).

[60] Sibel S. Dazkir, M.A. Read, "Furniture forms and their influence on our emotional responses toward interior environments." Environment and Behavior 44, no5, 2012, 722-732.

[61] Secil Ugur, *Wearing Embedded Emotions: A practice based design research on wearable technology*, (Milan, Springer, 2013).

[62] Witold Rybczynski, *Now I Sit Me Down: From Klismos to Plastic Chair: A Natural History*, (New York, Farrar, Straus and Giroux, 2016).

[63] Phillip Beach, *Archetypical Postures*, Accessed February 2017. URL: http://www.phillipbeach.com/archetypal_postures.html

[64] Gordon W. Hewes, World Distribution of Certain Postural Habits, American Anthropologist 57, No.2, (1955): 231-44.

[65] Witold Rybczynski, *Now I Sit Me Down: From Klismos to Plastic Chair: A Natural History*, (New York, Farrar, Straus and Giroux, 2016).

[66] On an interview with Rybczynski. Uri Friedman, "A Global History of Sitting Down: The world of chairs, from ancient Greece to the modern office", The Atlantic, August 2016. Accessed May 2017. URL: https://www.theatlantic.com/international/archive/2016/08/chairs-history-witold-rybczynski/497657/

[67] Witold Rybczynski, *Now I Sit Me Down: From Klismos to Plastic Chair: A Natural History*, (New York, Farrar, Straus and Giroux, 2016), Pages 1-40.