



BIOLOGICAL ALGORITHMS FOR DIGITAL MANUFACTURE

Louise Wotton

MArch (Prof) Programme
Victoria University of Wellington, School of Architecture, New Zealand

2018 // 2019

■ Biological Algorithms for Digital Manufacture ■

By

Louise Wotton

A 120 point thesis submitted to the Victoria University of Wellington in
fulfillment of the requirements for the degree of Master of Architecture
(Professional)

Victoria University of Wellington
School of Architecture
2019

ABSTRACT

Computational simulations are generally built upon a form or design that is near or mostly complete. Agent-based simulations are ones where the rules and behaviours are designed, creating an unpredictable output. In this research, these rules are derived from the complex systems in nature, utilising cross-disciplinary principles between architecture and biology. The abstraction of data and rules from biological structures are used to inform computational rule-sets for modelling 3D printed structures.

The simulations in this paper explore the concept of emergence: where systems have an irreducible complexity and adaptability - a series of smaller parts combined acting as a whole. The concept of agent-based simulations as a form of emergence is a tool used greatly within many areas of research as a speculative method to build form and space.

Computation rule-sets define a design intent for each simulation, demonstrating the ability to use agent-based systems and a spatial design driver. Informing the agents with design intent, allows them to adapt to their environment and to the ability and limitations of a free-form 3D printer.

The focus in this project is the design of emergent principles in nature and how they can be applied to optimize structures for use with digital fabrication methods, thus producing a new approach to designing fabricated forms.

Using a design by research approach, this research demonstrates the potential of free-form 3D printing as a technique for an integrated fabrication system. It outlines computational design techniques including the simulation of emergent phenomena to define a digital workflow that supports the integration of both emergent structures and free-form printing.

CONTENTS

Terminology	10
<i>INTRODUCTION</i>	14
Aims and Objectives	17
Research Methodology	18
Scope	19
<i>CHAPTER 1: DIGITAL RESEARCH</i>	
Parametric Design	23
Introduction	27
Precedents	28
Complexity Theory	32
Agent-Based Systems	34
Stigmergy	38
Architectural Application	40
Generative Design	43
Grasshopper Experiments	44
Physical Modelling	50
Vector Informed Simulations	54
Processing	58
Processing Code	62
Data Communication	73
Processing Development	74
Digital Authorship	77
<i>CHAPTER 2: DIGITAL APPLICATION</i>	
Introduction	81
Sun-shading	82
Sun Shade Designer	87

Dynamic Shading	89
Agents with Design Intent	90
Digital Application Work-flow	92

CHAPTER 3 : FABRICATION RESEARCH

Introduction	97
Precedents	98
Rapid Code + Planes	101
Robot + Workspace	102
VUW Extruder	105
Materiality	106
Script Development	108
Printer Set Up	110
Series 1, 2, 3	112
Print Errors	134
Agent Based Structures Printed	136
Fabrication Feedback Loop	138
Extruder Development	140
Conclusion	145

CHAPTER 4 : DIGITAL PROTOTYPE + FABRICATION

Research Reflection	149
Voxels	151
Voxel Combination Design	154
Print Iterations	162
Voxel-based Swarm Tectonics	172
Application of Swarm to Voxel Design	179
Vector informed 3D Print	180

CHAPTER 5: DIGITAL APPLICATION 2.0

Introduction	187
Sun-shading and Simulation techniques	188
Swarm Tests	192
Shading Results	194
Sun Shade Design	197
Density Manipulation	200
Conclusion	202

CHAPTER 6: PROTOTYPE FABRICATION

Introduction	207
Second Extruder	208
Model Development	210
Fabrication Development	214
Extruder Testing	216
Print Errors and Resolution	222
Printer Reflection	225
Conclusion	226

CHAPTER 7: DISCUSSION

Limitations	231
Opportunities	232
Conclusion	234

+

Works Cited	244
Appendix	250

“A meaningful building of the digital age is not one that is built using digital tools; it is one that could not have been built without them.”

-Mario Carpo. *'The Digital Turn in Architecture'*, 2012

TERMINOLOGY

ABB ASEA Brown Boveri, Robot Manufacturer

Additive Manufacturing ‘the process of joining materials to make objects from 3D model data, usually layer upon layer’. (ASTM, 2010)

Algorithm “a computational procedure for addressing a problem in a finite number of steps, it is the systematic extraction of logical principles and the development of a general solution plan”. (Terzidis, 2006)

Arduino is an open source electronics platform intended for interactive projects. It has its own coding language and hardware that is able to be controlled.

G-Code a text based programming language, mainly used to control automated machine tools.

Grasshopper Add-on visual programming parametric software for Rhino

Grasshopper Cluster a series of Grasshopper components clustered into one component for ease of script use.

HAL Robotics a plug-in for Grasshopper used to design Rapid code for the robot use, uses a series of components to customise the movement and actions of the robot.

Operations (HAL Robotics) Operations are commands fed to the robot in HAL that control the number of operations available, including pick up the extruder, turn fan on and other printing related commands.

Processing an open source software sketchbook and language for coding within the context of visual arts. Processing uses the Java language with additional simplifications.

Python an interpreted high-level programming language for general purpose programming.

Rapid Code a high-level programming language used to control ABB industrial robots. The language includes routine parameters, procedures, logical expressions, automatic error handling and functions of the robot.

Rhinoceros 3D Computer aided design software

XYZ Coordinates or also referred to as a Cartesian coordinate system. Where a point is referred to with a set of values in the X,Y and Z directions

INTRODUCTION

/ Biological Algorithms for Digital Manufacture

INTRODUCTION

Mario Carpo's 'The Alphabet and the Algorithm' details a digital architecture in the 1990s that was criticised for not contributing enough to fill the gap between construction and computer-aided design. Over the past twenty years a digital revolution as he calls it has brought technology into architecture in a similar way that the industrial revolution provided for mass-produced, identical copies: a repetitive, generic model of architecture has become culturally expected. (Carpo, 2011)

Digital architecture has taken another shift, with the introduction of generative design, big data and digital fabrication. These tools have provided the possibility of returning to an individualistic design, as well as the feasibility to build complex parts and intricate surfaces, giving unprecedented freedom that the 'digital revolution' initially removed.

The scope of a digitally controlled fabrication process has greatly expanded with the use of advanced CNC tools such as industrial robots in architectural research. Robots push the boundaries of fabrication far beyond traditional 3D printers, CNC mills and laser cutters. The customisation possibility pushes much further than just the fabrication process, but the digital reception and data it understands, allowing us to take advantage of generative processes.

This thesis will cover techniques and methods that connect both digital generative design and fabrication methods. Through design and experimentation with generative models, it will discuss the effects of new design paradigms and how this can be connected to digital fabrication methods. Both design considerations and fabrication possibilities and limitations will be considered, taking advantage of the versatile nature of an advanced CNC fabrication tool in an industrial robot.

To drive the generative design research, inspiration and form generative rules will be taken from emergent principles. Complexity, when considered in the biological world, are behaviours derived from nature, where many small parts together undergo spontaneous self-organisation. These complex, self-organising systems are adaptive, actively trying to turn whatever happens to their advantage. (Mitchell Waldrop, 1992)

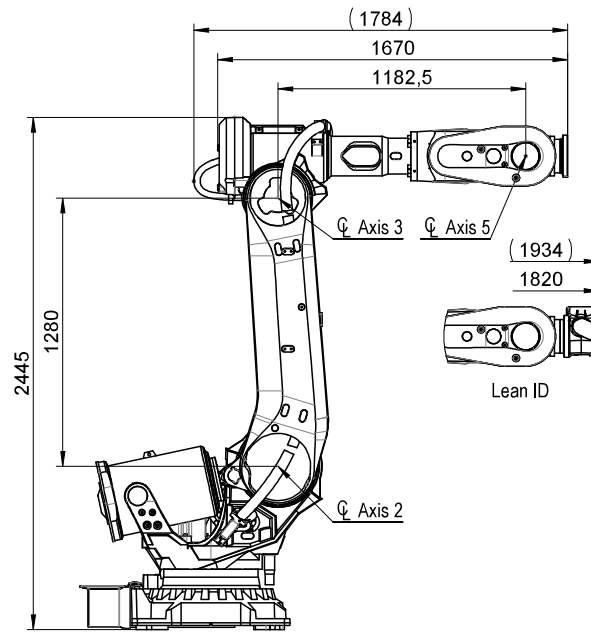
Taking advantage of the use of generative design and big data, emergent principles will be studied and simulated with intentions of translating them into a fabricated system. Utilising agent-based systems, the self-organisation aspects of emergent principles are able to be applied to a functional workflow.

In the field of architecture where robots have been introduced in the last decade, a remarkable number of small and sophisticated architectural structures and elements have already been tested and built. Examining some of these projects, this thesis will refine that work-flow, concentrating on the entire design and fabrication process as one fluid method. Using the emergent simulations as the form driver, 3D printing techniques will be integrated as design intent for simulations, while focussing on the individuality of both the digital and the fabricated elements.

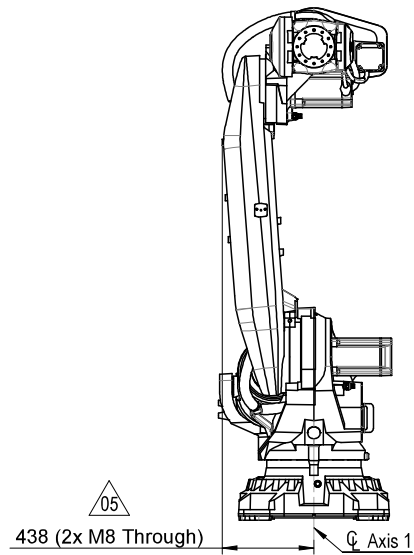
The thesis poses the research question of:

“how can digital tools be used to simulate patterns and phenomenon derived from nature and use these to inform architectural methods, creativity and physical production”

In answering the research question, the thesis aims to research and explore the use of agent-based systems in architecture and apply them in variable fabricated forms. Research into precedent projects and digital techniques will provide the foundation for initial experimentations , while considering the computational design and fabrication capabilities.



IRB 6700-205/2.80



IRB 6700-205/2.80

0.2

AIMS AND OBJECTIVES

The aim of this research is to explore the potential of a robotic freeform 3D printer combined with basic biomimicry and emergence principles to produce innovative structures.

1. Undertake research into previous projects using free-form printing and their use of the printer and materials.
2. Research into biomimicry principles which can be used to inform the structural nature of free-form prints.
3. Iteratively and parametrically model and simulate designs at a range of scales.
4. Structurally and spatially analyse results using a series of strengths and controls to assess the printing methods.
5. Test accuracy and output of the free-form printing extruder and its performance using extruder related parameters to establish optimal performance.
6. Establish potential applications in conjunction with the National Science Challenge

0.3

POSITION AND MOTIVATION

The motivation for this research thesis is to enrich my own work within the realm of digital design. The digital movement has begun informing architecture as a profession as a fast paced, growing section of both architectural research and practice.

Through the literature review, this growth became prevalent to me, and I hope to equip myself with the skills in both the digital and fabrication research areas. As well as the understanding and knowledge needed to create generative, emergent and responsive forms and environments, through programming explorations and research into the potential for integrating this into a creative design methodology.

0.4

RESEARCH METHODOLOGY

The research method used in this thesis focuses on a feedback loop, with the end goal to use digital models and 3D print them. In conjunction with research into past projects, fabrication methods and computational design tools, work is iteratively analysed.

Using design by research allows informed decisions of both key ideas and digital simulations, by reviewing the performance of digital models in terms of their suitability to an architectural application.

This is essential as while this project is informed by research, it is predominantly computer driven, so making informed decisions allows more control over future steps.

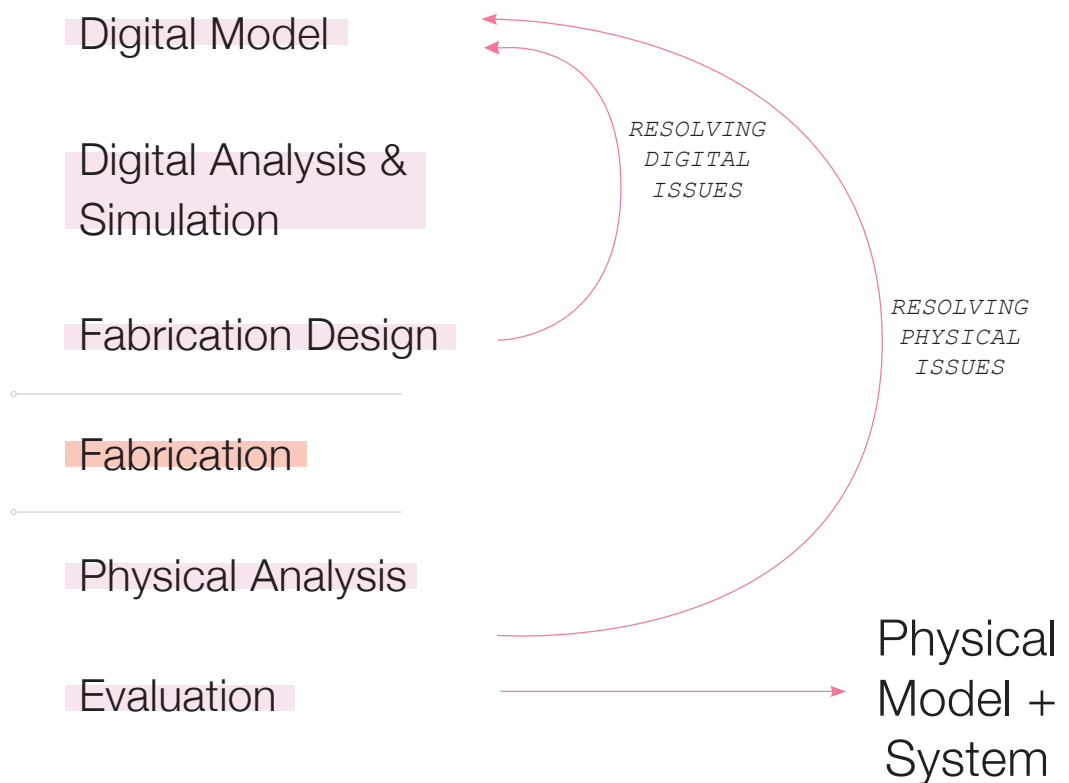


Figure 2. Feedback Loop method

SCOPE

This thesis focuses on three key parts: design intent, simulation and fabrication. Each part will be discussed individually over a series of chapters. The final two chapters will bring together each section to detail a final work-flow. Together these three sections are connected through software communication and data transfers.

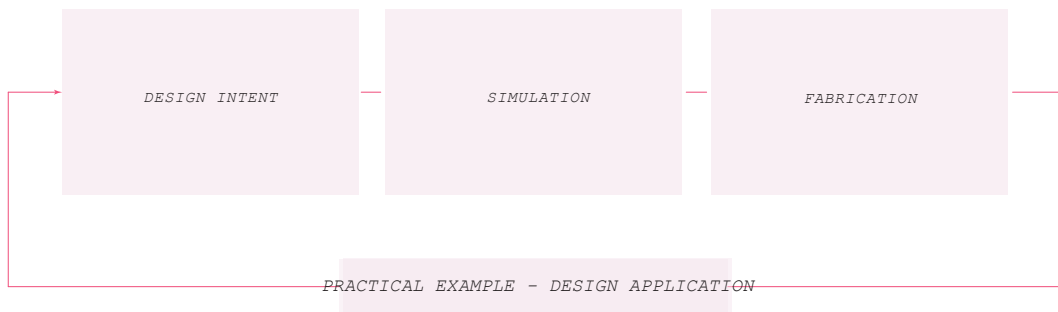


Figure 3. Scope

Through these three sections, the research will create a theoretical framework, and apply it to a practical example. The application to a practical example demonstrates the ability that the work-flow established in this thesis will have.

CHAPTER 1: DIGITAL RESEARCH

/ Biological Algorithms for Digital Manufacture

PARAMETRIC DESIGN

A key part of this thesis is the design and creation of variables within parametric software to inform a toolpath for freeform 3D printing. Parametric design is where values of parameters within a schema of relationships can be manipulated for the generation of a design.

“a logic of associative and dependency relationships between objects and their part-to-whole relationships” (Oxman, 2014)

A key advantage to parametric design is the flexibility for the user. The ability to adjust pre-modelled components, geometries or codes through pre-defined parameters. Designers are able to adjust aspects of design with intent towards a final product.

Used predominately in digital architecture since the 1990's, parametric architecture focuses on the use of digital tools to quickly develop a final product. Greg Lynn, an architect well known in the '*digital world*' of architecture describes parametric design as a "*computer modelling process whereby incremental decisions are defined in series and can be recalled and changed at any point in the chain of design decisions*". (Greg Lynn 2008)

The key part being the *incremental*, where minute changes are able to have an endless impact on the final design. Parametric design is used throughout this thesis, in both the digital simulations designed to build form, and the programming and coding of the robot. The timing and print attributes are all also able to be edited parametrically, resulting in a cohesive, connected digital design.

“nature is a
brilliant engineer”

- Neri Oxman, MIT

INTRODUCTION

This chapter of the research focuses on the simulation section of the thesis. Outlined in the introduction, agent based systems were described as the inspiration for the form of the design.



Exploration into a range of precedent projects relevant to the following areas in this research will be undertaken in an attempt to highlight the capabilities of both agent based simulations and robotic fabrication

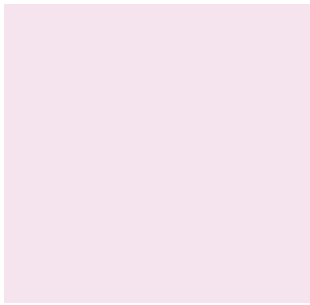
- Robotics
- 3D Printing
- Generative Design
- Biomimicry
- Agent-based systems

This chapter will then move to build a digital database of code capable of simulating agent-based systems for fabrication. At this stage the research is focussed on the development of this code, and future chapters will detail the design intent when a level of complexity is written into each simulation.

1.02

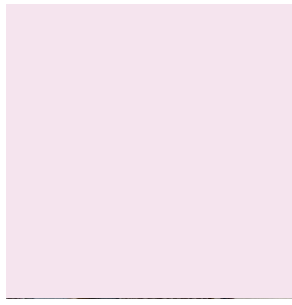
PRECEDENTS

Project Description			
	KOKKUGIA	CLOUD PAVILION	FIBRO CITY
Research Areas	<p>'Kokkugia' is a design research lab based in Melbourne, founded by Roland Snooks in 2004. Their current research is focused on iterative algorithms to produce and augmentation of intricate mass - leaving generative design to design rather than a predefined form. Snook's design approach utilises multi agent systems for the exploration of 'behavioural formation' (Snooks, 2016).</p>	<p>Alisa Andrasek's fabrication project 'Cloud Pergola', is a 3D printed installation designed using multi-agent system algorithms. The structure was free-form printed using thermoplastics to form the complex spatial lattice. Ai Build, a company specialising in robotic 3D printing fabricated the design. (Andrasek, 2018)</p>	<p>Fibrocitiy is project based on fibrous formations translated into architecture scale. Using carbon fibre for high performance through discreteness, the fabrication method developed in this project is interactive. The algorithms read the environment and the model, anchoring and weaving carbon fibre to build a high resolution, structural and aesthetic model. (Komar et al., 2014)</p>
	ROBOTICS	ROBOTICS	ROBOTICS
	3D PRINTING	3D PRINTING	3D PRINTING
	GENERATIVE DESIGN	GENERATIVE DESIGN	GENERATIVE DESIGN
	BIOMIMICRY	BIOMIMICRY	BIOMIMICRY
	AGENT-BASED SYSTEM	AGENT-BASED SYSTEM	AGENT-BASED SYSTEM



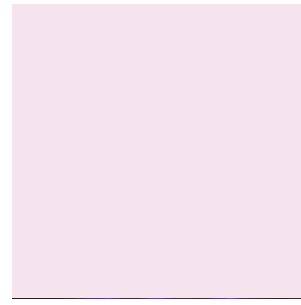
WIREVOXELS

WireVoxels is a Bartlett research project, aiming to challenge the limitation of a 'space frame' in architecture. The research proposes to fabricate building blocks out of robotically bent steel tubes, composed of a limited number of serialised steel elements, with the same connection system for efficient assembly. (Li et al., 2016)



MX3D BRIDGE

MX3D bridge is a project aiming to build a fully functional, intricate steel bridge in Amsterdam, designed to showcase revolutionary technology. Using industrial robotic arms with 3D tools, MX3D prints steel incrementally by welding in place, taking advantage of the ability to work outside a square box. (MX3D, 2018)



DAEDALUS PAVILION

Ai Build's 'Daedalus Pavilion' is a 3D printed architectural installation. This was part of NVIDIA's GPU Technology Conference in September 2016. In partnership with ARUP Engineers this project was aimed to showcase how the future of construction will be transformed by robotics and artificial intelligence. (AI Build, 2016)

ROBOTICS

3D PRINTING

GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

ROBOTICS

3D PRINTING

GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

ROBOTICS

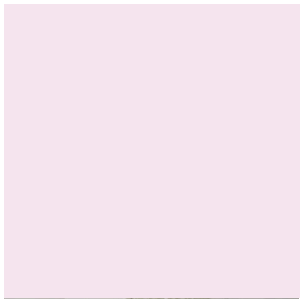

3D PRINTING

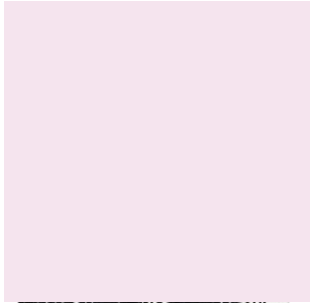
GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

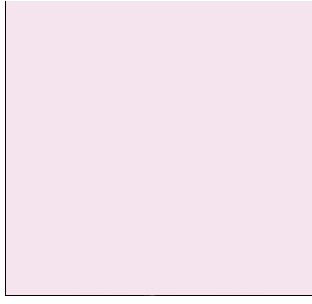
1.02 PRECEDENTS

Project Description			
	COMPOSITE WING	ICD PAVILION	SPACEWIRES - FILAMENTRICS
Research Areas	Composite Wing, by Studio Roland Snooks is an multi-agent design of a network of beams. Robotically fabricated from foam and silicon, provide the necessary structural depth for the 3mm thick fibreglass to be self supporting. For this project, large foam bodies were milled, and the smaller, robotically 3D printed. (Snooks, 2014b)	The University of Stuttgart's 2014 research pavilion focused on a biomimetic investigation and, development of robotic fabrication methods for fibre reinforced polymer structures. Taking biological influences from the 'Elytron' a protective shell for beetles wings as a role model for structure and system. The design was then fabricated using two robotic arms to weave a series of panels to be assembled together. (ITKE, 2014)	'Spacewires - Filamentrics' is a Bartlett project focusing on digital design and fabrication. Within the project there is a section on robotic 3D printing. The focus is on pushing the fabrication method beyond space frame structures and beginning to build a non-linear framing structure. This was then digitally scaled, a series of medium scale prints were completed as proof of concept. (Jiang et al., 2014)
	ROBOTICS	ROBOTICS	ROBOTICS
	3D PRINTING	3D PRINTING	3D PRINTING
	GENERATIVE DESIGN	GENERATIVE DESIGN	GENERATIVE DESIGN
	BIOMIMICRY	BIOMIMICRY	BIOMIMICRY
	AGENT-BASED SYSTEM	AGENT-BASED SYSTEM	AGENT-BASED SYSTEM



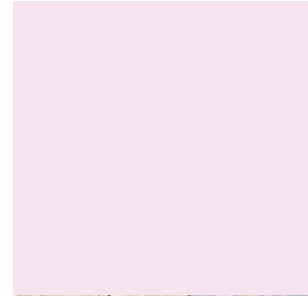
*FREE-FORM 3D
PRINTING*

Free-form 3D Printing: A Sustainable, Efficient Construction Alternative, is a research thesis by Armano Papageorge, investigating 3D free-form printing using computational tools and an industrial robotic arm. The primary branch of this thesis was the viability of the printing as a construction system, however the research involved rigorous testing and exploration of 3D free-form printing. (Papageorge, 2018)



AGNTENSE

Agntense, by Satoru Sugihara is an installation based of simulations of swarm agents and tension/compression forces. The simulations generate a stable tensile wire structure and a self organised form layer by layer. (Sugihara, 2016)



CURVVOXELS

Bartlett's 'CurVoxels' project explores the method of 'voxelization' from a traditional masonry influence in a new perspective and applying it to digital fabrication. Using a series of 3D printers on the end of an industrial robot, a 'design-fabrication integration system' was formed. Resulting in a series of digital models at various scales and printing medium scale models as proof of concept. (Kwon et al., 2015)

ROBOTICS

3D PRINTING

GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

ROBOTICS

3D PRINTING

GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

ROBOTICS

3D PRINTING

GENERATIVE DESIGN

BIOMIMICRY

AGENT-BASED SYSTEM

1.03

COMPLEXITY THEORY

Biomimicry, where flora, fauna and entire ecosystems are emulated as a basis for design is an area of research that brings driving principles and innovation to design. Nature is separate to architecture in a way that it does not focus on aesthetics, but optimal and perfect ordering of systems.

The systems in question in this research are agent-based systems, a form of emergence. Which are also referred to as a 'complex system'. A complex system is created through the interaction of multiple small, simple elements that combine to become behaviour that is greater than the sum of its parts. This self organising behavior generates complex forms and phenomena. (Perony, 2013)

'Natural processes rely solely on unique geometry and material properties, unlike mechanical systems and assemblies'

(Oxman, 2015)

In science, the word emergence refers to the production of forms and their behaviour, by systems that have an irreducible complexity. The structures are patterns that emerge via collective actions of many individual entities. Where they are the sum of their parts based on their autonomous interactions with one another. (Minar, Burkhart, Langton, & Askenazi, 1996)

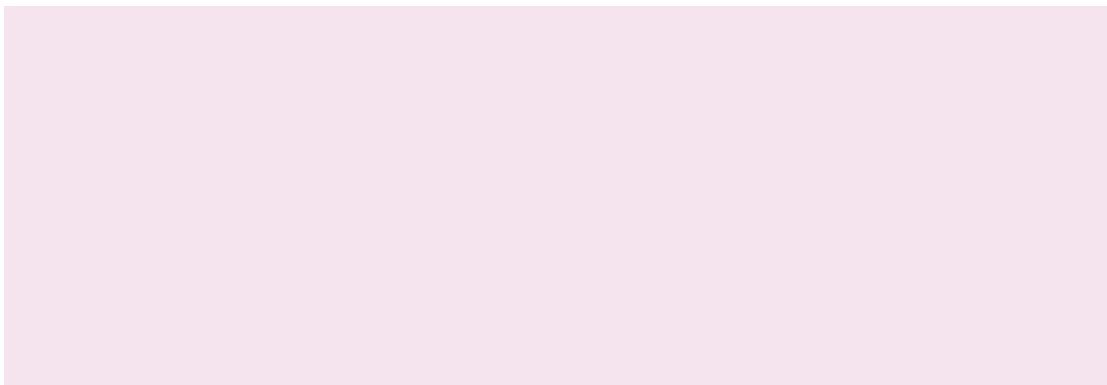
Complex structures can be found in many natural phenomena, for example the shape of weather patterns, as well as development and growth of complex, orderly crystals and movement patterns of wildlife such as termites, ants and birds.

Rather than focus on biological influences and structures, studying and simulating systems allows the processes to be quantified. There are many examples of agent-based algorithms being used all over the world in a variety of fields including architecture. Using them to optimise space using mathematics, populating areas exactly where structure is needed.

In the field of architecture, swarm intelligence focuses on the role of agency within iterative design processes. Using parametric programs to create high populations of self-organised elements into emergent intelligence. For example, visualisation, form finding and self-organisation.

“Complexity Theory offers a new epistemology, a new way of understanding patterns and structures that we observe in the world, as the emergent properties of iterative parallel actions of very simple autonomous processes. The complex outcome emerges entirely bottom-up from the multiple interactions between these events”.

(Popov, 2010)



*Figure 4. Fractal snowflake
(Bentley, 1902)*

*Figure 5. Termite cathedral
mound (Yap, 2005)*

*Figure 6. Symphony of the Stones
carved by a river (Jim, 2005)*

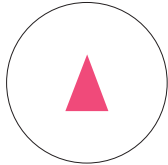
AGENT-BASED SYSTEMS

Computational multi-agents systems were first developed by John von Neumann as cellular automata in the 1940s, and further developed by John Conway's Game of Life in the 1970's (Waldrop 1993). These self organising systems influenced the work of many computer programmers in the 1970's, including the introduction of multi-agent systems by Paul Coates. However Craig Reynold's research holds the most connection to the computational understanding of swarm systems and developing models of this. (Reynolds 1987).

Reynold's Boids algorithm simulates the natural flocking behaviour of birds and fish. The agents, or '*boids*' respond to their surrounding agents, the environment of their current state.

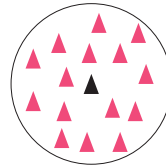
In the field of this research, simulations/plugin and example software and code have a range of names including agents, boids, and swarms. Each refers to the phenomenon that Reynolds simulated when deriving key elements of agent-based systems.

Each agent holds properties key to simulating systems computationally. Autonomy, Solidarity, Expandability, Resilience and Awareness are aspects of agent behaviours carried from natural system research. Awareness of each agent is the key concept that must be considered when manipulating simulations for design intent.



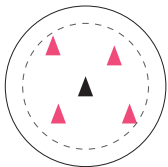
AUTONOMY

To self coordinate, each member must operate as an autonomous master (not as a slave)



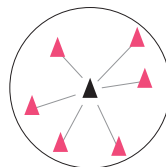
EXPANDABILITY

The system must permit dynamic expansion where members are seamlessly aggregated



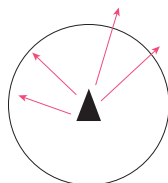
AWARENESS

Each member must be aware of its surroundings and abilities



RESILIENCE

When members are removed, the system must be self - healing



SOLIDARITY

When a task is completed, members should autonomously look for a new task

1.05

AGENT-BASED SYSTEMS CONT.

Other variables are also added into digital simulation to build an agent-based model. These simulations are vector based, through many simple vector calculations run iteratively the movement of each agent is calculated.

Speed is another factor than can be programmed into flocking algorithms, however in basic models each agent will typically move at a constant speed.

In order to achieve even more diverse collective behaviours the following rules can be added:

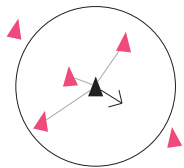
Acceleration: agents that are not part of a flock can fly faster to catch up, while birds that have flock mates adjust their velocity to the average velocity of their flock mates, and as a result the whole flock slows down

Chase, avoid and eat rules: these birds were developed in order to further diversify the simulation. There are two types of birds: predator and prey. The predators can diverge from their flock and chase the closest prey : prey agents try and avoid any predators in their field of view.

Obstacle avoidance: very similar to the separation rule. The agents detect the closest obstacle in their 'cone of vision' and determine the distance between themselves and the obstacle. They as they approach it they turn to veer away from it, retaining smooth movements. (Nikolay Popov)

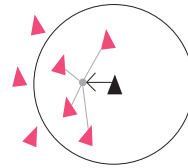
“the swarm consists of a number of individuals who are normally in a reflexive relation on to another – and will show emergent structure as groups of boids flock together into aggregates, then break up and reform”

(Paul Coates, Programming Architecture p88-89)



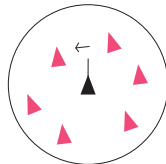
SEPARATION

The ability to maintain a certain distance from all or a select number of others nearby.



ALIGNMENT

The ability to align with all or a select number of others nearby characters.



COHESION

The ability to cohere (approach and form a group) with other nearby agents. Steering for cohesion is computed by finding all local agents and calculating the average position of the nearby agents. A force is then applied in the direction of that average.

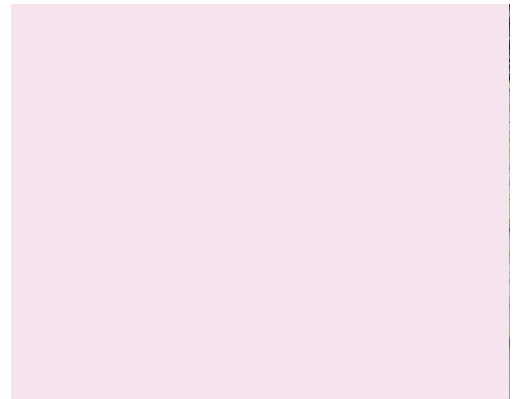
This algorithm is an example of the bottom-up nature of swarms, these systems function without any global control, relying solely on the interactions of agents close to one another. Philip Ball describes that within agent-based simulations ‘the coherent group behaviour emerges from simple, purely local interactions between individuals, who have no sense what the whole group is doing’ (Ball, 2009, p 128).

Focus on agent-based design, and its ability to rethink architecture through interactions, and its relation to many precedent projects in architecture and digital fabrication, began to define the critical development of research in this thesis.

STIGMERGY

The concept of Stigmergy was written by Pierre-Paul Grasse in the 1950's to describe the indirect communication between social insect societies, such as ants and termites. Through his research, Grasse demonstrated that the regulation and coordination of the activity of the insects does not depend on the individual workers themselves, but is mainly achieved by the nest. (Heylighen, 2016). A study into the reconstruction of termite nests showed a stimulating configuration triggers a response of a termite worker, transforming the construction of the figuration into another figuration, which in turn may trigger another, possibly different action performed by the same termite or any other in the colony. (Bonabeau, 1999)

Taken as an underlying element of agent-based behaviour, Stigmergy is able to provide a way for the designer to program and investigate the form formation. Taking Gasse's understanding of one individual worker having an on flow effect, and interpreting this as rules of design and architectural limitations, a form will begin to self organise and iterate. Using an informed agent-based system allows for a final design that is self organised in a way that it meets a design brief, but can also achieve a result with an array of surprising and unpredictable results stemming from the same initial input.



*Figure 8. Ant paths built from pheromone traces
(Karatay, 2007)*

VOLATILITY

As a forefront member of a group of both industry and education based practitioners researching agent-based systems, one of Roland Snooks' arguments of the strength of agents is their volatility.

volatile; characterized by or subject
to rapid or unexpected change *(Snooks, 2014)*

Complex systems have a volatile nature; through intensive processes of formation and self organization, and are capable of catastrophic change. The instability of agent-based systems enables an output that is radically different from its initial conditions. Examples such as Stephen Wolfram's cellular automata rule 110. Rule 110's behavior operates on the boundary between stability and chaos. It is similar to the cellular automata evolution Conway's, the game of life, within the conditions of rule 110, new characteristics and situations emerge from a constant state of flux, without any predictable order. Wolfram describes this as an unquantifiable phenomenon. It is these emergent capabilities within complex systems of iteration that has solidified computational design as an indispensable tool for generating complexity. (Wolfram, 2002)

Translating these possibilities within mathematical phenomena to architectural design, they can be developed as elements of risk or experimentation. Generally computational design within architecture is associated with parametric design and complex forms, shapes and sizes of great difficulty for one person to model on CAD software. These projects hold no element of experimentation, with a clear structure and relationships, using precise control and manipulation of digital models. Agent-based simulations sit in a category of unexpected outputs and change, basing a design or system on simulations opens possibility for the unexpected design to come through, while still holding the same design properties intended.

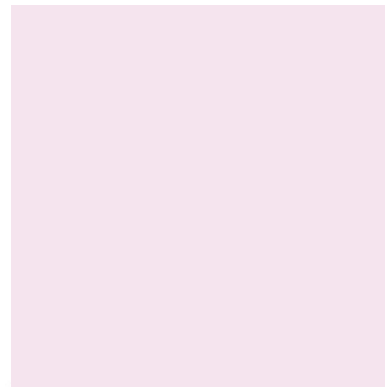


Figure 9. Rule 110, Wolfram (2002, p. 851)

APPLICATION OF AGENT-BASED SYSTEMS IN ARCHITECTURE

Agent-based simulations, also referred to as Swarm Intelligence, are principles which have been used and applied to many projects in architecture over the last two decades, and can generally be split into a series of topics:

Swarm space optimisation, swarm modelling, swarm urbanism, and swarm robotics (Petrš, 2016).

- Swarm space organisation – refers to the simulation of architectural spaces, examples such evacuation routes and space occupancy use agent-based simulations to optimise paths within a building. Agent-based simulations can be used for designing the width of routes, obstacle avoidance and smoothness of paths.

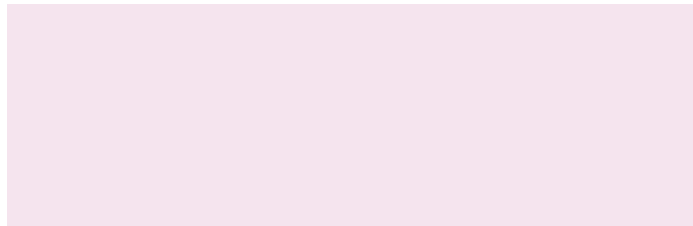


Figure 11. Swarm space organisation (Buš, 2016)

- Swarm modeling – agent-based systems are commonly used to build form of interesting behaviour. A multitude of programs and simulations are available as open source such as Processing, Grasshopper, NetLogo etc. Using secondary plug-ins for these such as Boid Library for Grasshopper and Plethora Project for Processing it is possible to achieve comprehensive agent-based simulations with basic knowledge of coding.

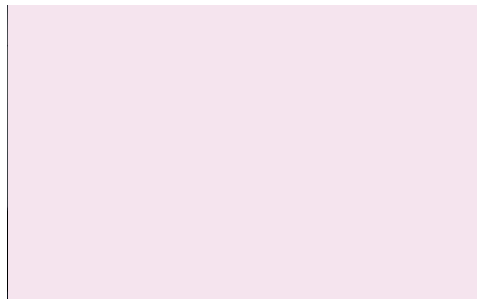


Figure 10. Composite Swarm (Snooks, 2013)

- **Swarm urbanism** – ‘*Emergence: The Connected Lives of Ants, Brains, Cities and Software Emergence*’ is a book written by Peter Merholz in 2002. Focussing entirely on the concept of swarm intelligence in urbanism. Merholz describes five fundamental principles observed of ants, and their bottom-up building principles. (Merholz, 2002)
 - *More is different* – a change in critical mass of ants is key for properties of emergence to appear
 - *Encourage random encounters* – Merholz describes how ants react to the activities of other ants to usefully modify their behaviour
 - *Ignorance in useful* – ants act as individual components, acting as singular beings but as a whole. Ants have no need to understand the colony as a whole, but an understanding of their individual tasks
 - *Look for patterns* – ants follow the behaviours and pheromones left by other ants
 - *Pay attention to your neighbours*

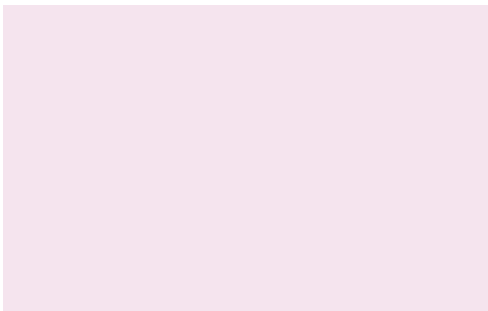


Figure 12. (Snooks, 2009)

GENERATIVE DESIGN IN THIS THESIS

Generative design in architecture follows a cycle through problem specification, design generation and design evaluation. With an initial system set up, the designer (architect) is able to interactively control it's behaviour by specifying the design intent that needs to be embedded in the system. This intent can represent constraints that must be met, a desired condition or logic that should be followed. The input data will have different implications for how the generative system will work.

As a designer, a key part of this system is the **evaluation**. A requirement of evaluations of each design and model is key in informing future generations. This thesis is not based on a computer system that is able to control and design itself, but a system that has an aspect of '**designer evaluation**'.

With this in mind, the following experiments and designs will each be explored generatively, with intent behind each simulation and model

1.10

GRASSHOPPER ALGORITHMS

Grasshopper a graphical algorithm editor integrated with Rhino3D modeling tools. Grasshopper is a visual based programming language, allowing designers to build form generators without knowledge of programming or scripting. Grasshopper has an open-source library of plug-ins available for download. BOID Library, is a swarm behavior library that works with agent locations (as Grasshopper Points) and agent vectors. Using a looping function, the component calculates motion vectors based on the agent behavior. Each vector is aggregated into a compound vector used to inform all of the agents.

This simulation requires a set amount of inputs, and manipulation parameters that will impact the swarm output.

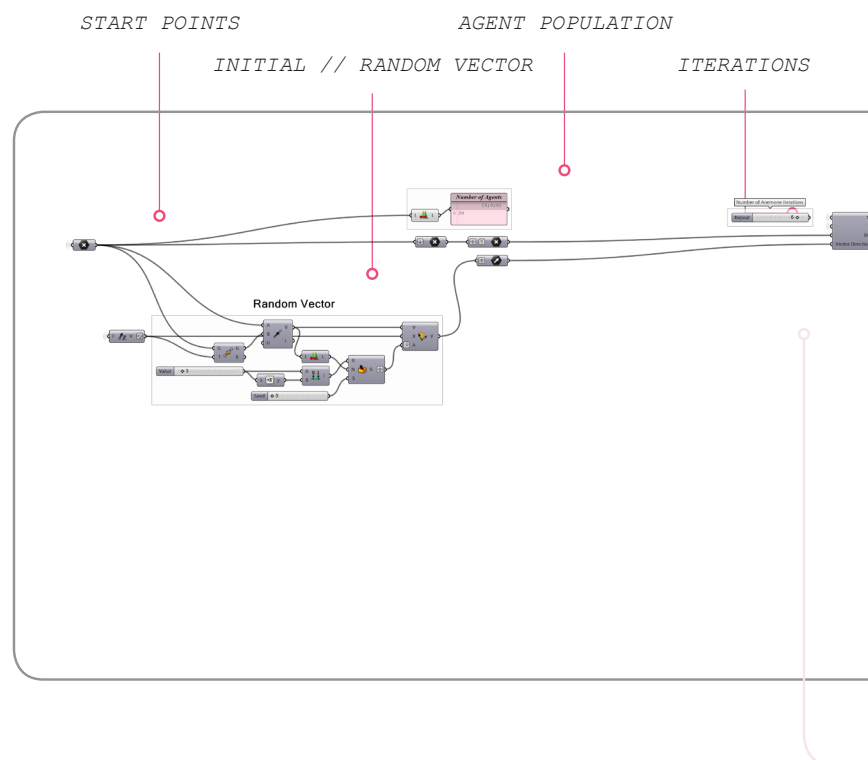
This simulation requires a set amount of inputs, and manipulation parameters that will impact the swarm output. The first phases of digital research and simulations was centered around a Grasshopper script, with the inputs and outputs described below.

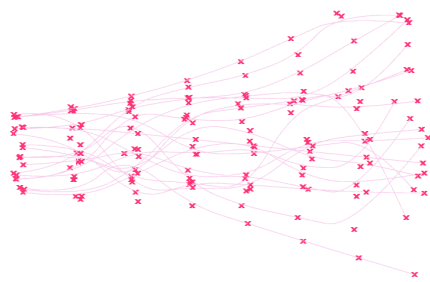
INPUTS

- Population start points
- Agent population
- Initial Vector
- Random wander vector
- Number of Iterations

PARAMETERS

- Directness of initial vector
- Adhere data
- Repulse Data
- Max Speed
- Containment Area





The simulation gives an output in the form of a series of points - as with all agent-based simulations. For the purpose of the following simulation examples, each of the points have been interpolated with a curve - this is to help visualise the movement of the agents in the space and show the relationships between individual agents.

Figure 13. Grasshopper Points to Curves

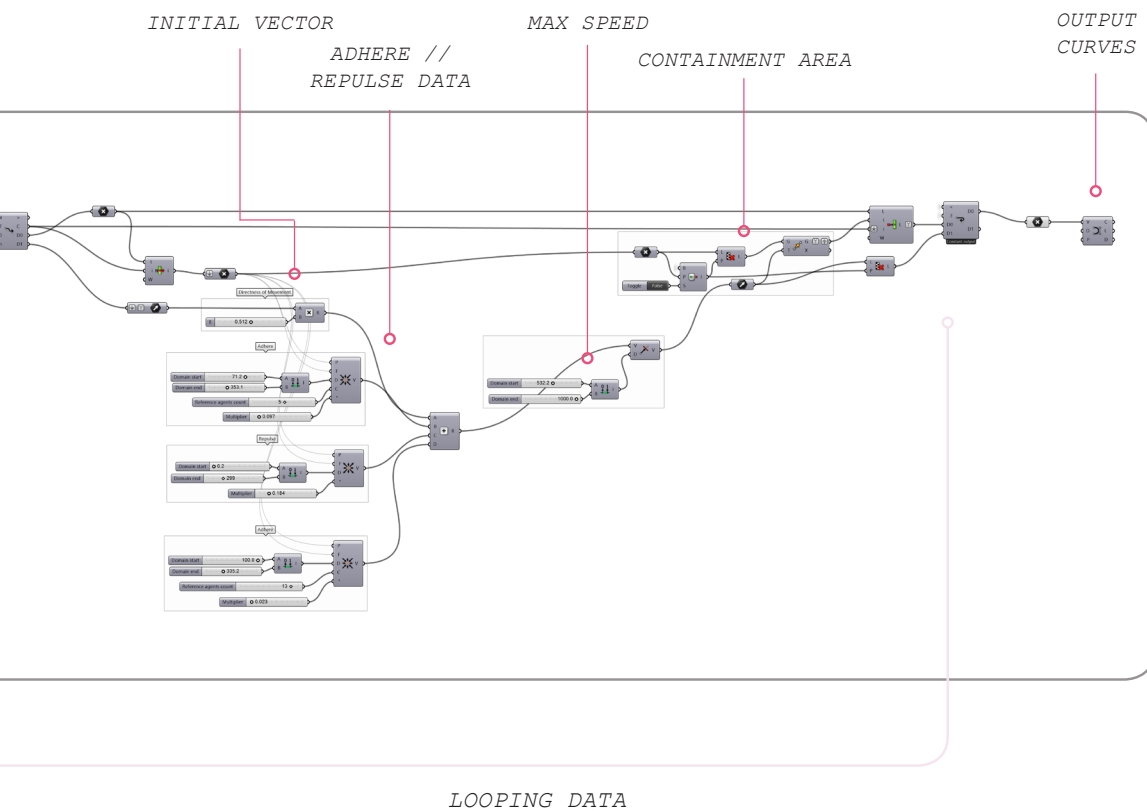
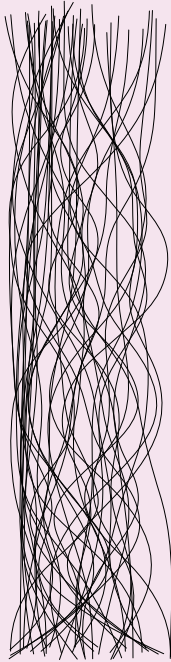


Figure 14. Grasshopper Script



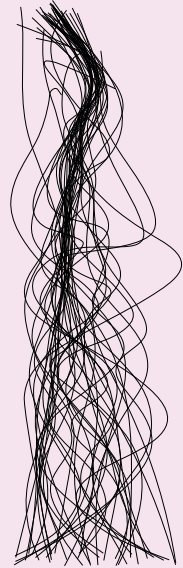
Iteration	1	Adhere	0.023
Adhere	0.1	Count	13
Count	5	Agents	40
Repulse	0.113	Strength	1



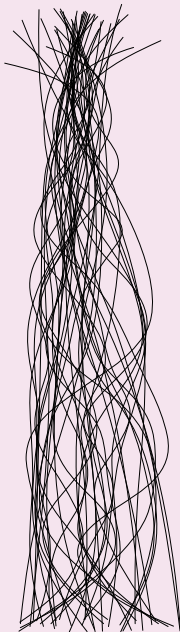
Iteration	2	Adhere	0.203
Adhere	0.1	Count	13
Count	5	Agents	40
Repulse	0.113	Strength	1



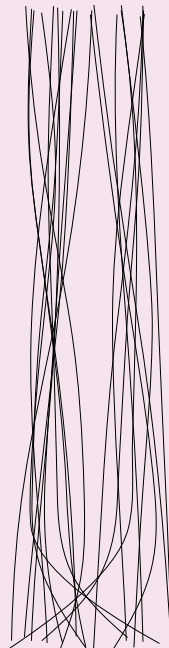
Iteration	3	Adhere	0.203
Adhere	0.267	Count	13
Count	5	Agents	40
Repulse	0.113	Strength	1



Iteration	4	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	40
Repulse	0.113	Strength	0.184



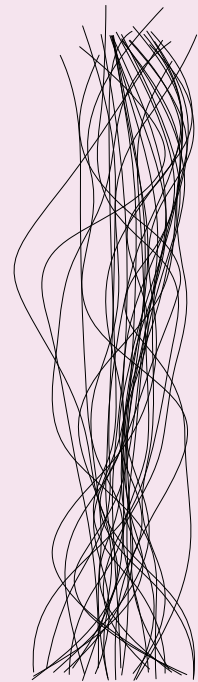
Iteration	5	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	40
Repulse	0.113	Strength	0.415



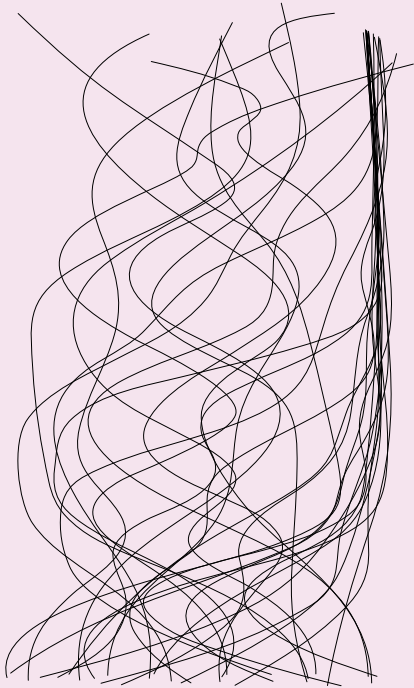
Iteration	6	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	20
Repulse	0.113	Strength	1.0



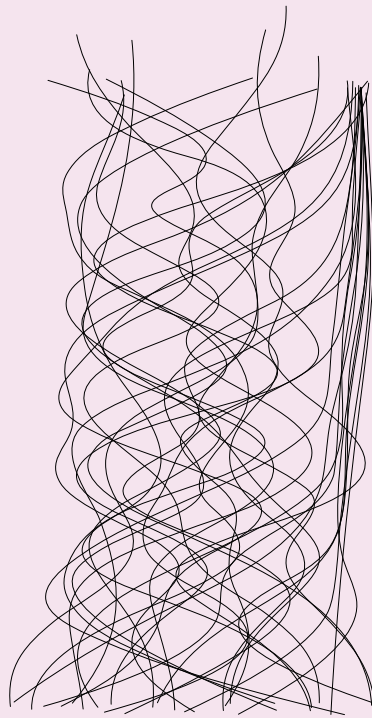
Iteration	7	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	40
Repulse	0.113	Strength	1.0



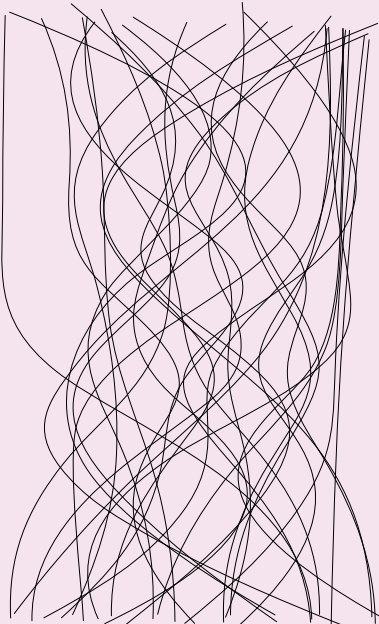
Iteration	8	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	40
Repulse	0.426	Strength	1.0



Iteration	9	Adhere	0.203
Adhere	0.267	Count	3
Count	10	Agents	20
Repulse	0.426	Strength	1.0



Iteration	10	Adhere	0.093
Adhere	0.144	Count	3
Count	2	Agents	30
Repulse	0.236	Strength	1.0

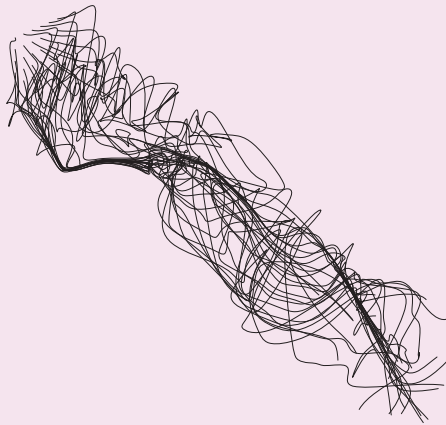


Iteration	11	Adhere	0.013
Adhere	0.055	Count	3
Count	2	Agents	30
Repulse	0.066	Strength	1.0

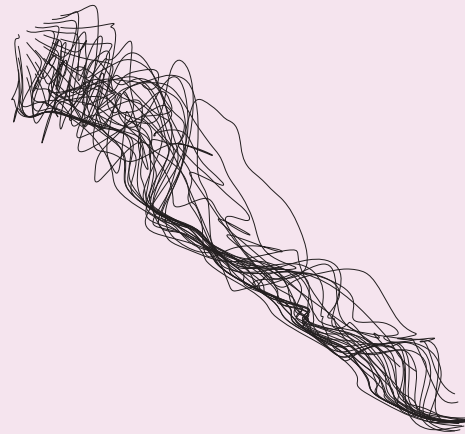


Iteration	12	Adhere	0.013
Adhere	0.055	Count	3
Count	2	Agents	30
Repulse	0.150	Strength	0.8

These simulations experiment with the three control elements of agent-based simulations - separation, alignment and cohesion. With a change in element strength, analysis distance and agent populations, a different density effect is achieved in two dimensions.



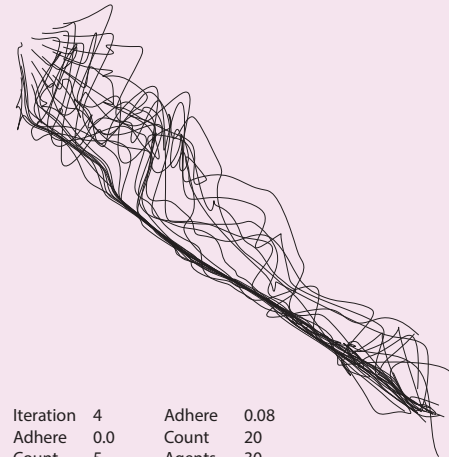
Iteration	1	Adhere	0.08
Adhere	0.0	Count	11
Count	5	Agents	30
Repulse	0.32	Align	0.197



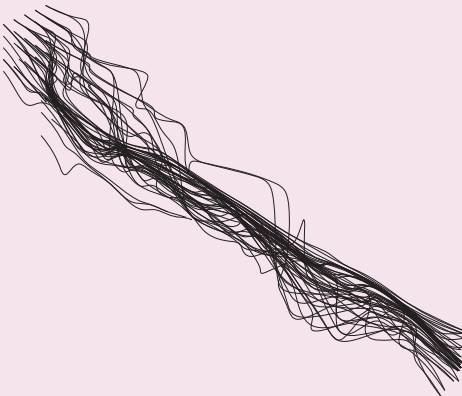
Iteration	2	Adhere	0.08
Adhere	0.0	Count	11
Count	5	Agents	30
Repulse	0.3	Align	0.197



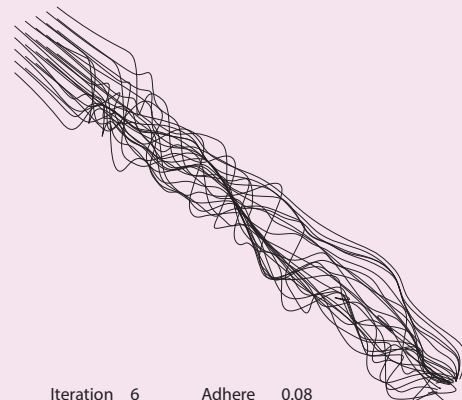
Iteration	3	Adhere	0.08
Adhere	0.0	Count	11
Count	5	Agents	20
Repulse	0.32	Align	0.197



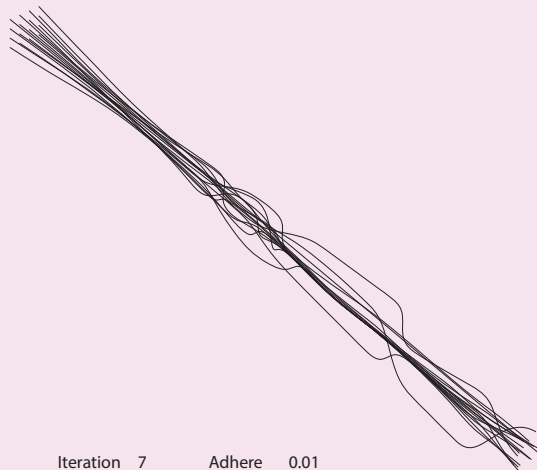
Iteration	4	Adhere	0.08
Adhere	0.0	Count	20
Count	5	Agents	30
Repulse	0.3	Align	0.197



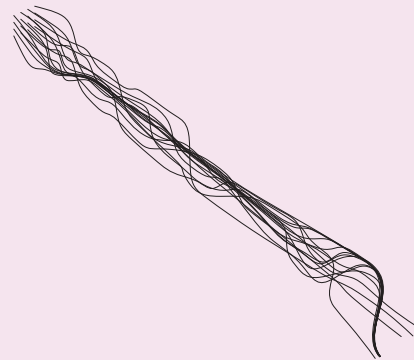
Iteration	5	Adhere	0.08
Adhere	0.0	Count	20
Count	5	Grid Base	
Repulse	0.32		



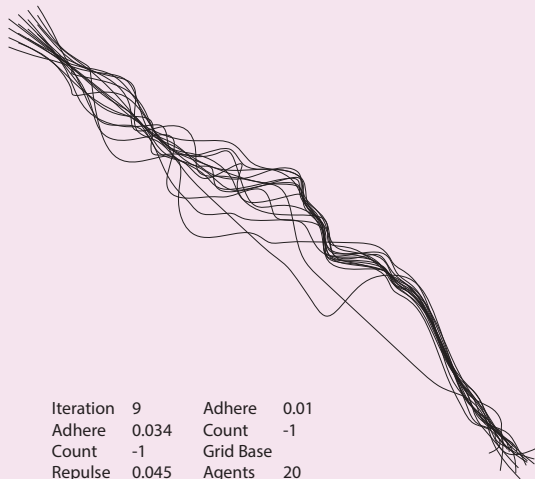
Iteration	6	Adhere	0.08
Adhere	0.1	Count	20
Count	5	Grid Base	
Repulse	0.32		



Iteration	7	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Grid Base	
Repulse	0.045	Agents	20



Iteration	8	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Grid Base	
Repulse	0.2	Agents	20



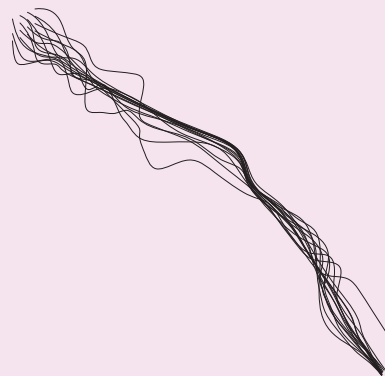
Iteration	9	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Grid Base	
Repulse	0.045	Agents	20



Iteration	12	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Random base	
Repulse	0.045	Agents	30



Iteration	11	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Grid Base	
Repulse	0.045	Less Iterations	



Iteration	10	Adhere	0.01
Adhere	0.034	Count	-1
Count	-1	Grid Base	
Repulse	0.045	Increase in search	

PHYSICAL SWARM MODELING

Using the described simulations in Grasshopper, a range of outputs were produced, these shown on the previous spread, are examples of the copious differences that are able to be achieved when changing simulation parameters. As this research continues multiple layers of parameters will be included in the simulations and each model will adopt different characteristics.

With each result sitting in digital space, it is difficult to visualise how these will translate to a physical model. In using agent-based systems as the driver for a fabrication system, it is key to consider how these will assemble physically.

Figure 18 shows a series of simulation results that have been traditionally 3D printed. These models were fabricated to visualise the thickness and density of the models, as well as the impact of the areas of the model that are unpopulated and points are greatly separated from each other.

When these models are modelled as free-form structures, the overall form will be significantly different, however the same density will show through. A physical demonstration of the simulations is also useful in understanding the impact of each of the parameters and how the models truly are in 3D space.

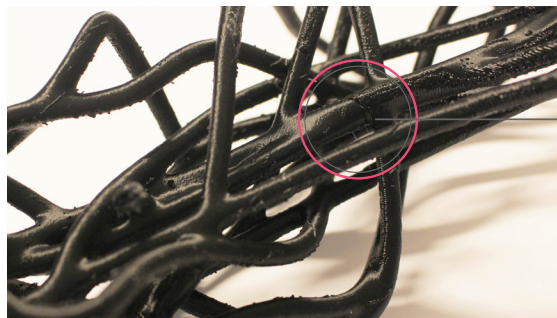
The thickness and body of each of these models has been achieved using a voxel-based geometry mapper. Chromodoris is a plug-in for Grasshopper which is used for the creation and manipulation of meshes. The points from each of agent simulations are translated into voxels (3D pixels in the form of boxes), these are then smoothed to build iso-surfaces, three-dimensional objects that bring through the density of the points and conglomerate these points into geometry with a certain thickness.

Thin agent paths broke during fabrication - could have impact on potential free-form printing models if not sufficient material buildup

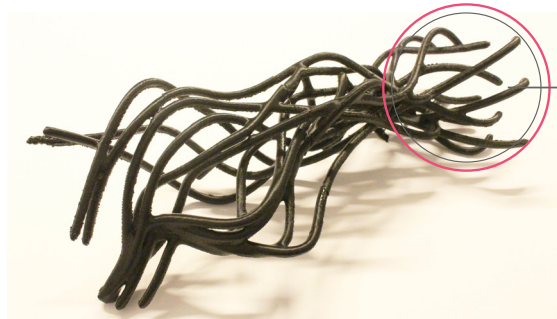


Multiple agents in one area build up thickness of print

Crossing of agent paths results in dense intersection area - consider how intersections are designed in free-form models



Multiple strands crossing// joining - precise ordering of this system will be key in free-form printing



Simulation begun from a grid of points opposed to random points. Builds stable footing for print to stand with.





Figure 16. Traditional 3D Print of swarm

1.12

VECTOR INFORMED SIMULATIONS

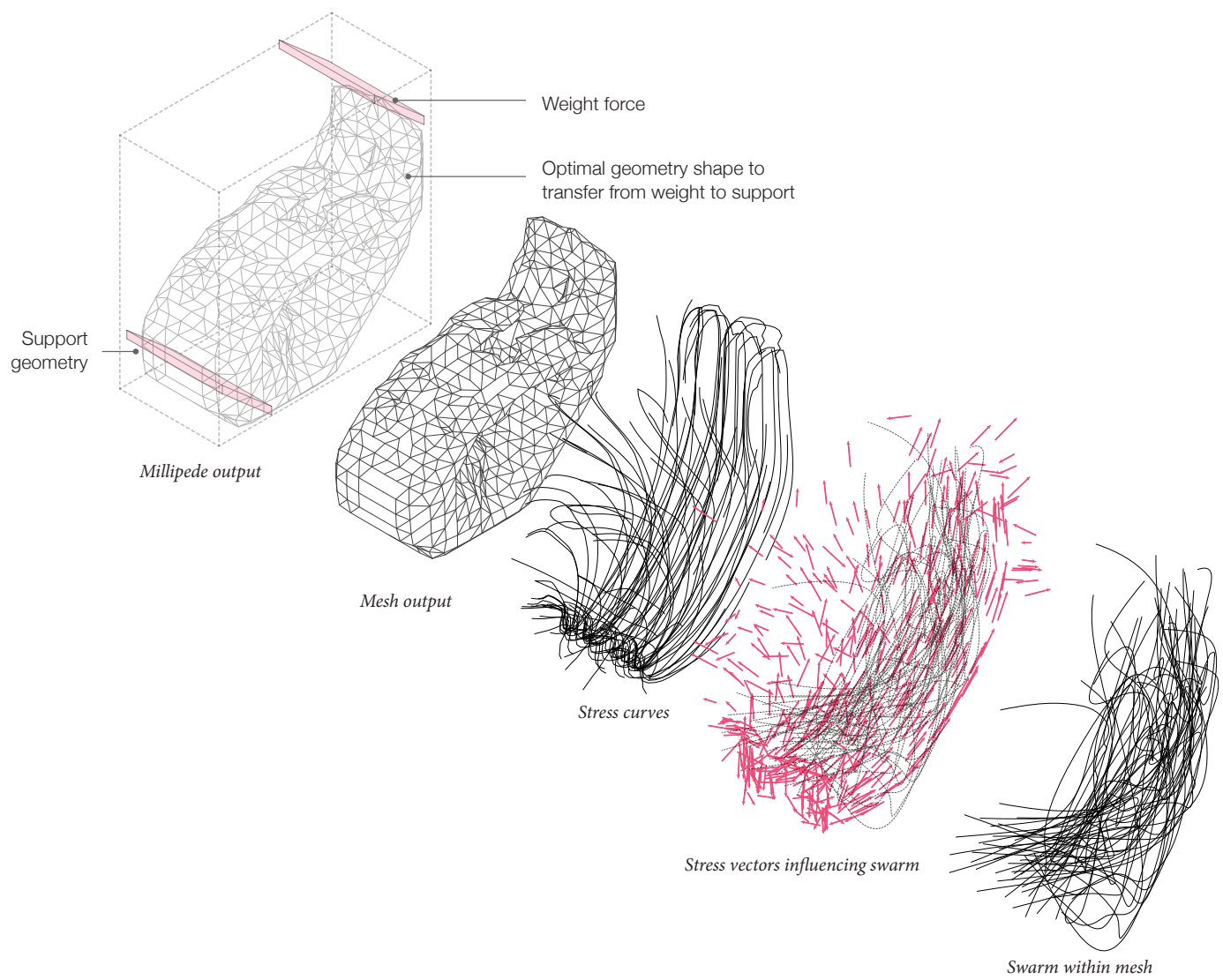
The key part of this thesis is the relation of this design methodology to the realm of architecture. Multi-agent algorithms have a unique script for each simulation, each simulation also has the ability to be encoded with architectural design intent within an architectural process.

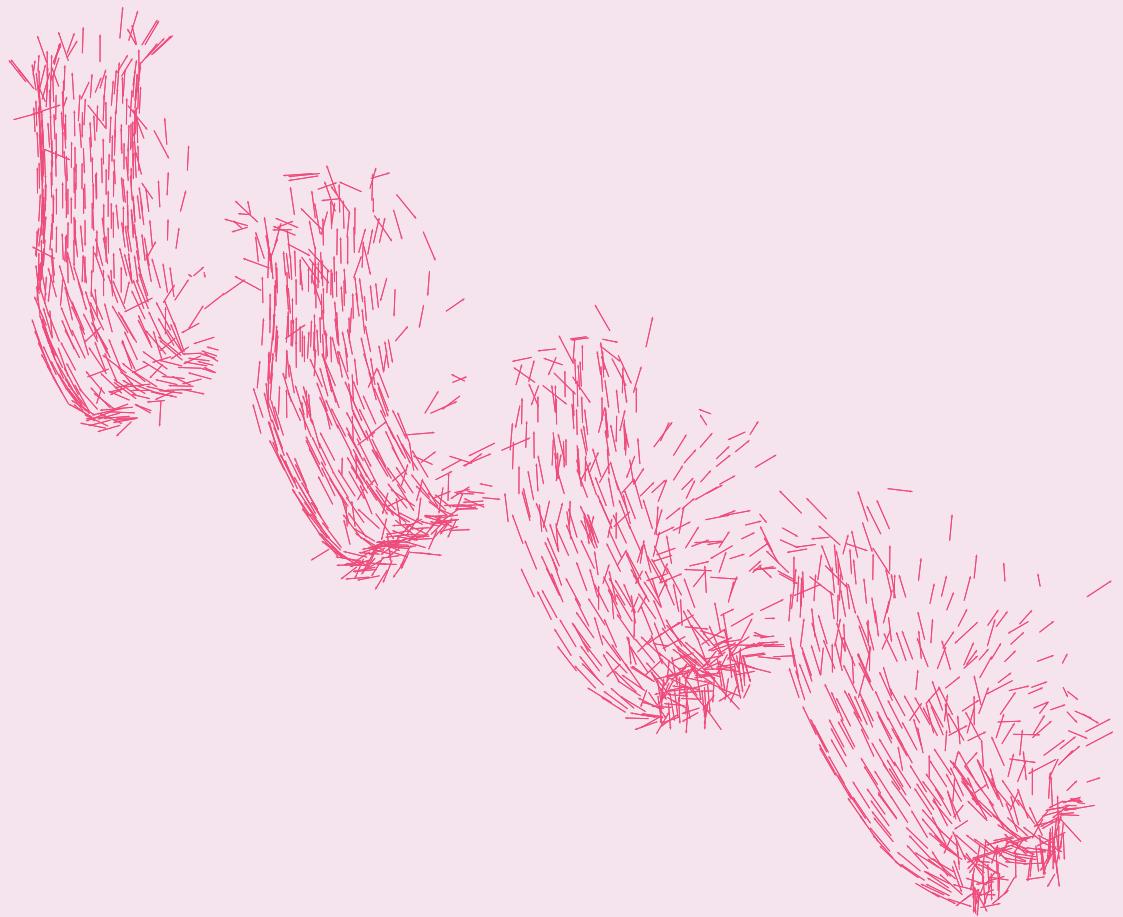
A feedback loop throughout a simulation allows evaluation and influences to inform future movements of agents. The previous simulations appear to have arbitrary behaviours that self-organise to create emergent patterns and forms. While they have local interaction between each agent, they are not specific to architecture or a design goal. This can be given a further level of detail, with evaluation and influences assessed against architectural objectives of the simulation. This results in an emergent formation that is fit for architecture.

These objects are able to include behaviours with the intention of creating a specific outcome, for example where agents self-organise to building a system that performs as the designer prescribed.

The following examples take a step towards encoding architectural intent within the agent-based simulations. With each simulation holding both point and vector data, these are also able to be manipulated, giving the agents information and tasks to fulfill. Similarly to assigning agent parameters on the simulation, the agents are also able to be informed by vectors and path networks, as described as a possible example of swarm urbanism. Figure 17 depicts using a series of vectors as informants for how the agents will move through a field. These vectors are outputs from a structural analysis tool. Millipede – is a form finding analysis plug-in for Grasshopper, producing a model **“of the most structurally efficient natural form”**.

The principal stress vectors that are used to build the geometry between the support and the weight are used as secondary guidance vectors for the swarm – added to the initial vectors input of the data. The use of these vectors is an example of how secondary elements can be included to inform the overall simulations. In the way that this informs the swarm, many other architectural, environmental or structural information is able to be passed to the simulations – giving almost infinite possibilities of form generated using agent-based simulations.





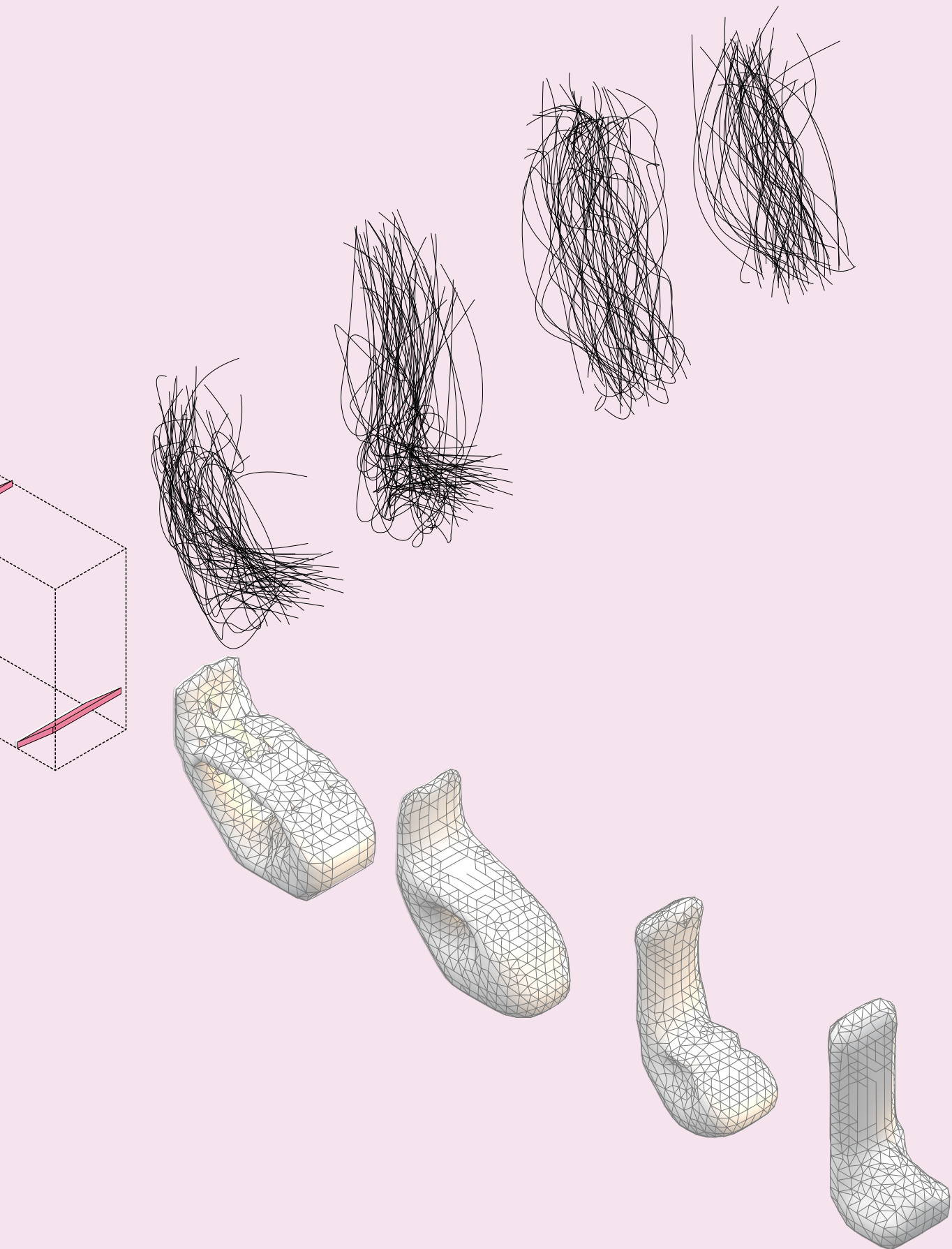
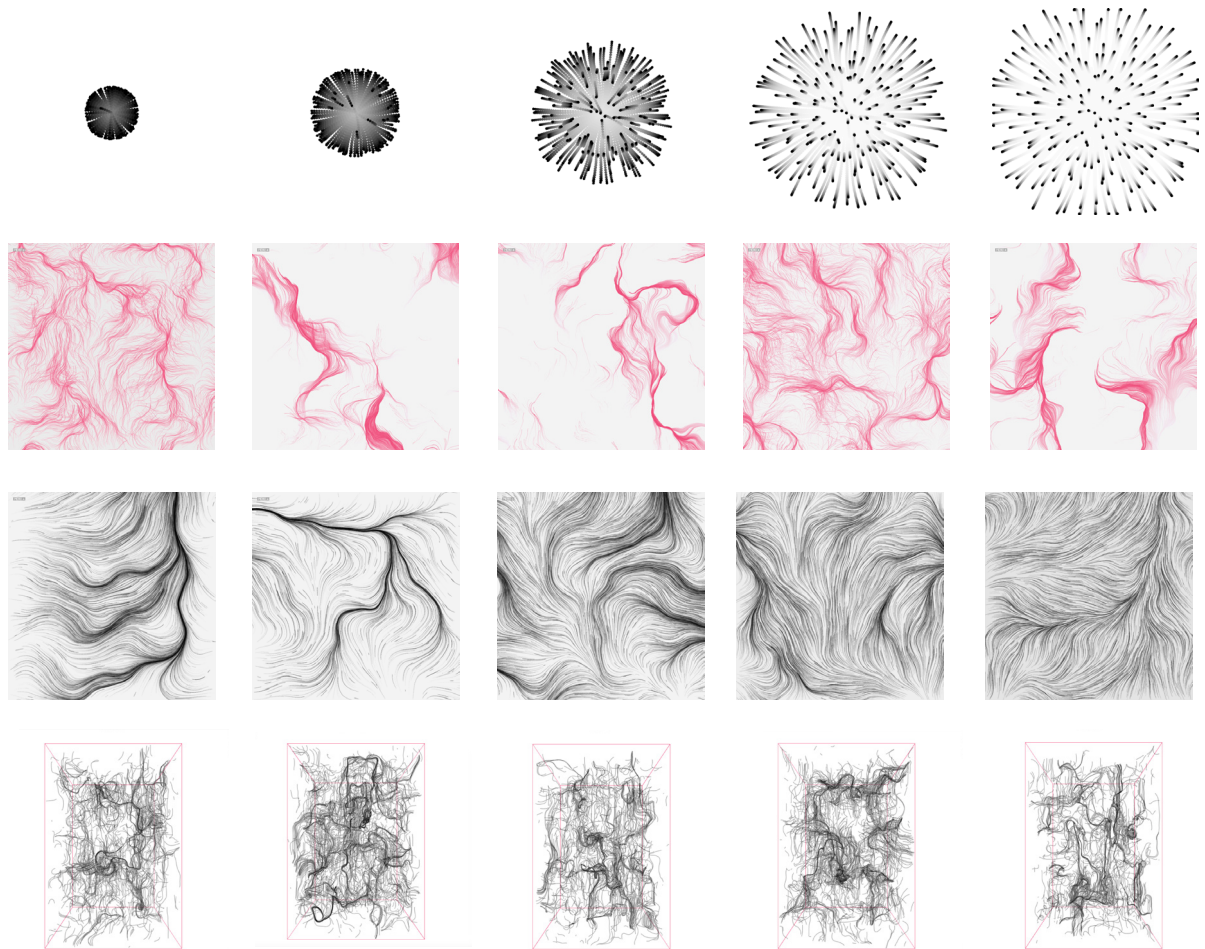


Figure 18. Vector informed swarm matrix



1.13

PROCESSING

Processing is an open source graphical library and integrated development environment built for visual arts. Processing uses the Java computer language to compute and simulate visuals, producing visual outputs, each simulation is able to contain text data about the simulation. Processing will be used for the next phase of the project to further explore agent-based simulations. The move from Grasshopper to Processing allows for more customization and informed decisions, with control over the specific code itself. (Taron & Parker, 2014)

Within the scope of the project an amalgamation and learning of Java programming will allow the next stage of the research to progress.

Daniel Schiffman's book, the Nature of Code

“focusses on programming strategies and techniques behind computer simulations of natural systems”,

working towards using Processing to build a foundation for experiments in generative design.

This text has been used as a driving force behind both the understanding of algorithms as coming from nature, and a core learning device that has allowed to write the code itself.



Figure 20. Daniel Schiffmans, 'The Nature of Code'

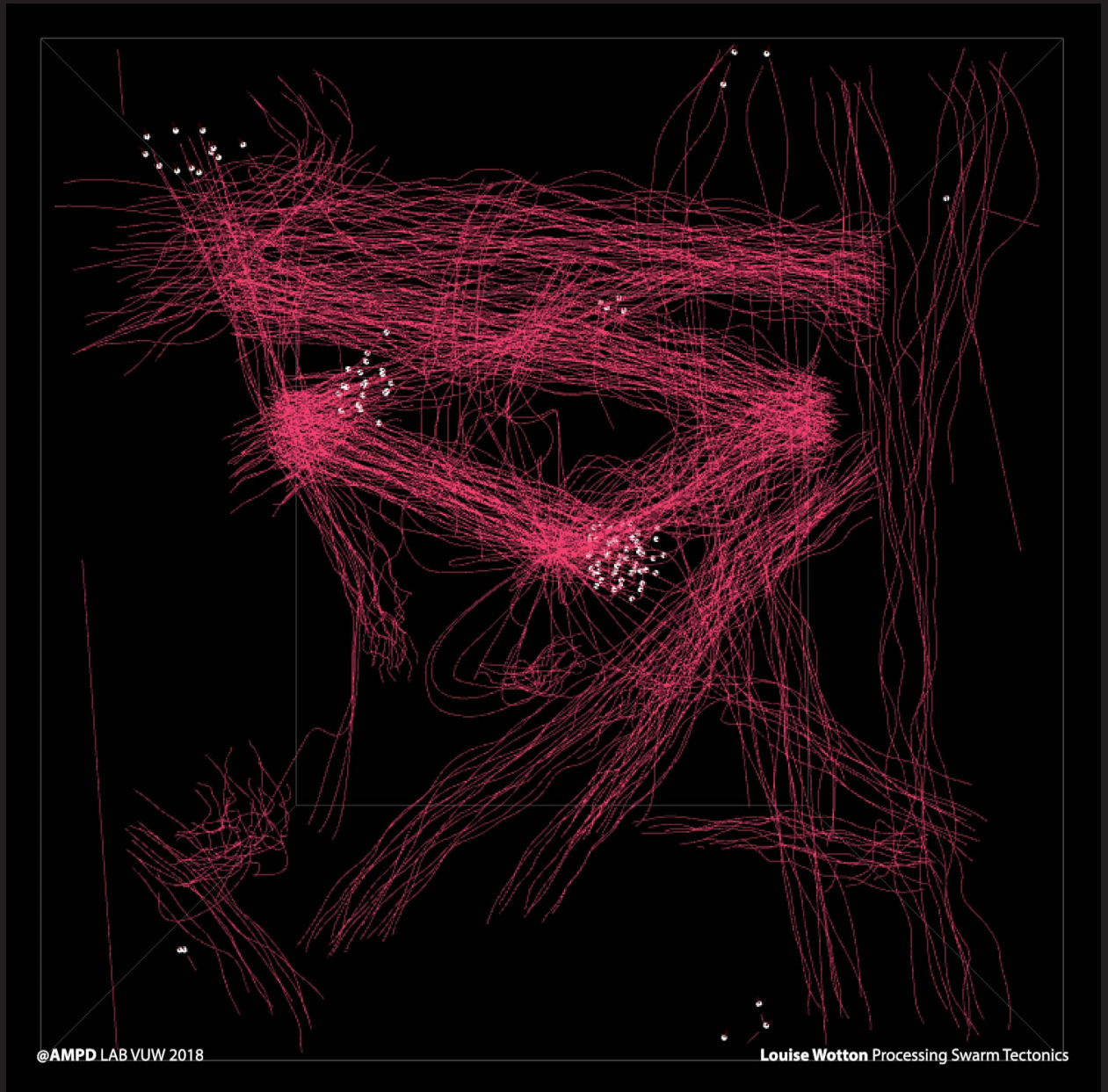
PLETHORA PROJECT

Plethora Project, is a library of code for Processing. Developed by the design studio of the same name, by Jose Sanchez - an architect and programmer based in Los Angeles. Part of his research has been around the simulation of agent-based systems, and creating an accessible library for people to simulate without programming the specifics of a physics system. (Sanchez, 2016)

“...is a library of open source code for the simulation of steering behaviors... based on Craig Reynolds *Boids* behaviors for the study of complex adaptive systems.”

This library was used as a base model for the next phase of the project, it has been used by hundreds of researchers in the study of complex adaptive systems. The design of the code is very easy to adapt re-model, allowing for hundreds of possibilities and entities that are able to be altered.

Having no base knowledge of Java, it was essential to use this as a building platform to begin simulations that could be transferred to a 3D model. Over time the simulations run were added to, and customised, but this library was the core model of the steering behaviours as well as the visualisation techniques.



1.15 PROCESSING CODE

A range of digital experiments were run in Processing using the Plethora Project library, the following controls are set up at the beginning of each simulation and are able to be manipulated to achieve the intent desired. Each control has a large impact on each experiment and are individually coded and edited.

AGENT CONTROL

1. Agent Behaviours

Implementation of Separation, Alignment and Cohesion controls

2. Start Points of agents

As found through the Grasshopper simulations of agent-based behaviours, the start points of the agents have great impact on the behaviours. If the agents start in a grid based configuration rather than a random coordinate range, the result reflects agent output. Due to the fact that there is no number of agents starting closer to one another than others in the population.

3. Population of agents

Similarly to the start points, this has an impact on the distance range between each agents if there is an increase on the amount of agents starting within a given area, there will then be less distance between each agent if the start area is not increased accordingly.

4. Initial Vector of agents

In most cases this is in the Z direction, however some simulations are run with a minor X and Y random number to encourage angular movement and the agents to interact.

5. Maximum Speed

The maximum velocity the agent can reach. This is a vital factor as the agents speed up when moving as a group. The script works in a looping manner, so the speeds compound over a large amount of iterations.

PROCESSING JAVA CODE

```
float a=1.2; //cohesion  
float b=0.2; //alignment  
float c=1.1; //separation
```

```
float a1 = 70; //cohesion dist  
float b1 = 200; //alignment dist  
float c1 = 50; //separation dist
```

```
pa.flock(boids, a1, b1, c1, a, b, c);
```

```
int [] start_points = new int[pop];
```

```
v = new Vec3D (random(-100, 100), random(-100, 100), -300;
```

```
int pop = 15;
```

```
Vec3D initialVelocity = new Vec3D (random(-0.1, 0.1), random(-0.1, 0.1), 3);
```

```
pa.setMaxspeed (4);
```

CODE DEVELOPMENT**Stage 1: Control of agent behaviours using Plethora Project**

The beginning of a base level script sets up all the parameters, and the section shown here describes the *for loop* (looping function name) and calls each agent, applying both behaviour parameters and visual display attributes.

Stage 2: Introduction of steering controls and attractors

Similarly to informing the Grasshopper simulations with vectors. Processing tests are able to be controlled using points and target strengths. Each are written at the beginning of the code as float values, to be called on later in the looping process.

Stage 3: Manipulation of attractors

Beginning to use agent-based systems for architectural intent requires the programmer to manipulate points of steering controls and designated areas to populate. This stage will be fulfilled in the following part of the thesis: Part 2: Digital Application.

PROCESSING JAVA CODE

```
for (int i = 0; i < boids.size(); i++) {
    L_Agent pa = (L_Agent)boids.get(i);

    stroke(239,75,122);
    strokeWeight(1.0);
    pa.drawTrail(100);
    pa.flock(boids, a1, b1, c1, a, b ,c);
    pa.setMaxspeed(4);
    pa.dropTrail(1,5000);
    pa.bouncespace(DIMX/2, DIMY/2, DIMZ/2);
    pa.updateTail(7);
    strokeWeight(Connectivity);
    stroke(239, 75, 122, 240);
    pa.drawLinesBetweenTails(boids, d, e);
    strokeWeight(5);
    stroke (255);
    pa.displayPoint();
pa.update();
}
```

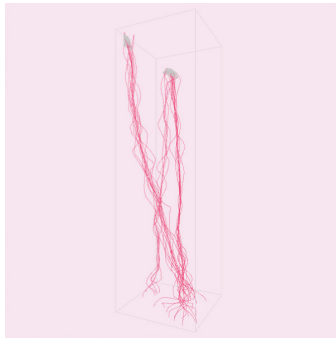
```
float target X = -100;
float target Y = 150;
float target Z = 0;
float strength = 0.0;

float target X2 = -300;
float target Y2 = 400;
float target Z3 = 0;
float strength 2 = 0.9;

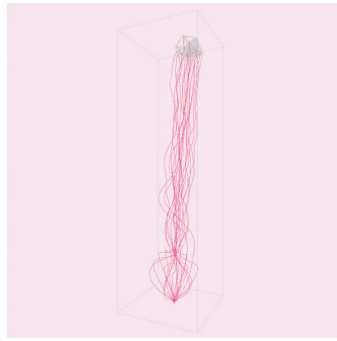
pa.seek(target, strength2);
```

Stage 1: Control of agent behaviours using Plethora Project

*Figure 22. Swarm based simulations
in Processing*



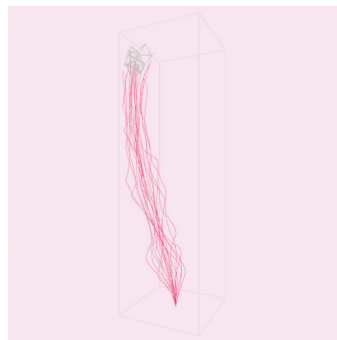
Start: Rndm Dist 1:63 Dist 2: 200
Dist 3:30 Flock: 1.2, 0.2, 1



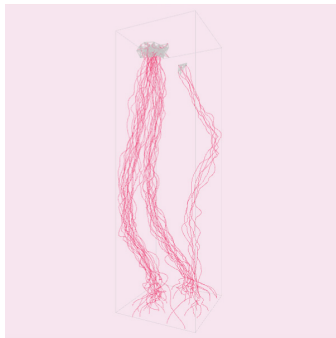
Start: Point Dist 1:120 Dist 2: 40
Dist 3:30 Flock: 1.5, 1, 3



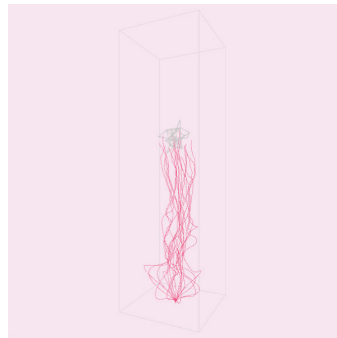
Start: Rndm Dist 1:70 Dist 2: 200
Dist 3:20 Flock: 1.2, 0.2, 5



Start: Point Dist 1:120 Dist 2: 40
Dist 3:340 Flock: 1.5, 1, 2



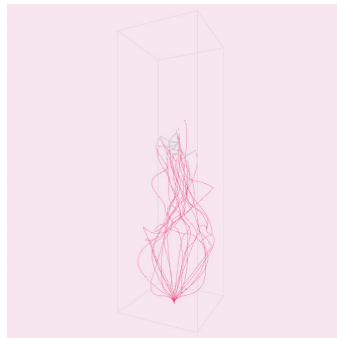
Start: Point Dist 1:70 Dist 2: 200
Dist 3:20 Flock: 1.2, 0.2, 5



Start: Point Dist 1:120 Dist 2: 40
Dist 3:340 Flock: 1.5, 1, 3



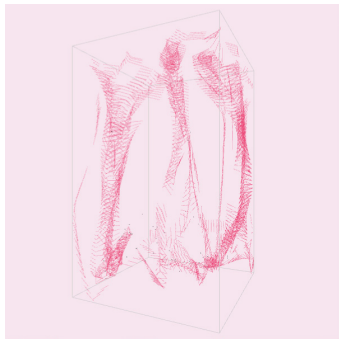
Start: Rndm Dist 1:70 Dist 2: 200
Dist 3:20 Flock: 1.2, 0.2, 5



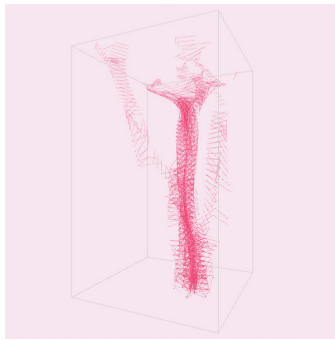
Start: Point Dist 1:120 Dist 2: 40
Dist 3:40 Flock: 1.5, 1, 3

Stage 1: Control of agent behaviours using Plethora Project

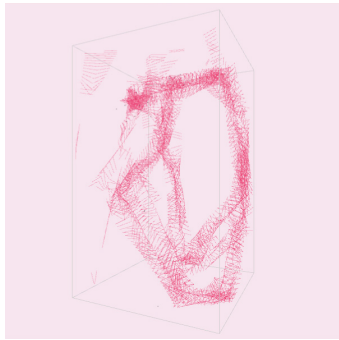
*Figure 23. Swarm based simulations
in Processing: connectivity graphs*



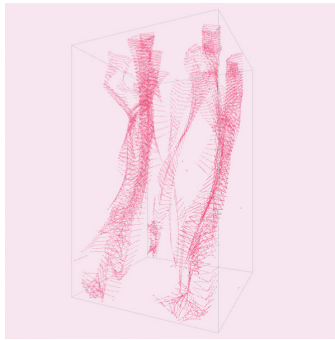
Start: Rndm Dist 1:100 Dist 2: 120
Dist 3:200 Flock: 1.2, 10., 1.2



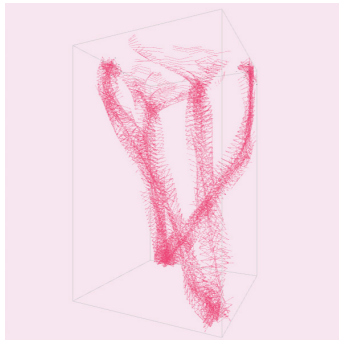
Start: Rndm Dist 1:150 Dist 2: 100
Dist 3:70 Flock: 1.2, 1.0, 1.2



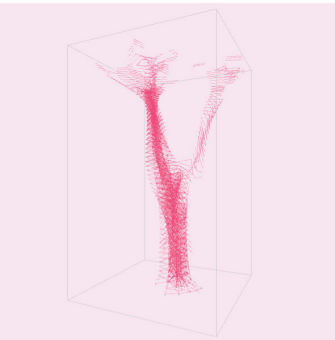
Start: Rndm Dist 1:100 Dist 2: 100
Dist 3:70 Flock: 1.2, 1.0, 1.2



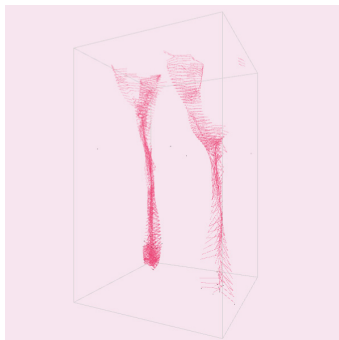
Start: Rndm Dist 1:100 Dist 2: 80
Dist 3:200 Flock: 1.2, 1.0, 1.2



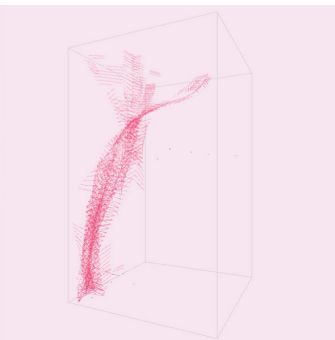
Start: Rndm Dist 1:150 Dist 2: 100
Dist 3:70 Flock: 1.2, 10., 1.2



Start: Rndm Dist 1:150 Dist 2: 100
Dist 3:70 Flock: 1.2, 10., 1.2



Start: Rndm Dist 1:100 Dist 2: 100
Dist 3:70 Flock: 1.2, 1.0, 1.2



Start: Rndm Dist 1:100 Dist 2: 100
Dist 3:100 Flock: 1.2, 1.0, 1.2

Stage 2: Introduction of steering controls and attractors

*Figure 24. Agent-based simulations using point based orientation:
experimenting with gravitational manipulation*



DATA COMMUNICATION

Transferring this data back to Rhino + Grasshopper was essential in developing the models for 3D fabrication. The code used to program an industrial robotic arm is formed using HAL Robotics, a plug-in for Grasshopper.

Both Processing and Grasshopper have similarities that they work in a 3D space using coordinates, allowing easy communication between the two softwares. While there is no plug-in or software that will allow simple data transfer, the data has been transferred using text files and simultaneously importing into Grasshopper. Through a series of reformatting, the point data from Processing is able to be interpreted in Grasshopper immediately.

PROCESSING DEVELOPMENT

While the Plethora Project library gave a base model for complex simulations, in order for the model to be simulated with design intent, rework of the base script was essential. Through a series of gravitational experiments, connectivity diagrams, and interactions with 'external' features, base knowledge of Processing and Java grew so that sections of the code were able to be rewritten. These changes are able to cater to the needs of both an architectural model, and a 3D printed one.

At this stage in the research, the digital simulations moved towards a specific intent, giving detail to the design. This is separated into two sections:

FABRICATION ELEMENTS

Manipulation of start points - a series of start points to control space populated during simulation

Introducing agents later in the simulation to add density

Removing agents from simulation to increase porosity, and removing agents that travel outside a bounding box - culls future movement and eases data conversion between Processing and Grasshopper

SIMULATION CONTROL

Control of agents in consideration to distance between one another - too many points in one section will cause print path errors.

Encouraging agents to move apart a certain distance and move back together - will limit the spans of the sections of free-form printing without support.

Setting agents to have start points on a plane - ensuring that the print is able to start on a flat surface.

Figure 26 demonstrates the introducing of agents at a certain point in the simulation. Adding items to the *for loop* opens many possibilities when considering encoding the simulations with intent and a desired output.

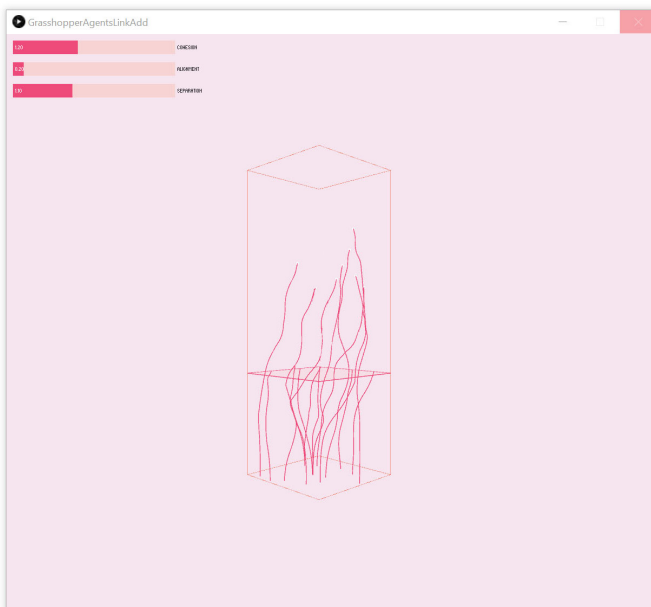
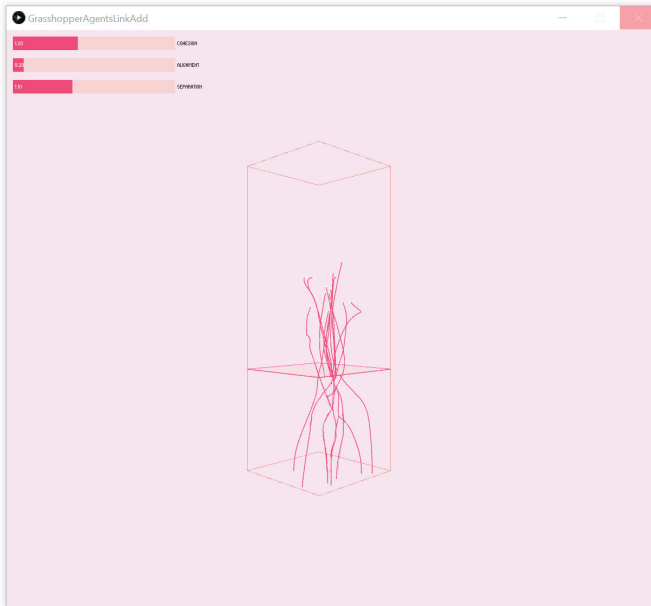


Figure 26. Adding and removing agents from array



DIGITAL AUTHORSHIP

Computation design in architectural, and many other technical fields is growing rapidly. Online databases are available with code libraries, written by people originating from different code-based fields. The nature of architectural design is changing with the inclusion of digital processes, made easier by the availability and on-line sharing of digital data. The result of sharing designs and scripts on-line is the circulation through many research and professional institutions. This prompts discussion around plagiarism and authorship. If designs are adapted, shared and customised, using someone else's script, or tutorial content, is that designer the author, or the original data owner? Opinions would argue that when data is put out into the web, it is intended to be shared, the smallest change to a line of code can cause drastic changes to the design output and be completely unique.

“the creative processes is enriched and informed by computational tools, but those do not substitute the role of the creative mind”.

(ProgramArchitecture, 2013)

AD Magazines Digital Property : Open-Source Architecture issue discusses the realm of authorship within digital architecture. A truly open source architecture involves the sharing of ideas, software and scripts. David Ruy's article, *Serving, Owning, Authoring* focuses on the importance of originality. **“authorship above all requires originality”**. (Ruy, 2016, p.18) While this is an expectation within a creative profession, is it also a legal requirement of copyright law. This is an issue within architecture, but in the digital world, as discussed, small changes can make a great difference.

This is clearly shown in the past chapter where small parameters have been changed and the output has been greatly different. Ruy's viewpoint would argue that within the realm of digital architecture, small changes to digital scripts are not in fact plagiarism. Instead they would be seen as a helping hand forward into the coding and scripting world.

The base sections of the Processing code development in this thesis relied on open source platforms. Data available to anyone with access to the Internet. Without the availability of these, certain sections of the past chapter would be unachievable. Through the use of an open-source database like Jose Sanchez's 'Plethora Project', designs, and the world become more connected, spreading the advancement of knowledge in design.

CHAPTER 2:

DIGITAL APPLICATION

/ Biological Algorithms for Digital Manufacture

INTRODUCTION : NATIONAL SCIENCE CHALLENGE

The National Science Challenge (NSC), which this thesis is partially funded by, is a research based initiative “designed to take a more strategic approach to the Government’s science investment by targeting a series of goals, which, if achieved, would have major and enduring benefits for New Zealand.” (Ministry of Business, Innovation and Employment, 2018).

Based under the Science for Technological Innovation branch, this research aligns with the materials, manufacturing and design areas. With representatives from Scion and the Universities of Auckland and Waikato, this research aims to test materials and printing techniques. The results from this section of the research will be passed back to these organisations to coincide with their research.

Through collaboration with the NSC, a physical application of the agent-based systems were proposed. The developed digital code would be used to populate facade areas for sun shading, focusing on the efficiency of the shading and its relation to a environmentally informed design.

The application of the digital system gives the geometry architectural and environmental intent, adding a layer of detail and information to the work flow.

The sunshades will be simulated and modelled using agent-based systems, where printing techniques will be tested against a series of model strengths and controls. The experiment performances will be analysed using environmental software, focussing on sunshade performance across an entire day, as well as certain hours of the day.

The aim for the results is to be able to optimise the form structure, and begin to design the sun shade with dynamic properties in mind for future development and speculative design. Through the NSC, focus is on the development of materials and technologies, with hope to take this research of shading goals and parameters throughout the day, and develop materials that are able to respond to environments and the structure become dynamic and responsive through material expansion and contraction. The output itself will build a digital foundation and provide data and models to Scion and the Universities of Auckland and Waikato.

2.2

SUN SHADING

Daylighting and sun shading as an area of architectural design is one that involves a high level of scientific testing and/or rules applied. This section of this research aims to investigate agent-based algorithms where they are able to be used with intent to shade an area of a facade throughout the day.

daylighting; the use of light from the sun and sky to complement or replace electric light

To accurately simulate and test the digital models as sun shading systems, a local weather file of Wellington was used. This weather file (an epw file), is able to be input into Ladybug, a Grasshopper plug-in of a collection of applications that support environmental design. This open source software imports and analyses standard weather data, and is able to draw diagrams presenting this analysis. For the purpose of this research Ladybug will be used to determine the levels of sun shading that a structure would provide across a given day, and year.

2.3

LADYBUG FOR GRASSHOPPER

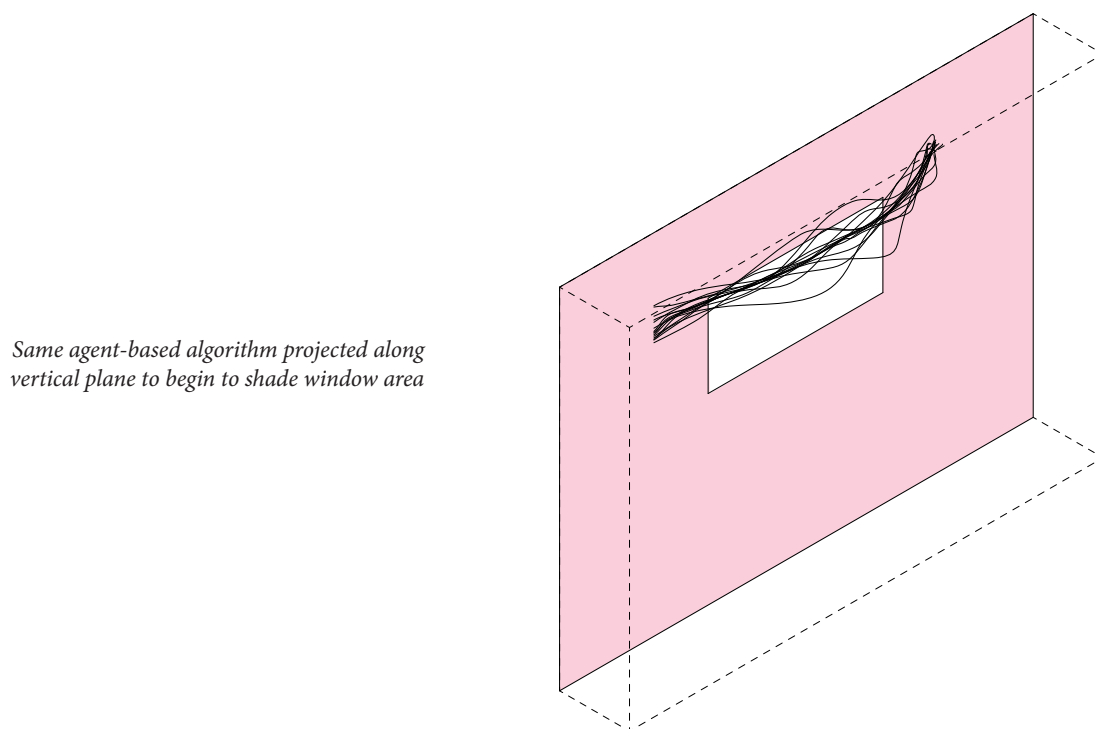
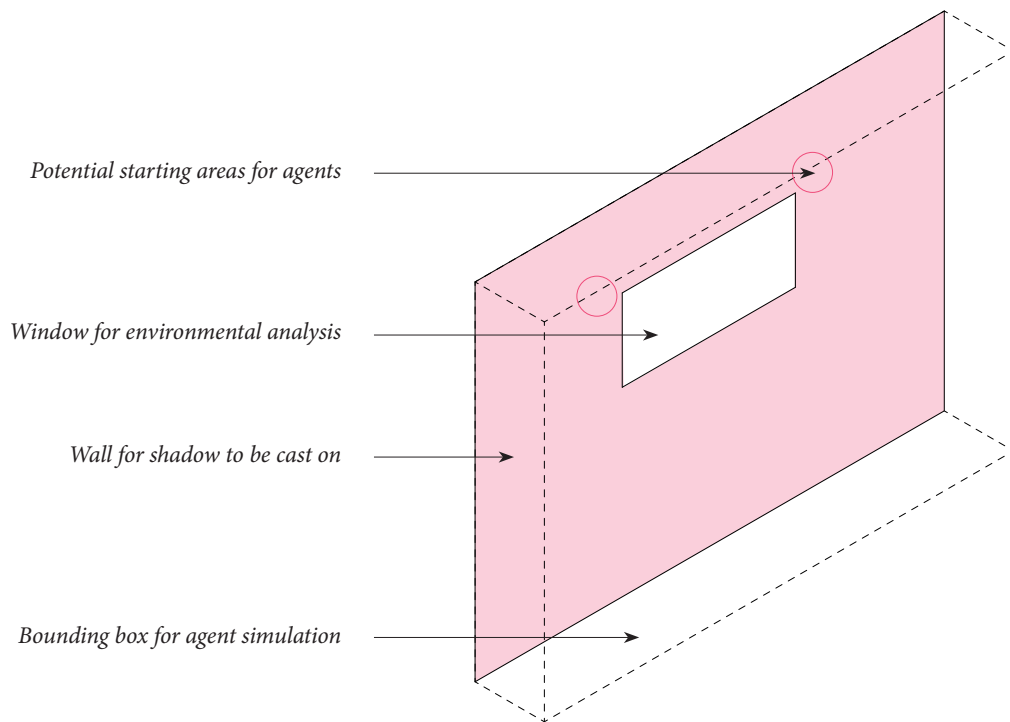
Figure 28 shows a simple model set up of a window/wall situation, which is used to both analyse sun data and simulate the agents within.

This model serves as a base, primitive sunshade, the core concepts grasped from this phase of the project was the concept of the shading across an entire day, and then to determine what time of day that the sunshade would be needed for. If this system was to be put on the exterior of an east facing office building for example, this would suggest the afternoon sun would be the harshest, so the agent simulations could be programmed to populate the area that would shade the interior of a building in the mid to late afternoon.

Similarly in the way that Processing data is transferred back to Grasshopper for fabrication purposes, the data is also transferred back for analysis purposes with Ladybug.

- The cross software communication allows for a quick transfer of the Processing sketch through to Grasshopper for analysis.

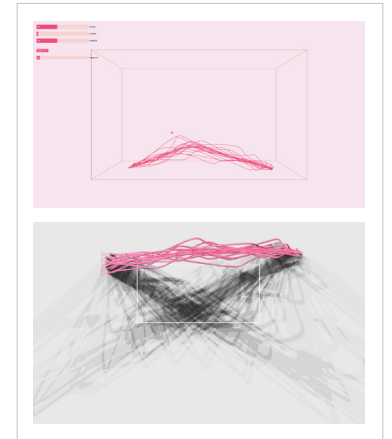
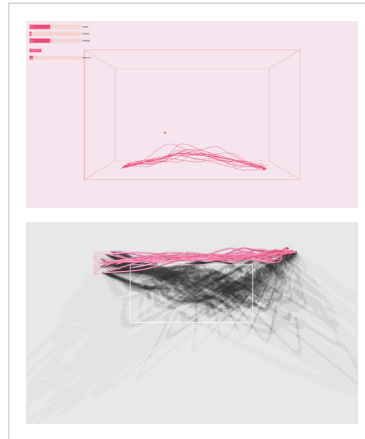
Figure 28 shows analysis of a series of Processing simulations across a single day. With attraction points being manipulated to transfer the bulk of the model to move the shade area. While this had little impact, increasing the population of the agents increased the amount of shade, however this impacts material use, so it is necessary to investigate a model where this can be optimised.



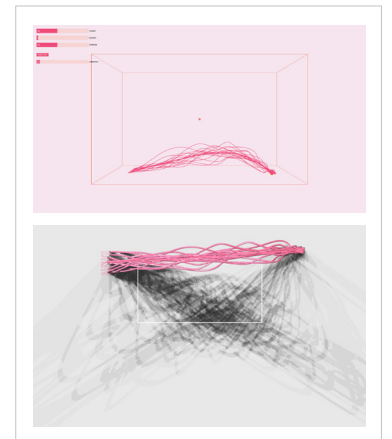
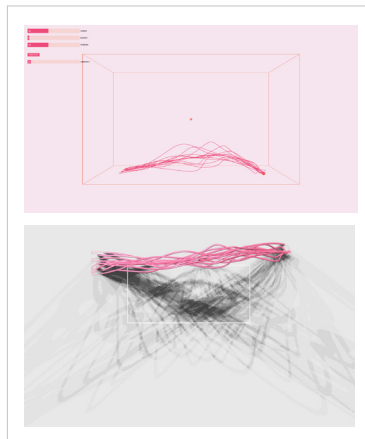
15

20

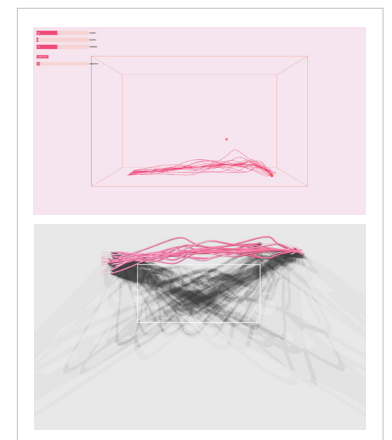
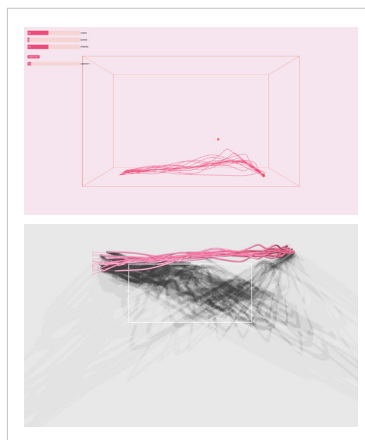
West attraction point



Central attraction point



East attraction point



POPULATION

25

30

35

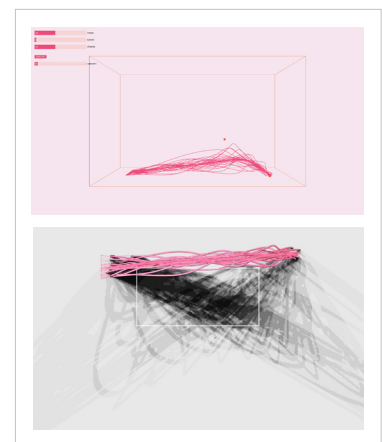
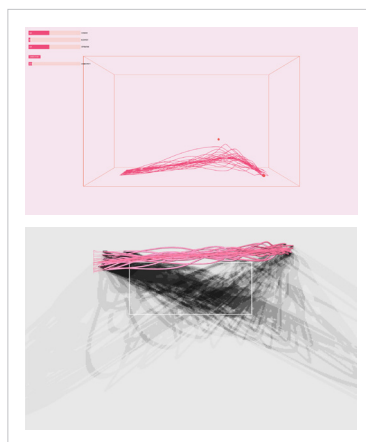
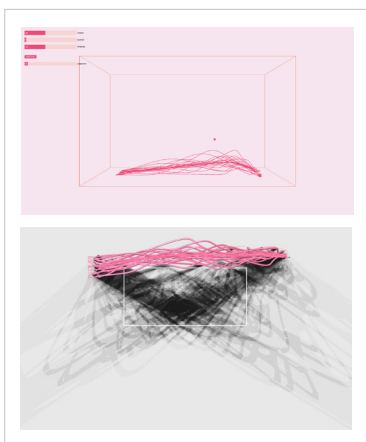
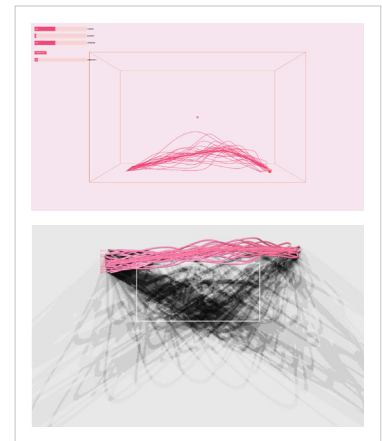
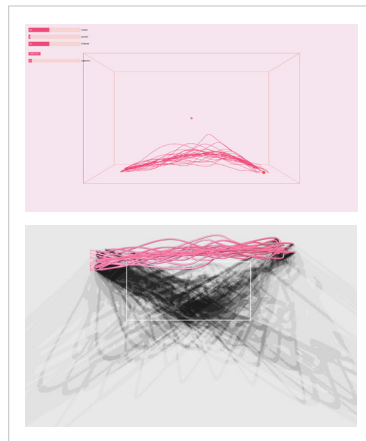
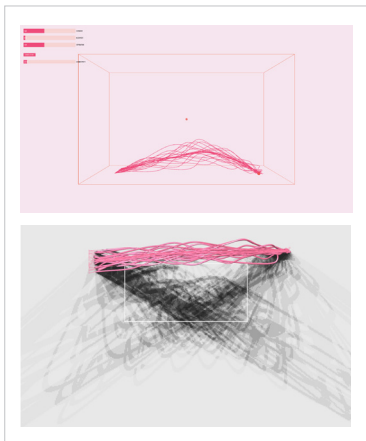
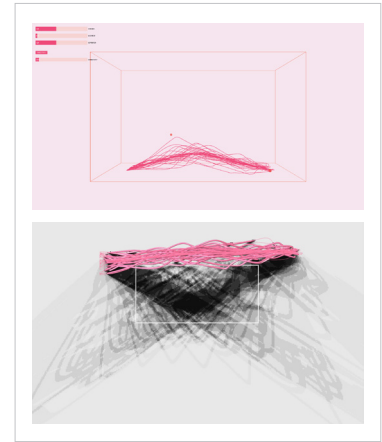
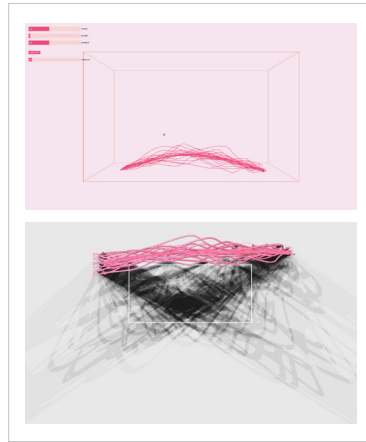
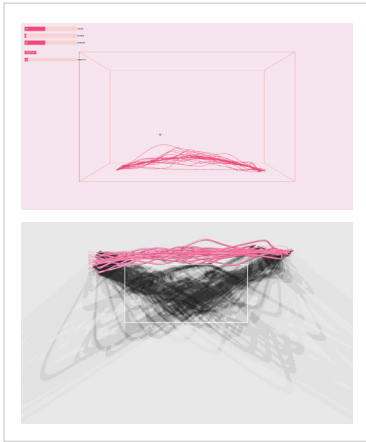
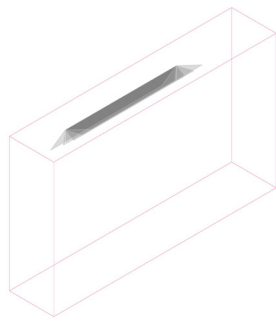
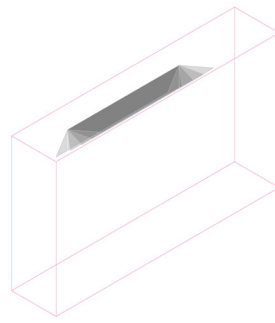


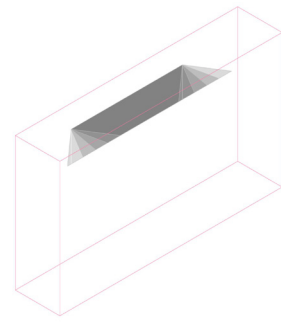
Figure 29. Sunshade analysis matrix



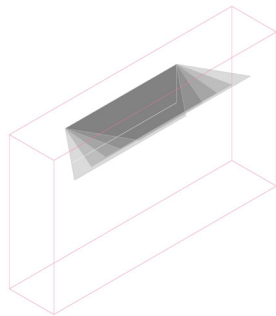
January



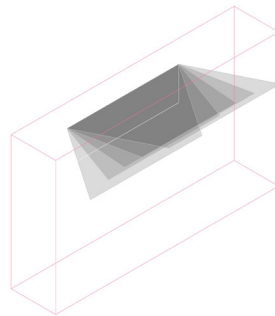
February



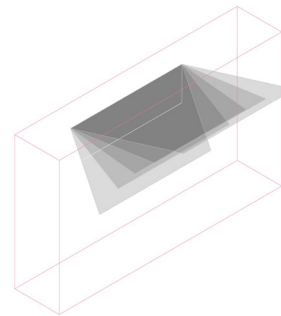
March



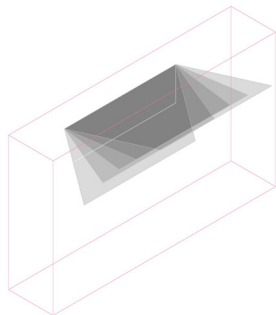
April



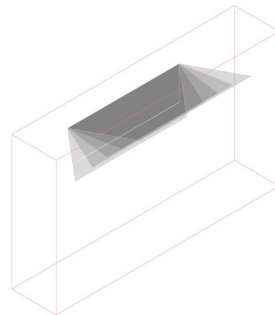
May



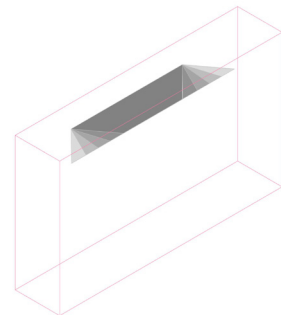
June



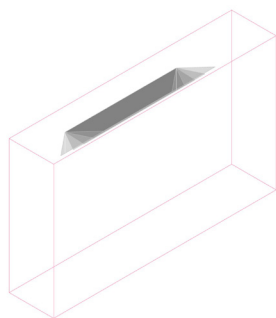
July



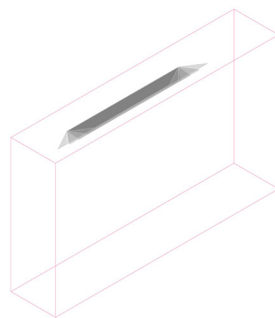
August



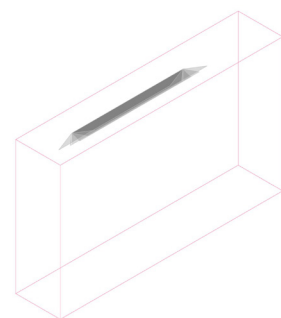
September



October



November



December

2.4

SUN SHADE DESIGNER

Ladybug also has the possibility to analyse a model such as the one built and determine the area where shading is necessary, this is also able to be used to determine the areas the agents should populate.

The polygons represent where shading should be at a specific time during the day, using this geometry and translating this into areas where the agents should fill. This desired shading area can be used for design intent, tapping into the form finding aspects of agent-based modeling.

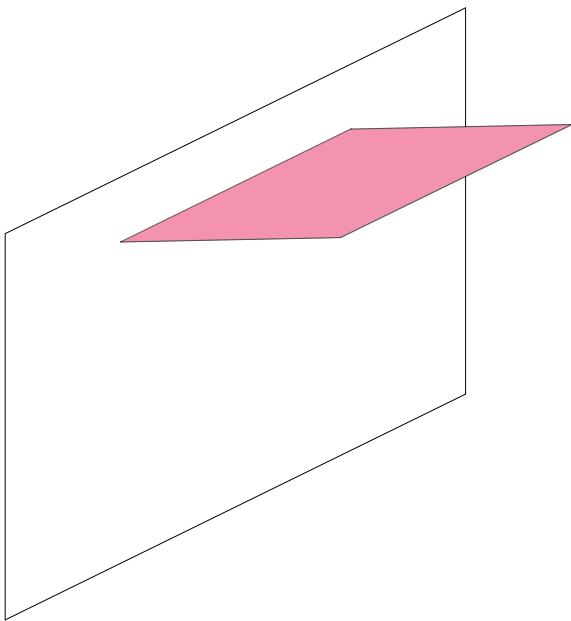


Figure 31. Sun shade for 10am

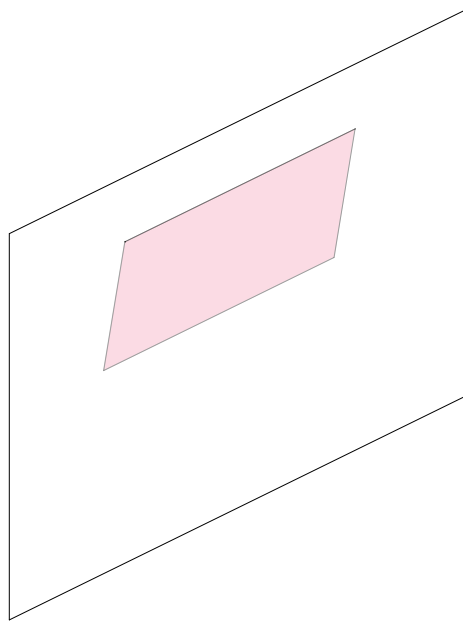
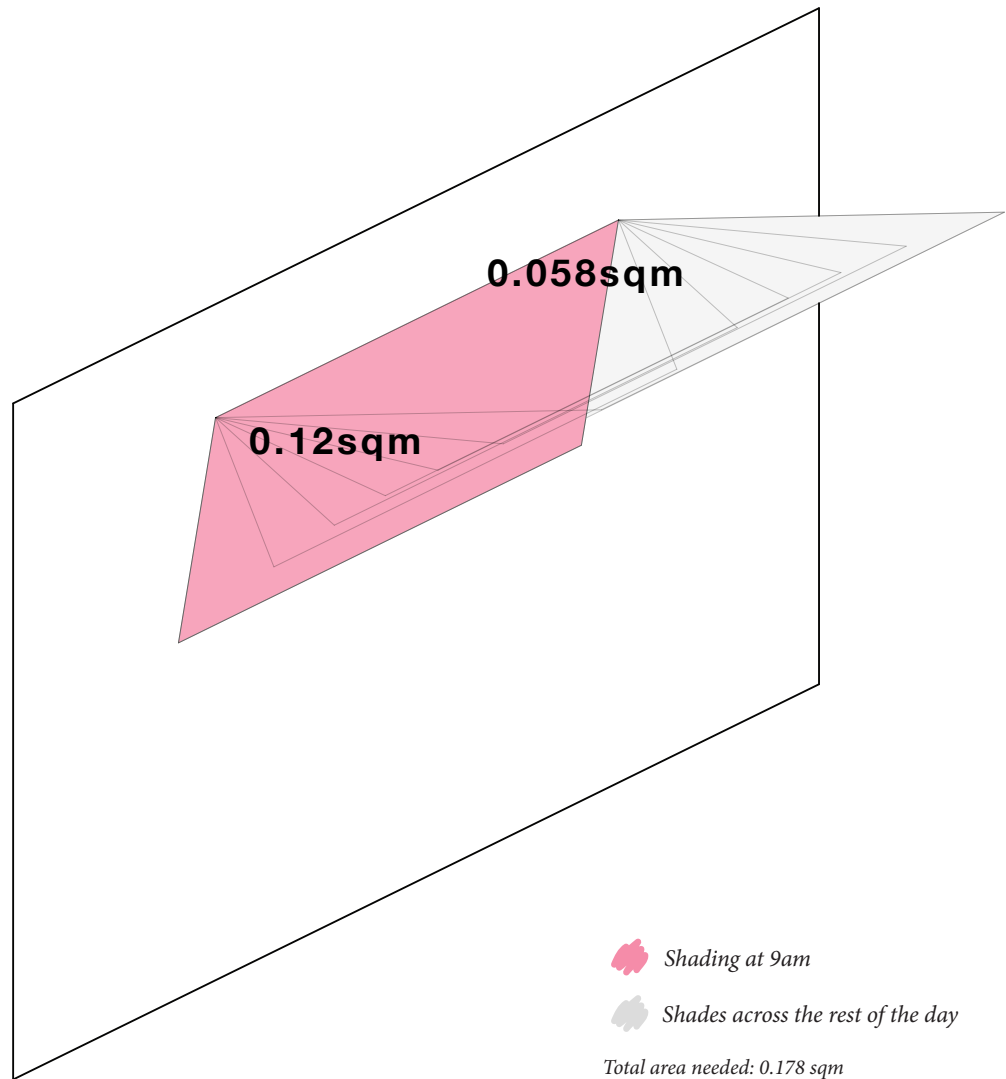


Figure 32. Sun shade for 4pm



Using kinetic shade structures that move with the sun levels on the window below,

Will save up to 32.6% material use

2.5

DYNAMIC SHADING

A key part of this section of the research is to develop a series of inputs that are able to be used to inform the agents and their movements. However it is also needed to develop a series of controls and models that are able to be passed on through the National Science Challenge to experiment with materiality and introduce the concept of 4D printing.

As a speculative design concept, 4D printing is a renovation of 3D printing, where based on 3D printing concepts, it uses designed materials and sophisticated designs that are “programmed” to prompt the print to change its form or state (Tibbits, 2015). Changes may be triggered through water, heat, wind or other forms of energy. For the purpose of this branch of the NSC, the states of sun throughout the day will be translated into heat and light data, where scientists will use this data to develop material properties.

Using sun shading as a proof of agent-based structure use, these experiments resonate with the concept of lower material use using free-form printing. While the plastics used for 3D printing have the ability to be recycled back into more filament to used, from a sustainable point of view it is ideal to minimise both material use and wastage.

4D printing lies outside the scope of this research, but analytics output from Grasshopper will be used to inform future designs within the NSC focussing on 4D printing.

2.6

AGENTS WITH DESIGN INTENT

With the inclusion of an ideal shading area for a window or internal area, it is essential to include parameters and variables that will encourage the agent simulations to populate certain areas. The population and steering of agents has been programmed using a series of 'attraction' points to drive the agents around the space.

Figure 35 demonstrates the unpredictability of the results using attraction points. Shown in the customising of Processing, the attraction points each have a strength value associated with each attractor.

Testing and experimentation of this process resulted in a series of preset parameters that are able to be added into a simulation to achieve a certain movement – the addition of a single or multiple attractors and increased strength vector to inform the simulation.

Stage 2: Introduction of steering controls and attractors

```
float target X = -100;  
float target Y = 150;  
float target Z = 0;  
float strength = 0.0;  
  
float target X2 = -300;  
float target Y2 = 400;  
float target Z3 = 0;  
float strength 2 = 0.9;  
  
pa.seek(target, strength2);
```

Figure 34. Processing Code used to include attraction parameters

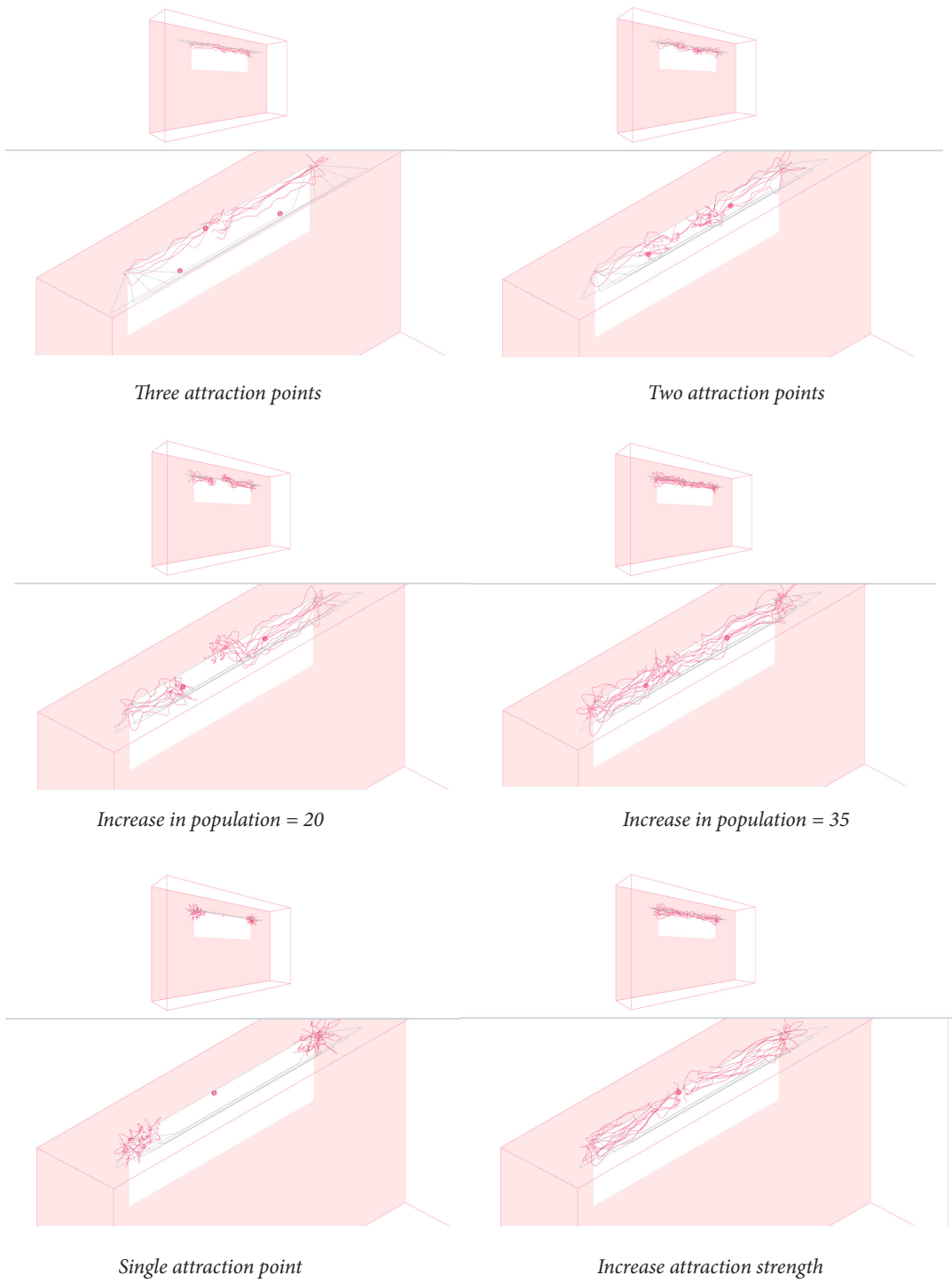


Figure 35. Processing simulations with variable parameters

2.7

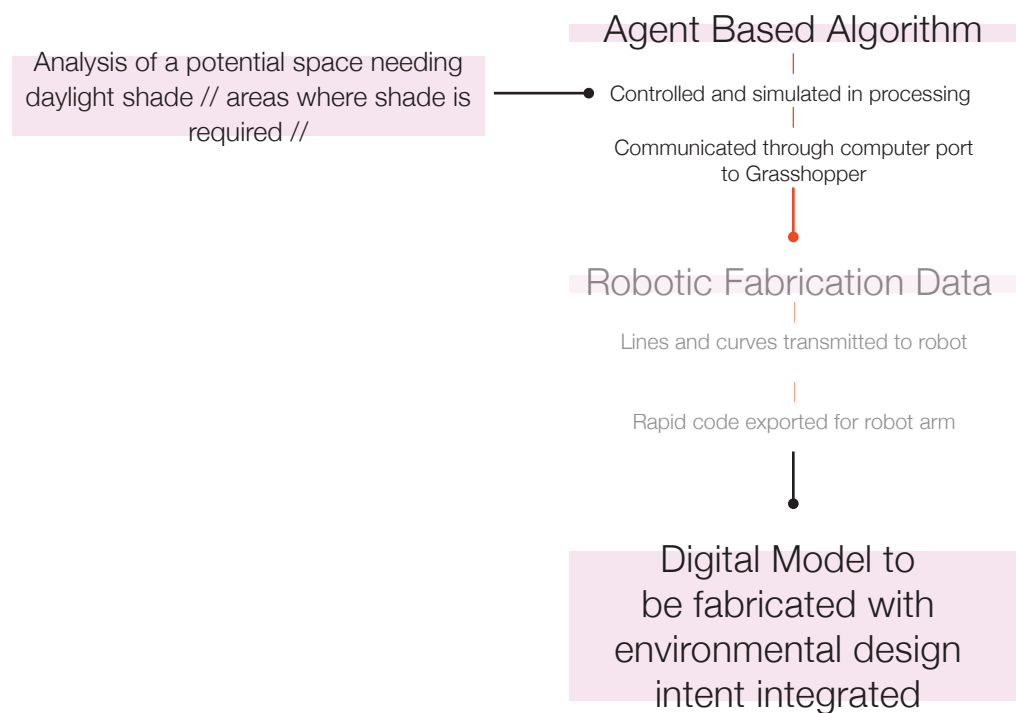
DIGITAL APPLICATION WORK-FLOW

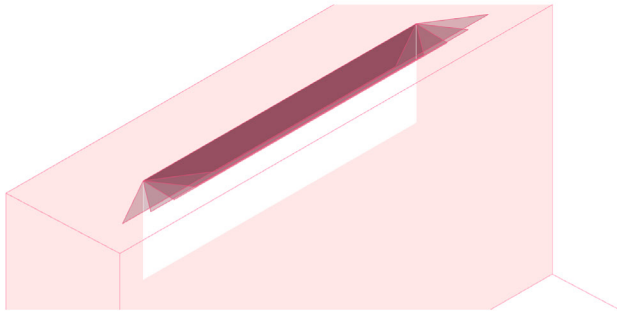
The focus of this chapter was to introduce the possibility of having a secondary design driver. Through the National Science Challenge this research will continue to produce digital models, the key output for this will be a digital model that is able to be printed if needed. In the form of a sunshade, this design has thus far taken on environmentally informed design geometry, and then using analysis tools, evaluated the effectiveness of each simulation.

Shown in Figure 36, this digital work-flow draws on multiple data sources in an attempt to encode environmental data within the agent design.

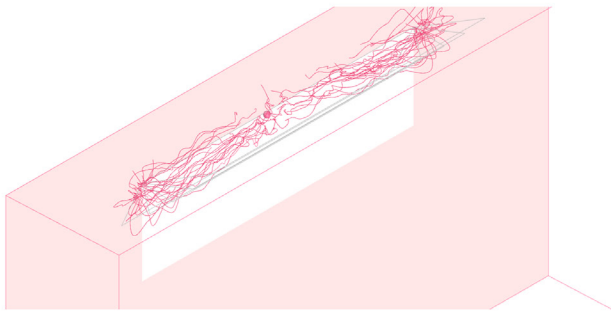
The next phase of the digital application will focus on the fabrication of these systems, and how the sun data is able to impact the model when it is translated into a printing system.

It is important to note that the environmental design integrated into this section of research is site specific climatic data (sun hours and location). It is not true environmental analysis, but a design tool that gives calculations and analysis of a specific area. The nature of the design and the variable orientations of a site are factors that can impact results.

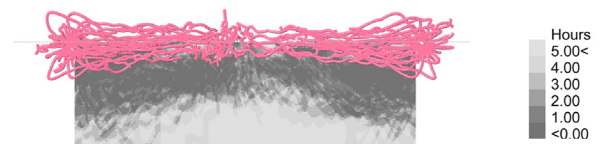
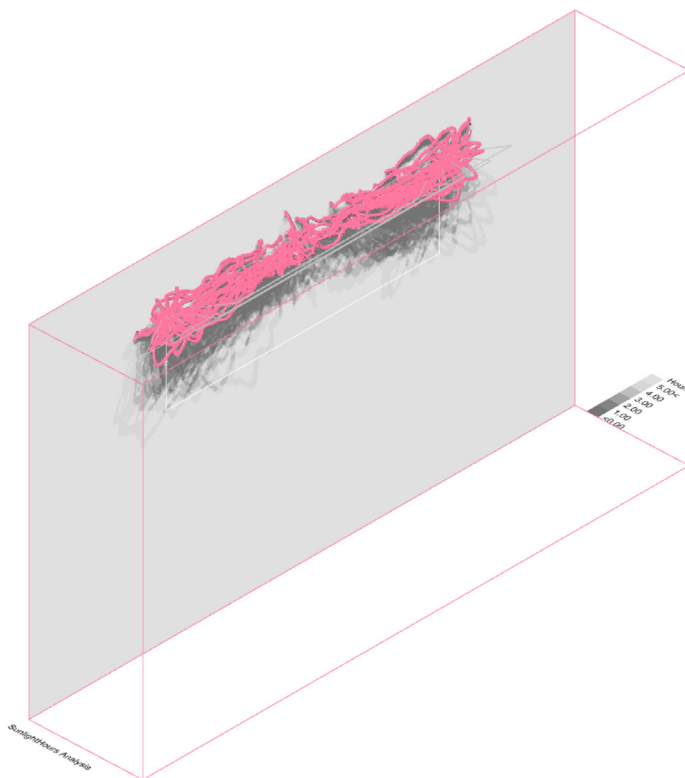




Sun shading areas 10am - 2pm



Scripted agents reacting to optimal shading area



Shading analysis on window

CHAPTER 3:

FABRICATION RESEARCH

/ Biological Algorithms for Digital Manufacture

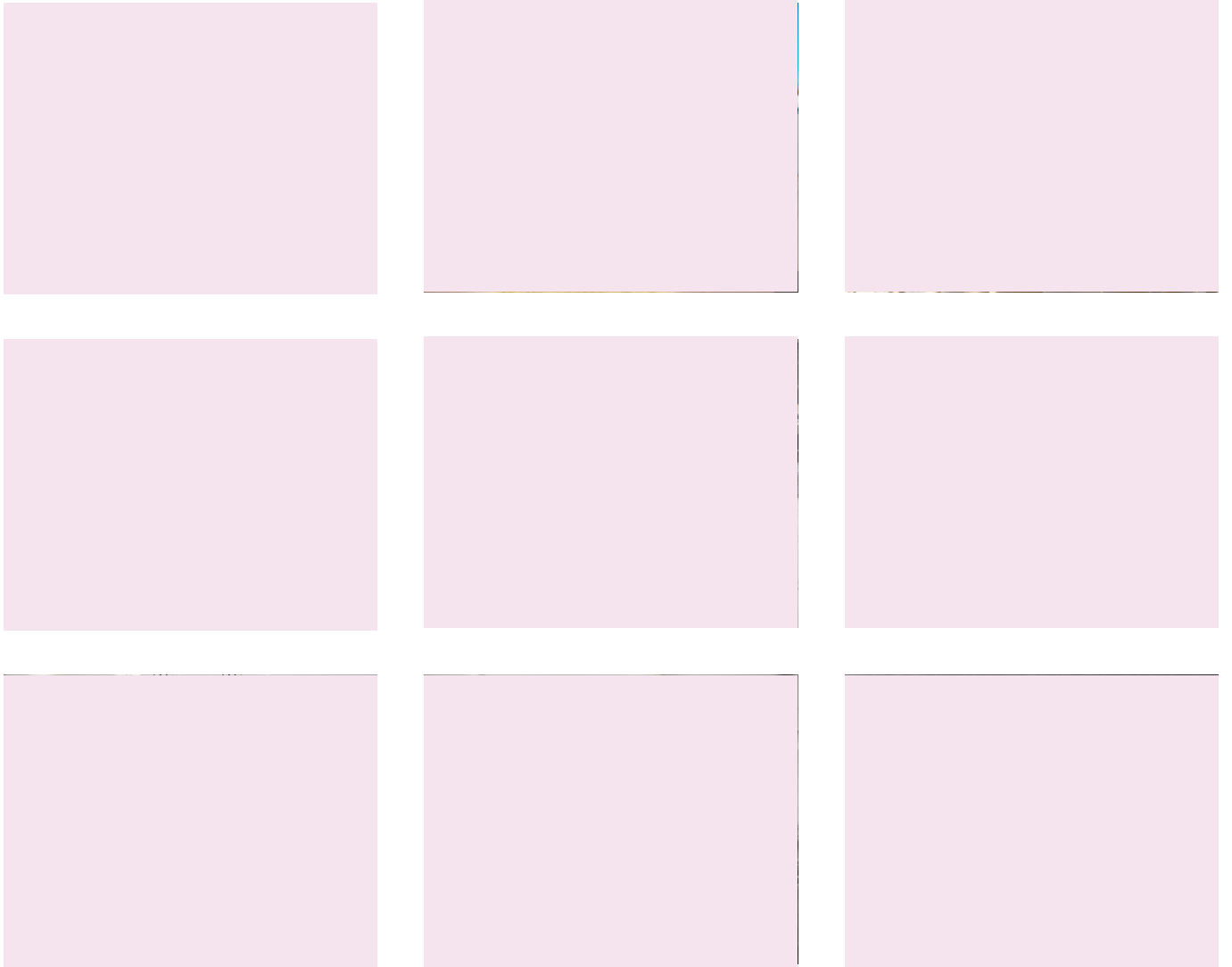


Figure 37. Examples of robotic fabrication in both permanent and semi-permanent architecture.

3.01

INTRODUCTION : ROBOTICS IN ARCHITECTURE

As highlighted in the introduction, the inclusion of robots in the realm of architectural research has introduced many possibilities within fabrication. The versatility of robotic fabrication and 3D printing is able to work in parallel with the agent-based simulations designed, and able to be fabricated. The nature of robotic 3D printing is very experimental, it is currently predominantly explored at research universities around the world and used to fabricate semi-permanent architectural installations.

Historically, 3D printing has been an incremental process, building up layers and layers to form a single object. This method has been researched and developed as having possibilities to be used as large scale projects, entire houses for people to occupy.

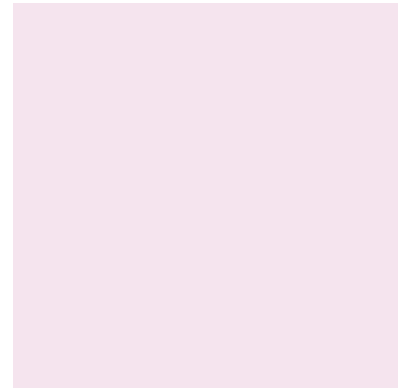
Free-form printing is the concept of printing structures in the air, the ability to create space frame-like structures without the need for excess material use.

The printer that is able to be used for this research is a custom modified extruder which is held at the end of the robot with a set of grippers. This extruder works independently from the robot, where it is turned on before a build starts, printing PRO HT 3mm filament, this filament is much larger than standard 1.5mm filament, allowing for larger, thicker prints than common 3D printers.

3.02

PRINTING AND STRUCTURAL APPROACHES

The following projects are both academic and industry examples of free-form 3D printing using robotic arms. Moving forward and beginning to fabricate the agent-based structures, it is key to consider how the structure will be printed and formatted as a digital model. These three examples each have different printing techniques and extrusion devices, each which are equally important in building a sophisticated model for fabrication.

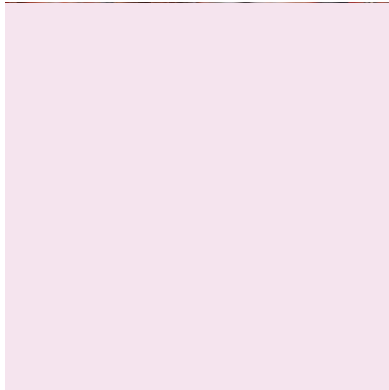


DAEDALUS PAVILION

Ai Build's 'Daedalus Pavilion' is a 3D printed architectural installation. This was part of NVIDIA's GPU Technology Conference in September 2016. In partnership with ARUP Engineers this project was aimed to showcase how the future of construction will be transformed by robotics and artificial intelligence.

MATERIAL	"Biodegradable filament by Formfutura", a variety of PLA and ABS materials (160kg)
SIZE/TIME	5m x 5m x 4.5m
DEVICE	Kuka Industrial Robot
EXTRUSION DEVICE	Custom Build Extruder
PRINTING TECHNIQUES	Linear movements
PRINTING STRUCTURE	Divided Grid of superstructure
SCALE	Scale - Large Scale 3D printing
THESIS RELATION	Approach of layering techniques in grid like structure. Consistency across the structure by using grid.

Figure 38. (AI Build, 2016)



CURVVOXELS

Bartlett's 'CurVoxels' project explores the method of 'voxelization' from a traditional masonry influence in a new perspective and applying it to digital fabrication. Using a series of 3D printers on the end of an industrial robot, a 'design-fabrication integration system' was formed. Resulting in a series of digital models at various scales and printing medium scale models as proof of concept.

2.85mm ABS filament

50cm x 50cm x 85cm (14 hours)

ABB IRB 1600

A series of custom built extruders

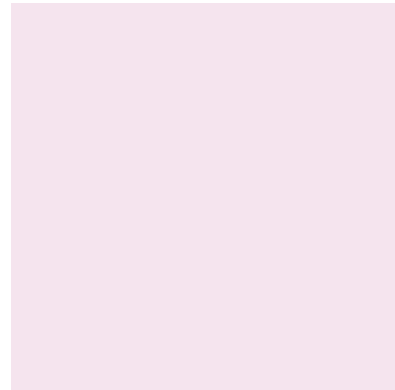
Linear movements with ability to stop/start printing

Voxel-based structure

Small scale experiments moving to prints the size of a chair (medium scale)

This project uses voxels as a method of space division - a system which allows for infinite scalability and transfer of data

Figure 39. (Kwon et al., 2015)



SPACEWIRES – FILAMENTRICS

'Spacewires - Filamentrics' is a Bartlett project focusing on digital design and fabrication. Within the project there is a section on robotic 3D printing. The focus is on pushing the fabrication method beyond space frame structures and beginning to build a non-linear framing structure. This was then digitally scaled, a series of medium scale prints were completed as proof of concept.

3mm ABS filament

70cm x 35cm x 30cm / 40 hours

Small robotic arm

A series of custom built extruders

Space frame with explorative 3D approach - towards a non-linear framing structure

Linear grid pushed to be organic

Small scale experiments which have been digitally scaled using fabrication theory

The research method of testing fabrication at small to medium scale and then up sizing it digitally to prove research at large scale.

Figure 40. (Jiang et al., 2015)

```

{0;0;0}
0 WaitTime \InPOS, 0.9;
1 MoveL tPath0,s2,z1,LouiseExtruder1\WObj:=WObj0;
2 WaitTime \InPOS, 0.9;
3 MoveL tPath1,s1,z1,LouiseExtruder1\WObj:=WObj0;
4 WaitTime \InPOS, 0.9;
5 MoveL tPath2,s1,z1,LouiseExtruder1\WObj:=WObj0;
6 WaitTime \InPOS, 0.9;
7 MoveL tPath3,s1,z1,LouiseExtruder1\WObj:=WObj0;
8 WaitTime \InPOS, 0.9;
9 MoveL tPath4,s1,z1,LouiseExtruder1\WObj:=WObj0;
10 WaitTime \InPOS, 0.9;
11 MoveL tPath5,s1,z1,LouiseExtruder1\WObj:=WObj0;
12 WaitTime \InPOS, 0.9;
13 MoveL tPath6,s1,z1,LouiseExtruder1\WObj:=WObj0;
14 WaitTime \InPOS, 0.9;
15 MoveL tPath7,s1,z1,LouiseExtruder1\WObj:=WObj0;
16 WaitTime \InPOS, 0.9;
17 MoveL tPath8,s1,z1,LouiseExtruder1\WObj:=WObj0;
18 WaitTime \InPOS, 0.9;
19 MoveL tPath9,s1,z1,LouiseExtruder1\WObj:=WObj0;
20 WaitTime \InPOS, 0.9;
21 MoveL tPath10,s1,z1,LouiseExtruder1\WObj:=WObj0;
22 WaitTime \InPOS, 0.9;
23 MoveL tPath11,s1,z1,LouiseExtruder1\WObj:=WObj0;
24 WaitTime \InPOS, 0.9;
25 MoveL tPath12,s1,z1,LouiseExtruder1\WObj:=WObj0;
26 WaitTime \InPOS, 0.9;
27 MoveL tPath13,s1,z1,LouiseExtruder1\WObj:=WObj0;
28 WaitTime \InPOS, 0.9;
29 MoveL tPath14,s1,z1,LouiseExtruder1\WObj:=WObj0;
30 WaitTime \InPOS, 0.9;
31 MoveL tPath15,s1,z1,LouiseExtruder1\WObj:=WObj0;
32 WaitTime \InPOS, 0.9;
33 MoveL tPath16,s1,z1,LouiseExtruder1\WObj:=WObj0;
34 WaitTime \InPOS, 0.9;

```

Figure 41. Rapid Code Output from HAL Robotics

3.03

RAPID CODE

The robot is controlled using Rapid Code, a high-level programming language used to control ABB industrial robots. This language includes routine parameters, procedures, logical expressions, automatic error handling and functions of the robot. Each are essential in the movement of the robot. The 3D printer that will be used for this research works independently from the robot, and is controlled using G-Code. Due to there being no connection between the programming of the extruder and the robot, the Rapid Code only needs to be written to control the movement and orientation of the robot and is not associated with the extruder.

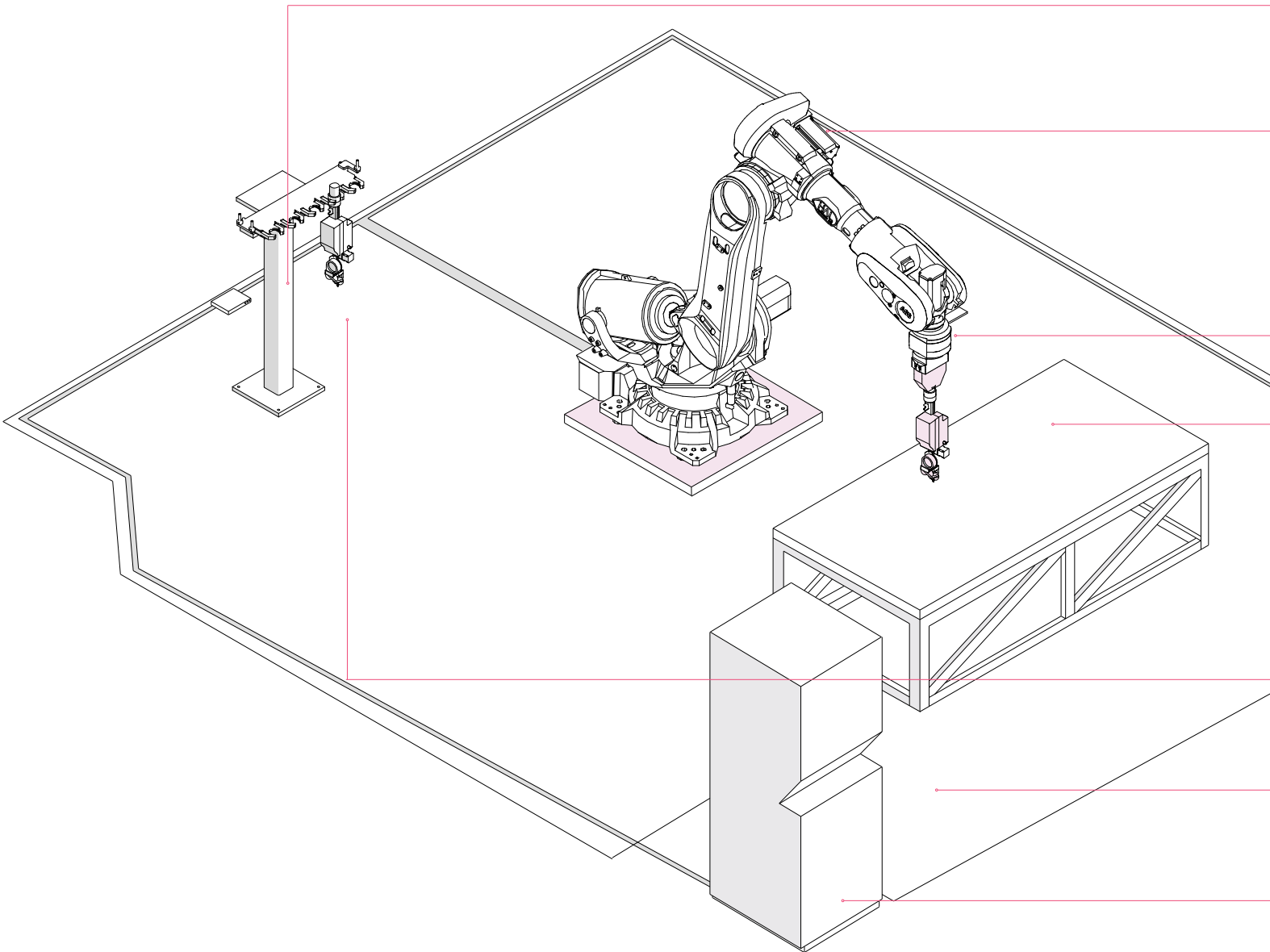
3.04

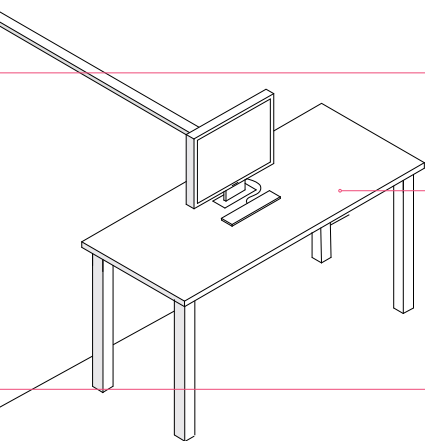
PLANES

Rapid Code uses the location of planes in space, working off a Cartesian XYZ system. The end of the robot will orientate itself to a point programmed and move between each of these until the end of the routine. The joint orientation and speed is also able to be controlled, these are all key in ensuring a 3D print is completed successfully. The Rapid Code is generated using a plug-in for Grasshopper, HAL Robotics, this is a plug-in which is simple to use and reads digital plane coordinates and translates this into code that the robot reads.

The digital geometry needs to be converted from a curve-based geometry into planes, a Grasshopper component is used to either divide the curve into points, with a parameter defining the distance between points, or a component which is able to take the points where the robot arm needs to change direction. Multiple components are also known as a Grasshopper cluster, a series of actions aggregated to act as one function computationally. This generally is developed within the printing script, as the form is generated as a series of straight lines which connect end to end to build a print.

3.05 ROBOT AND WORKSPACE





POWER SUPPLY & CABLING

24V Power supply for free-form printer

ROBOTIC ARM

ABB IRB6700. Six axis robotic arm.

FILAMENT

A roll of filament is located near the robotic arm and fed through a series of loops to hold it in place.

PNEUMATIC GRIPPER

A physical attachment to the robot used to pick up various objects, including the free-form printing tool

TABLE

Holds the work surface for the 3D printing

CONTROL COMPUTER

Used for programming the robot, and controlling the free-form printer. Through Firefly, a plug-in for Grasshopper, live communication is made to the free-form printer. The Arduino micro-controllers transfer this data and allow the switching on of filament heating, printer fan and extrusion.

TOOL BUFFER

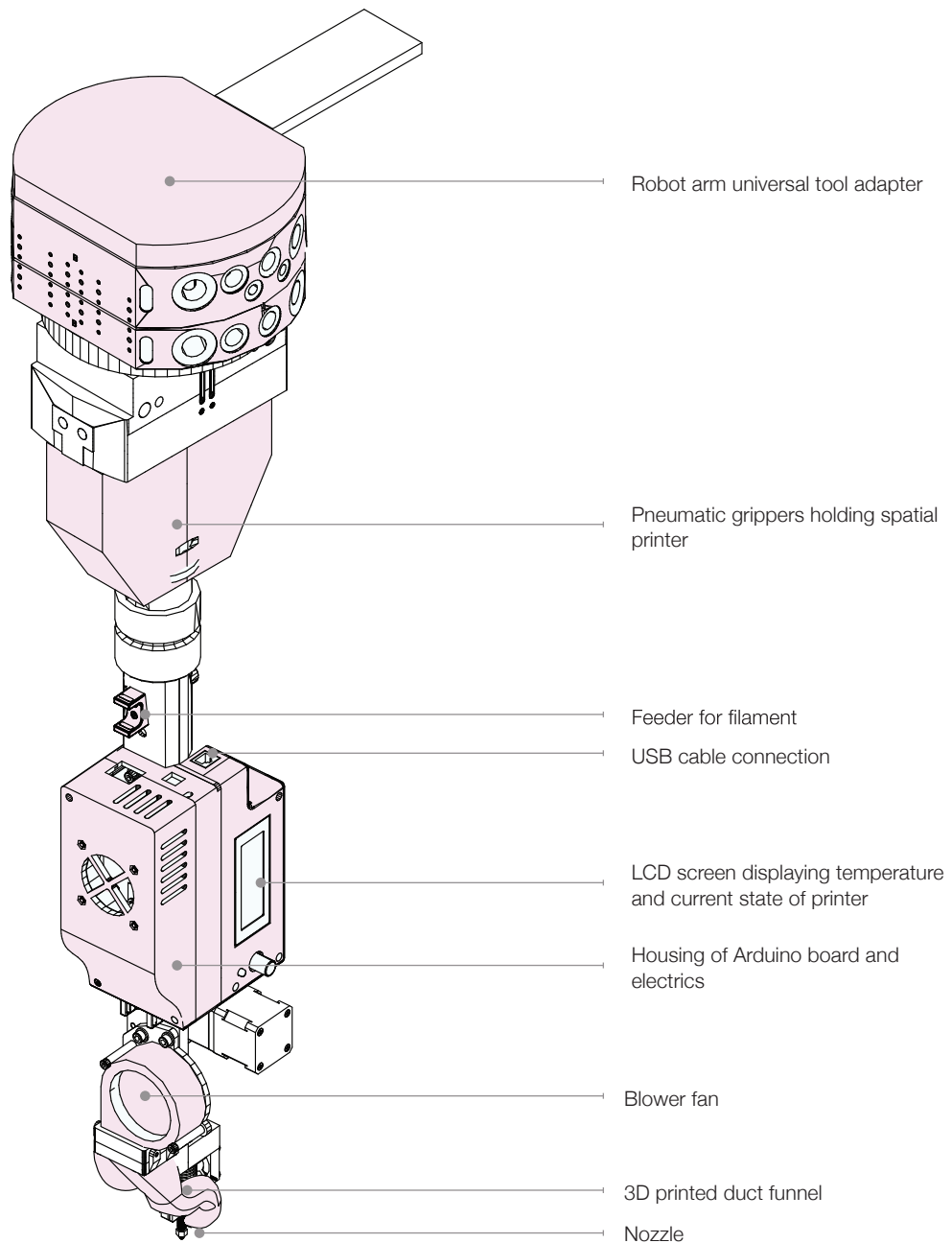
Holds a range of attachments that the robot is able to pick up – the free-form printer is held here when not in use.

FLOOR

The floor in the robot workspace is uneven. This affects the accuracy of the robot and anything sitting upon it.

CONTROL CABINET

Responsible for the motion and input output signals of the robot. The control cabinet has a touch pendant where the programs are uploaded to and the robot is able to be jogged and controlled with analogue and digital controls.



3.06

VUW EXTRUDER

The 3D printer that is mounted on the robot is a custom built extruder of the university's. This extruder was built using an Arduino board to control the electronic componentry, this receives the G-Code commands that control the printer. A stepper motor is used to extrude the filament through a hotend, the temperature of this is controlled through the G-Code which is able to be regulated and input each time. At the end of the extruder by the nozzle is a blower fan, the blower fan is funneled through a 3D printed duct towards the end of the nozzle, assisting the plastic in cooling and solidifying during the printing process.

There is no connection between the G-Code and the robot system, so the feed-rates will have to be set at a constant speed and the extruder is left to run continuously. An integration of the G-Code and the Rapid Code used to control the robot would result in better communication and print optimisation. A variable print speed and the ability to have the extrusion turn on and off would allow for customisation, but using this extruder the prints will be restricted to a continuous line model.

3.07

RESEARCH AT VUW

Free-form 3D Printing: A Sustainable, Efficient Construction Alternative, is a research thesis by Armano Papageorge, investigating 3D free-form printing using computational tools and an industrial robotic arm. The primary branch of this thesis was the viability of the printing as a construction system, however the rigorous testing and exploration with the same 3D Print extruder is extremely valuable as base knowledge for how these non-linear forms will be fabricated in a similar manner. Optimal temperatures and extrusion speeds gave a starting point for the fabrication testing of this research to stem from. (Papageorge, 2018)

MATERIALITY OF 3D PRINTING

The research focus is on a digital work-flow, undertaking experiments in simulating emergent processes through to digital fabrication, however through research, the system is able to consider the materiality of fabrication.

Material filaments come in a range of types, with three being the most regularly used. These are:

- ABS (Acrylonitrile butadiene styrene) This is a cost effective filament and flexible. It is petroleum based and not biodegradable. Generally seen as not being eco-friendly.
- PLA (Polylactic Acid) is made from renewable resources. It is a biodegradable material, derived from corn-starch or sugarcane.
- PVA (Polyvinyl alcohol plastic) is a filament that is used for 3D printing to create a support structure during the printing process.

Also available for use is a type of PLA called ProHT, manufactured by BigRep. This filament is non-toxic product which has been used in the past for free-form printing and achieved optimal adhesion and warping during the printing process. ProHT is also high-heat and weather resistance, with extremely high strength properties (BigRep, 2013).

Through experimentation and research it is expected a large amount of filament will be used, all of the material used will be collected for recycling. BigRep's PRO HT filament is unable to be recycled itself, but the material is able to be melted down and extruded back out into filament for future 3D prints.



Figure 45. PRO HT Filament

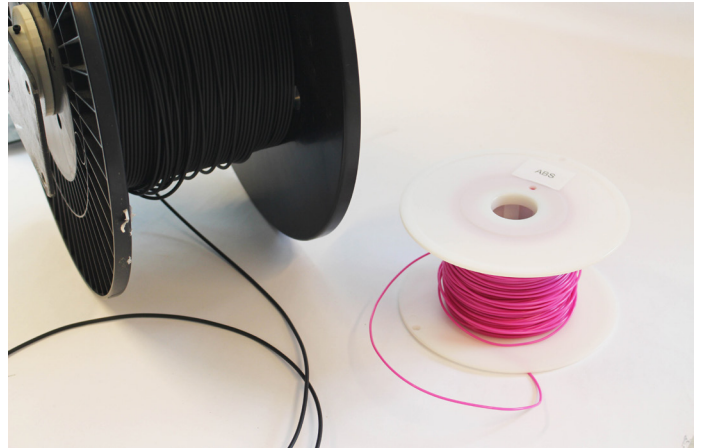


Figure 44. ABS Filament in Pink

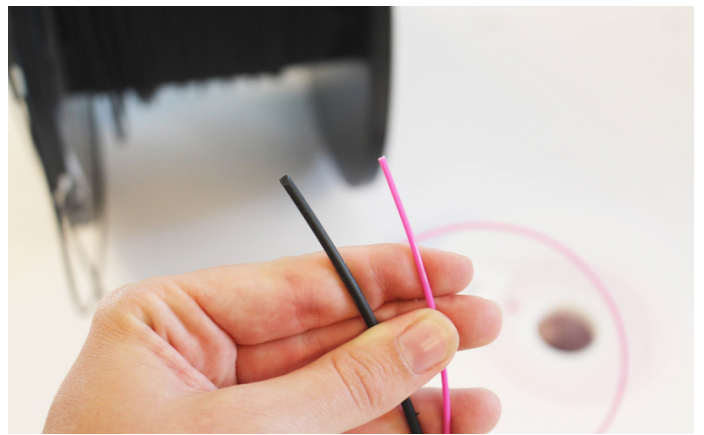


Figure 46. PRO HT vs ABS Filament

3.09

DEVELOPMENT OF SCRIPT

The code for the robot is built through a Grasshopper plug-in called HAL Robotics, capable of building and editing RAPID Code.

Through a series of edits, the code is built to accommodate and overcome each of the limitations of the printer. Variable speeds and the wait time segments were added over time when issues arose during the test print phase.

Fed into this script is a series of planes built using Grasshopper. The orientation has already been edited and corrected prior to passing the data to HAL. For initial tests when the orientation wasn't considered, HAL Robotics translates any point data from Grasshopper to an XY plane for the robot to use.

The RAPID Code generated is export to a USB drive or sent to RobotStudio, the connection software for the robot. Formatted into its own folder and read by the robot.

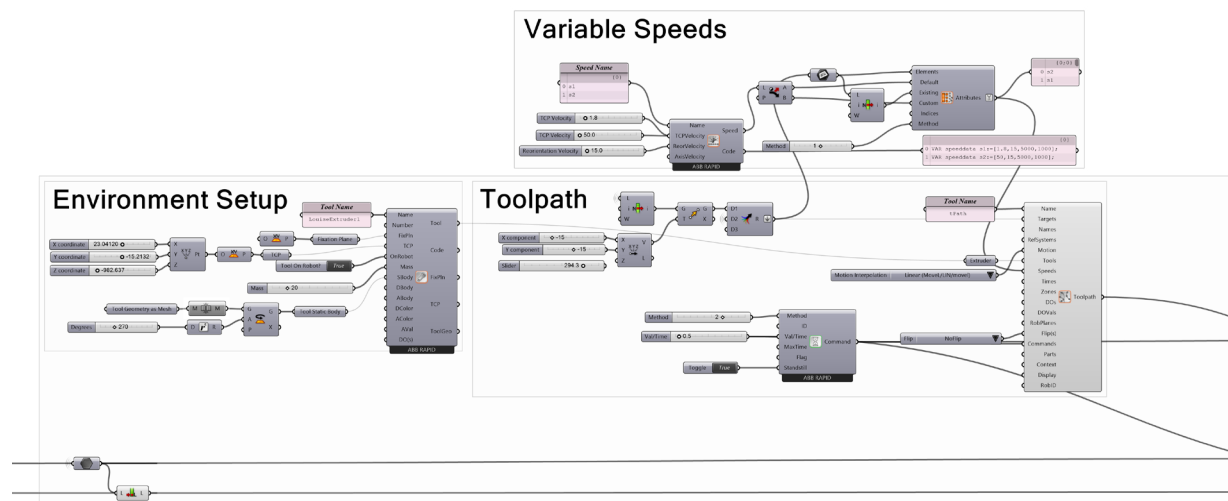


Figure 47. HAL Robotics section of Grasshopper Script

ENVIRONMENT SET UP

Section of script responsible for robot data, tool using, location of robot and area to move within.

VARIABLE SPEEDS

Adds variable speeds to the script. The robot will then move at a faster speed to the start point of the routine, then move at the designated speed (1.2 - 1.8mm/s).

TOOLPATH

Input of tool path for the robot to move through. Initial start point added in for the purpose of giving the robot a point to move to at a higher speed before beginning routine.

CODE GENERATION

HAL Robotics node that takes plane data and translates it into points in space of RAPID Code for the robot to interpret.

PRINT TIME

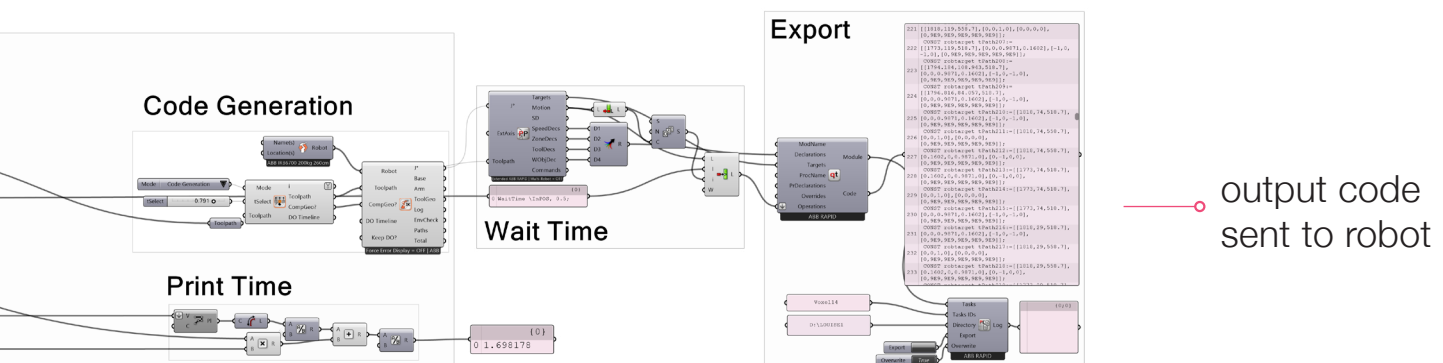
Grasshopper cluster used to determine the total print time.

WAIT TIME

RAPID Code commands for the robot to pause at points are inserted between each line of code, or certain areas of the code. This allows the print to stabilise and cool before moving on.

EXPORT

RAPID Code is exported using this HAL component to a designated folder.



3.10

PRINTER SET UP

The accuracy of the robot in a number of the test prints and when being used in the past of by other students hasn't been to a high level. The accuracy is essential to ensure that the connections and layers of the 3D prints are formed correctly. The tolerance in the connections and the distance between sections of the print require a high level of accuracy and calibration in the extruder.

tcp: tool centre point // the end of the attached end effector on the robot

The tool centre point (TCP) is a point in space relative to the end of the tool. On the extruder this is the very end tip of the nozzle. When setting the centre point the robot is manually jogged into position that the end is located on a point (Figure 48) from multiple angles. The location for these points are given and then inserted into the digital model and a new location of the TCP is programmed into the RAPID Code.

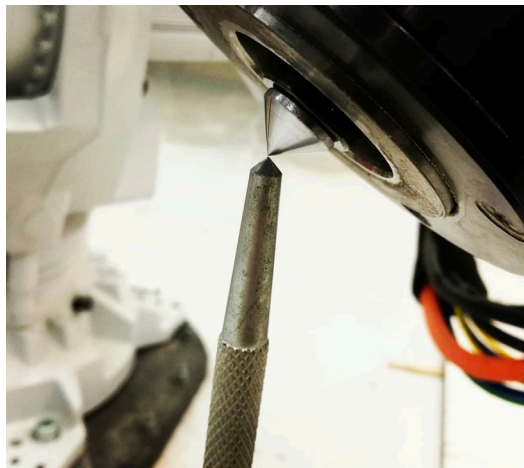


Figure 48. Extruder being TCP'd before use

There were certain points where it was possible that the extruder was not calibrated correctly, and when re-orientating the nozzle would come off the print. At multiple points throughout the research the extruder was re-calibrated and this did not entirely fix the problem. Each time the extruder was re-calibrated to a precision level of within 0.1mm accuracy, it is likely that there was instead issues with the robot or the extruder moving within the pneumatic grippers. To prevent this each use the grippers were checked for pressure and the print surface was leveled.

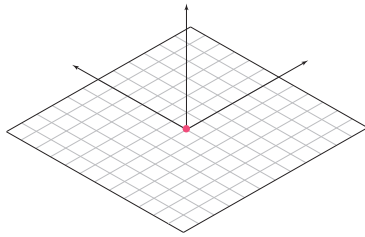


Figure 51. Digital Plane Orientation

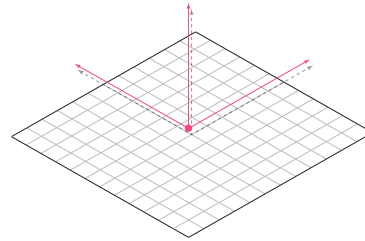


Figure 52. Incorrect plane orientation occurs when TCP incorrect

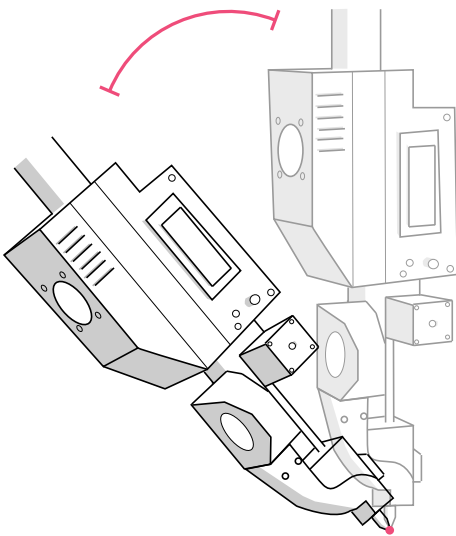


Figure 49. Correct TCP of printer

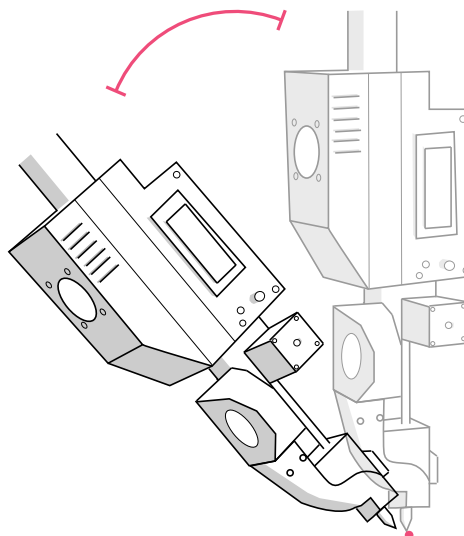


Figure 50. Slightly incorrect TCP of printer causes accuracy to decrease

3.11

SERIES 1

The research into the capabilities and technical details of the extruder, created a platform to begin testing the printer with. An initial series of linear frames were printed to test the programming and printing at a small scale. Attention to the programming of the script at this stage built a clear digital model that was able to adapt to different shapes and printing attributes.

During the printing of Series 1, a number of print errors occurred due to print surface being incorrect in the digital model. If the print surface is not set correctly the print won't begin correctly.

- too high:** if the print surface is too close to the extruder when the print starts -the plastic will be squished to the surface, resulting in an different thickness at the base level. See print test 1.03. This also affects material printed on top as the material will be lower than expected.
- too low:** if the extruder is too far above the print surface the plastic is unable to adhere to the acrylic and the print will not form. If this is a negligible amount it is likely a portion of the print will form until the extruder pulls the plastic off the surface due to not enough print surface cooled and set on the surface. (See print test 1.04)

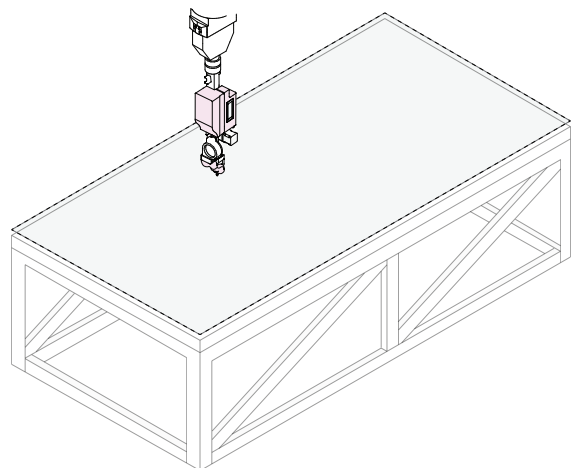


Figure 53. Print surface digitally modelled higher than physical model

1.01



print number	1.01
intention	print 2 x 2 grid
outcome	fail
reflection	print speed too fast - resulting in the PLA not forming solid and collapsing.
print speed	2mm/s
wait times	-
offset	-

1.02



print number	1.02
intention	print 2 x 2 grid
outcome	fail
reflection	print speed decreased - better connections at bottom pins, however upper struts failing.
print speed	1.5mm/s
wait times	0.8s
offset	-

1.03



print number	1.03
intention	print 2 x 2 grid
outcome	fail
reflection	printer calibrated too close to print surface - resulting in wide first layer stuck to base of print.
print speed	1.5mm/s
wait times	1.0s
offset	-

1.04



print number	1.04
intention	print 2 x 2 grid
outcome	fail
reflection	printer calibrated too far from print surface - the print was able to unattach itself from the surface and move mid-print.
print speed	1.5mm/s
wait times	1.0s
offset	-

1.05



print number	1.05
intention	print 2 x 2 grid
outcome	fail
reflection	print speed too fast - resulting in the PLA not forming solid and collapsing.
print speed	1.7mm/s
wait times	1.0s
offset	-

1.06



print number	1.06
intention	print 2 x 2 grid
outcome	printed
reflection	with a lower print speed and proper calibration of the TCP, the bottom layer printed correctly, failure came when building a second layer.
print speed	1.5mm/s
wait times	1.0s
offset	-

1.07



print number	1.07
intention	print 2 x 2 grid
outcome	fail
reflection	printer calibrated too close to print surface - resulting in wide first layer stuck to base of print, then unbalanced torsion pulled the print off the base.
print speed	1.5mm/s
wait times	1.0s
offset	2mm

1.08



print number	1.08
intention	print 2 x 2 grid
outcome	printed
reflection	with a slower wait time speed - the print was able to build strong connections at each of the junction points.
print speed	1.5mm/s
wait times	1.2s
offset	2mm

1.09



print number	1.09
intention	print 2 x 2 grid
outcome	fail
reflection	printer interrupted itself - the print was orientated in a different position on the work surface - causing the printer to hit into previous plastic.
print speed	1.5mm/s
wait times	1.2s
offset	2mm

1.10



print number	1.10
intention	print 2 x 2 grid
outcome	fail
reflection	printer calibrated too far from print surface - the print was able to unattach itself from the surface and move mid-print.
print speed	1.5mm/s
wait times	1.2s
offset	2mm

1.11



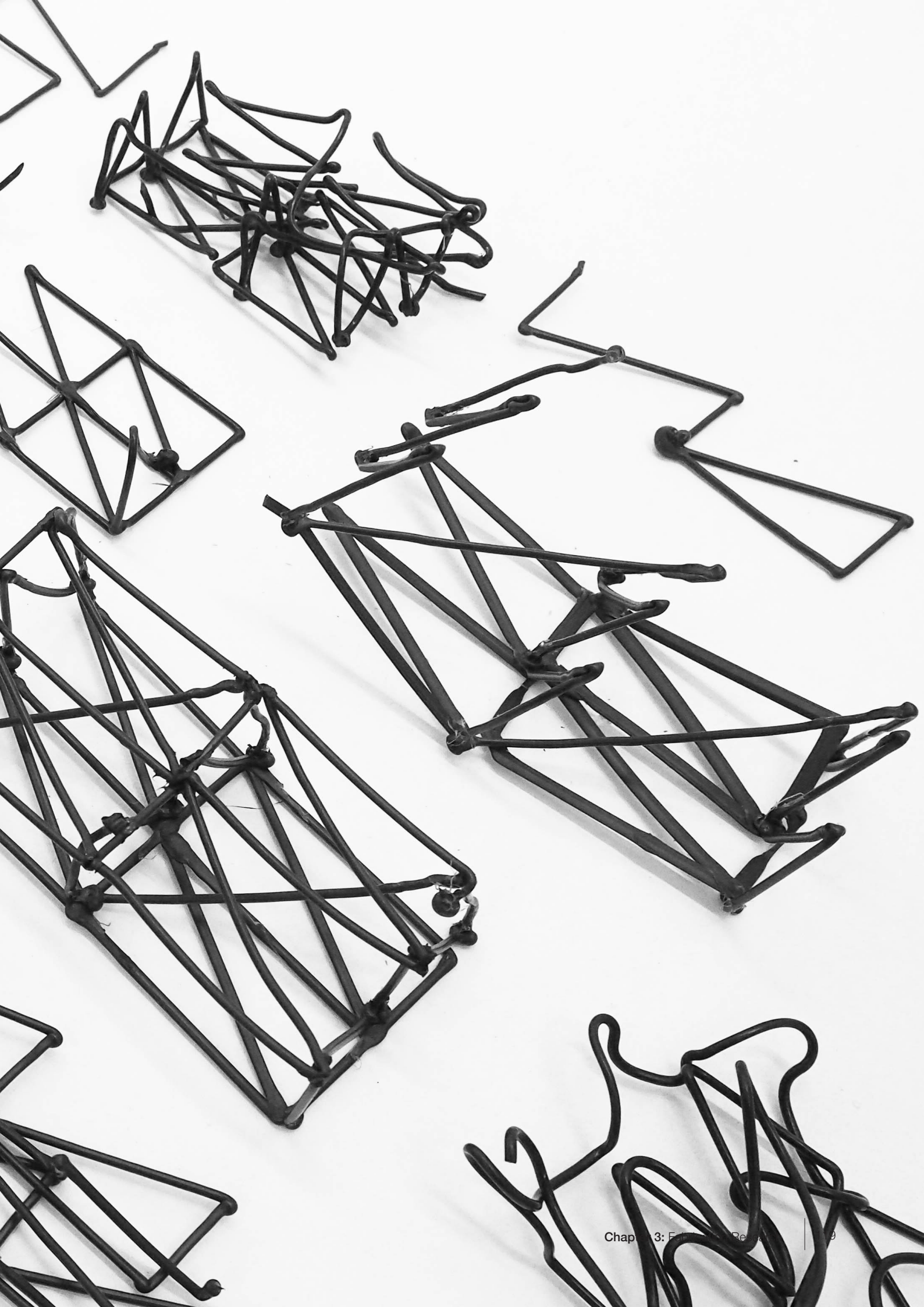
print number	1.11
intention	print 2 x 2 grid
outcome	printed
reflection	print was able to form correctly, however the low height difference between each layer meant the second layer sunk into the first.
print speed	1.5mm/s
wait times	1.2s
offset	2mm

1.12



print number	1.12
intention	print 2 x 2 grid
outcome	fail
reflection	print formed correctly, over time the robotic arm pulled the printer off the surface, causing the second layer to be printed slightly off centre.
print speed	1.5mm/s
wait times	1.2s
offset	4mm





3.12

SERIES 2

This series focussed on voronoi patterns. Voronoi, a mathematical peculiarity with the division of space into contiguous cells, also referred to as

“the golden mean of computational architecture”

(Bazalo, 2014).

Each voronoi cell is partitioned based on the distance between surrounding points. Seen as an optimal division of space. Voronoi was chosen for this section of test prints with the intention it would produce a series of structurally sound forms if fabricated efficiently. This would be done steering towards an un-orthogonal grid and focussing on how the print order and robot orientation is programmed. The nature of the extruder calls for each print to be fabricated in one continuous line unable to stop and start at different areas of the geometry. This results in a digital model without crossing paths, printed top to bottom.

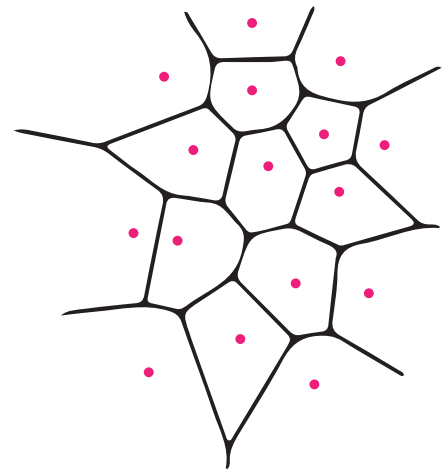


Figure 54. Voronoi structure

It was during this section of the research that issue arose and the true limitations of the printer showed, highlighting how it inhibited the printing process. These are described in the following print evaluations and section 3.14: Print Errors

During the organisation of the printing and the work flow from a digital model to printed object, this phase allowed for refinement of ideal printing parameters. Optimal temperature, extrude rate and speed for the printer to move at were each highly important to build a successful print.

Figure 55-Figure 57 demonstrate the issues that arise when the robot arm is not orientated correctly. The first two images the planes are all at a base orientation, with the robot facing upright. Printing this way the robot moves back, pushing downwards and causing a kink in the print.

Re orientating the plane allows for the robot to make these movements, care must be taken in the programming as double planes at one point are required for the robot to re-orientate. Wait times must also be factored into this as double planes in one spot must not result in double the wait time.

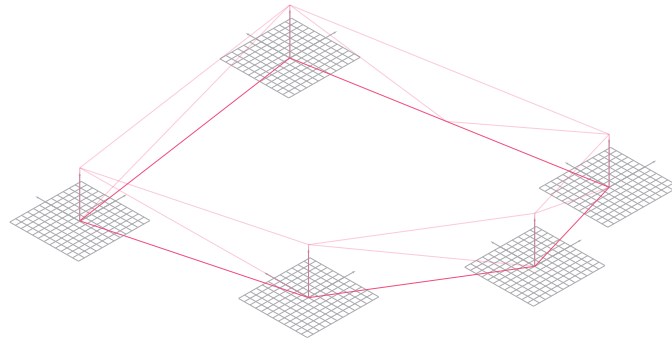


Figure 55. Base layer of cell printed with upright robot orientation

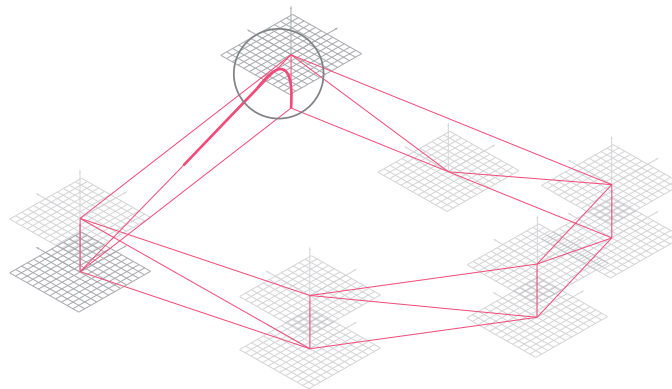


Figure 56. Result at second layer if robot prints form upright

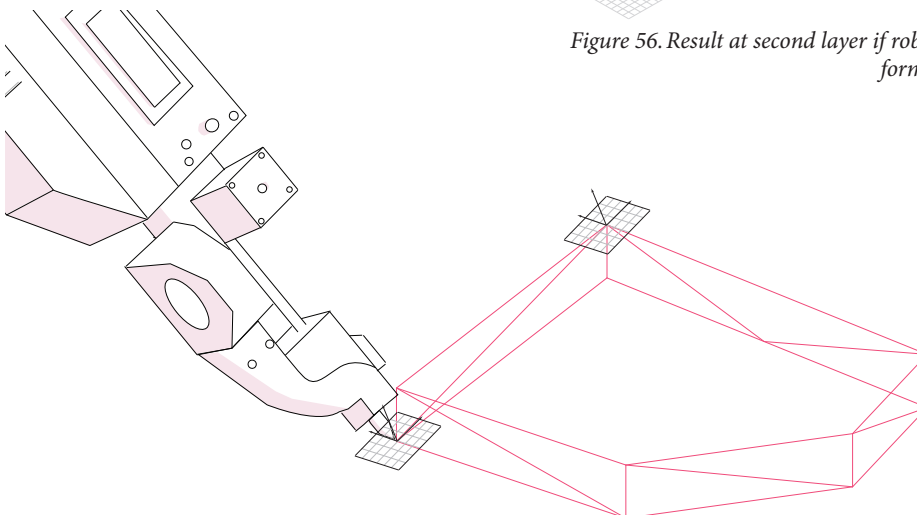


Figure 57. Orientation on diagonal struts at an angle to ensure no plastic is caught by extruder.

2.01



print number	2.01
intention	print single voronoi cell
outcome	fail
reflection	print speed too quick - PLA was unable to cool and set.

2.02



print number	2.02
intention	print single voronoi cell
outcome	fail
reflection	print speed too quick - diagonal connections unable to form and sagged.

2.03



print number	2.03
intention	print single voronoi cell
outcome	fail
reflection	print speed too quick - PLA was unable to cool and set.

2.04



print number	2.04
intention	print single voronoi cell
outcome	fail
reflection	error when programming the orientation angles - angle on the final side of the print wasn't enough to stop the extruder pushing plastic down.

2.05



print number 2.05

intention print single voronoi cell

outcome printed

reflection print finished with the correct robot arm orientation - height of cell not too high - could be increased for clearer connections

2.06



print number 2.07

intention print single voronoi cell

outcome printed

reflection increase in cell height resulted stronger connections.

2.07



print number 2.07

intention print single voronoi cell

outcome printed

reflection increase in cell size - the larger size was able to be accommodated even with the long spans on the one side.

2.08



print number 2.08

intention print double voronoi cell

outcome fail

reflection print stopped on upper layer - the movement was slightly off - likely due to the second layer being too high - causing the lower side of the top layer to not stick.

2.09



print number	2.09
intention	print double voronoi cell
outcome	printed
reflection	adjustment of tolerance for second layer resulted in correct print - previously the robot thought the print was higher than it was - due to sagging the physical model didn't match the digital.

2.10



print number	2.10
intention	print triple voronoi cell
outcome	fail
reflection	likely the same issue as print iteration 2.08 - issues with tolerances caused the print to fail at the third level.

2.11



print number	2.11
intention	print triple voronoi cell
outcome	fail
reflection	the same issue as print iteration 2.08 and 2.10 - issues with tolerances caused the print to fail at the third level.

2.12



print number	2.12
intention	print twin voronoi cell
outcome	fail
reflection	horizontal differences between the sides of the larger cell meant that the printer couldn't achieve such a long horizontal and was unable to stick.

2.13



print number 2.13

intention print twin voronoi cell

outcome printed

reflection decrease in the scale of the cell meant the design was able to be printed with correct joints.

2.14



print number 2.16

intention print twin voronoi cell with angles

outcome printed

reflection location of the upper level being at an outwards angle to the lower didn't affect the same script able to be run.

2.15



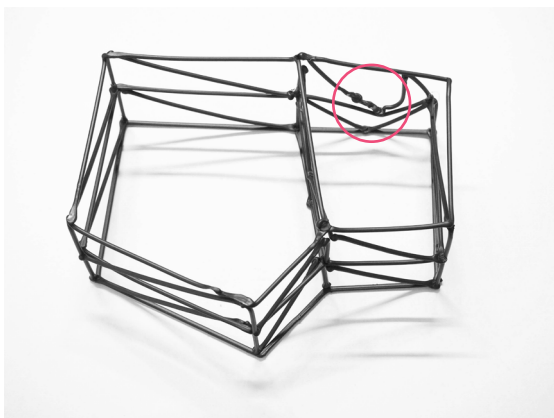
print number 2.15

intention print double twin voronoi cell

outcome fail

reflection the same issue as print iteration 2.08, 2.10 and 2.11 - issues with tolerances caused the print to fail at the third level.

2.16



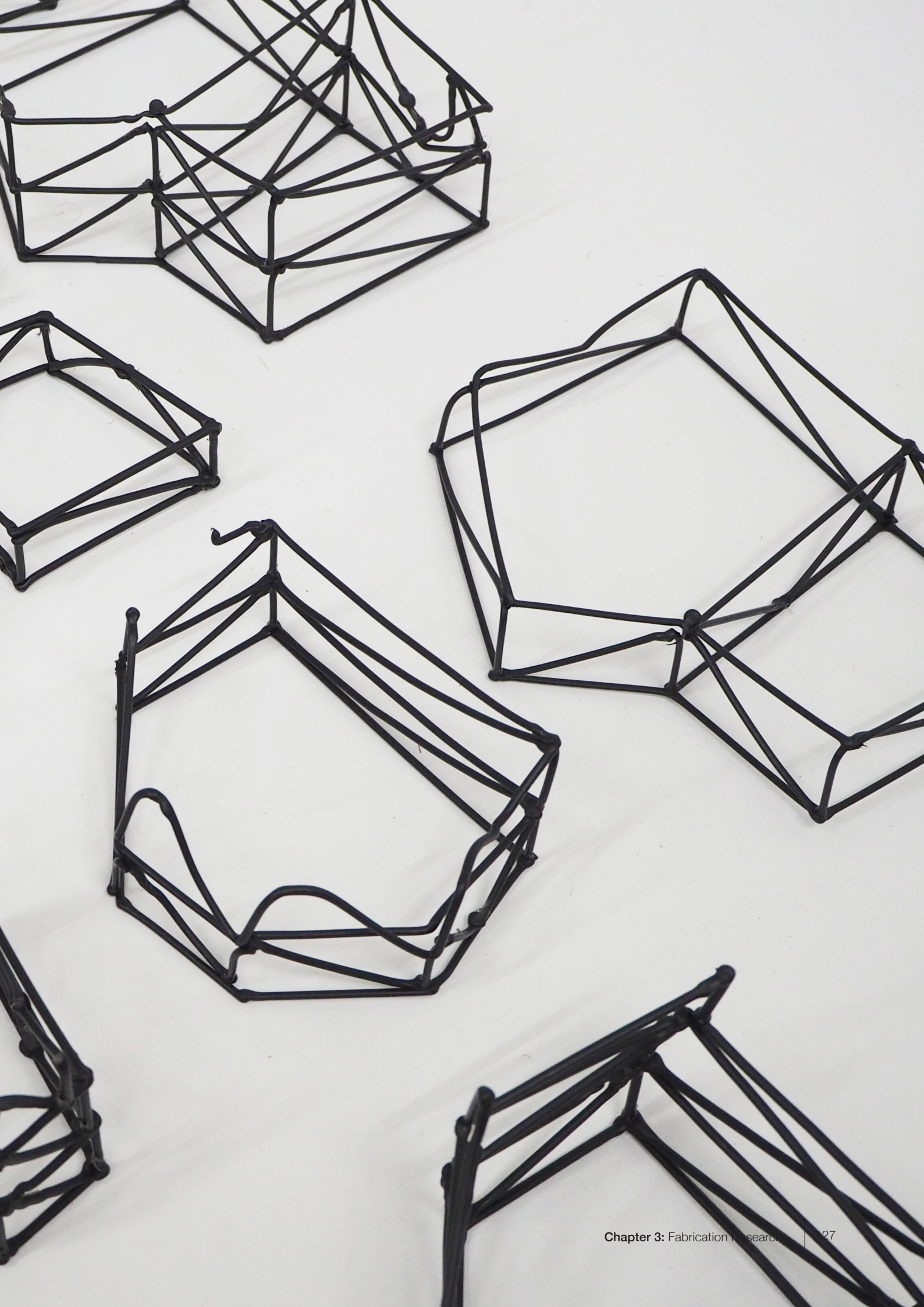
print number 2.16

intention print double twin voronoi cell

outcome printed

reflection while there was a single error in the print - shown in photo - the print itself formed properly - a scripting error in the circle meant the printed interrupted previously printed plastic.





SERIES 3

Taking the knowledge from precedent research and previous experience with the printer available, a series of lattice structures were modelled in Grasshopper, with an aim to begin latticing single lines. This technique will then be introduced to the polylines that are the output from each of the agent-based structures.

A series of different latticing techniques were modelled. Curves were tested against the potential of wait times. The previous series included wait times at each junction or point where the robot changed direction or orientation. For each point of direction change to be enforced in curves will cause an issue with multiple wait times along a single curve.

To combat this the curves were instead modelled as straight lines, but left as a longer member to encourage the PLA to sag over time, creating a curved effect. Workarounds such as this are required if the print is wanting to achieve certain aesthetic qualities as the extruder and coding has certain limitations that prevent this.

The following experiments have been printed using a feedback loop method, this section of the test prints focusses on building a stable model using best print practice, rather than aiming for a direct copy of the digital model.

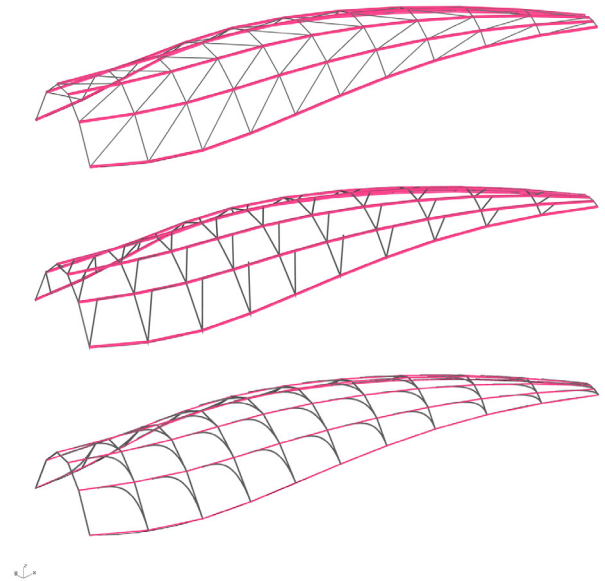


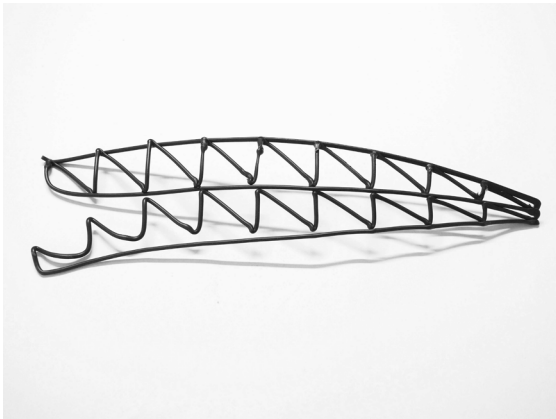
Figure 58. Simple latticed curves

3.01



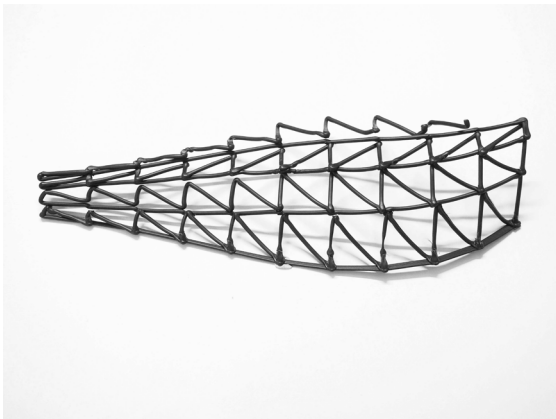
print number	3.01
intention	print wing shape with lattice
outcome	fail
reflection	tolerances were not scripted correctly - the change to having the second layer at both a y and z offset results in the gravitational effect on the print being unpredictable at first.

3.02



print number	3.02
intention	print wing shape with lattice
outcome	fail
reflection	second curve able to be printed - but tolerances still causing issue with connection points.

3.03



print number	3.03
intention	print wing shape with lattice
outcome	printed
reflection	tolerances now able to be calculated and accounted for in print script with an offset.

3.04



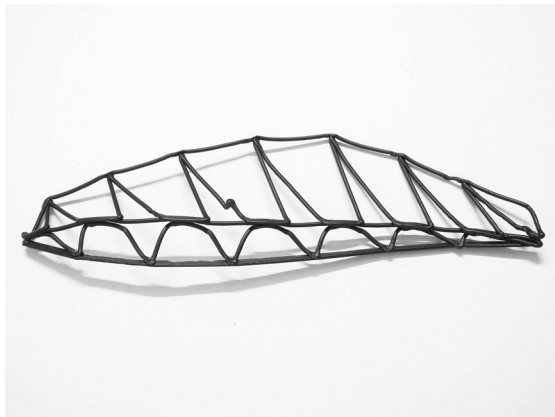
print number	3.04
intention	print wing shape with lattice
outcome	printed
reflection	the tolerances are a smaller factor to consider when the curves are printed closer to one another - giving less chance of the print sagging as there is a smaller mass.

3.05



print number	3.05
intention	print wing shape with lattice
outcome	printed
reflection	an issue with this print arose when the extruder wasn't calibrated at the correct height to the work surface - causing the print to be pulled slightly off the surface and the ends overlapping.

3.06



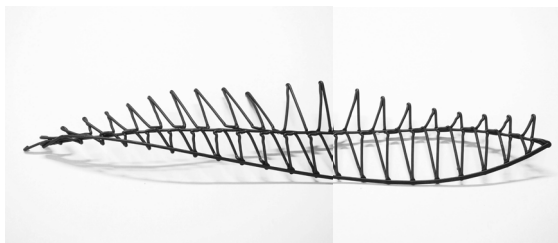
print number	3.06
intention	print wing with different curve pattern
outcome	printed
reflection	the different pattern was able to be accommodate with the same script used in the previous 3 series prints - a small print error occurred, but this was due to operator error and accidentally pausing the robot mid-print.

3.07

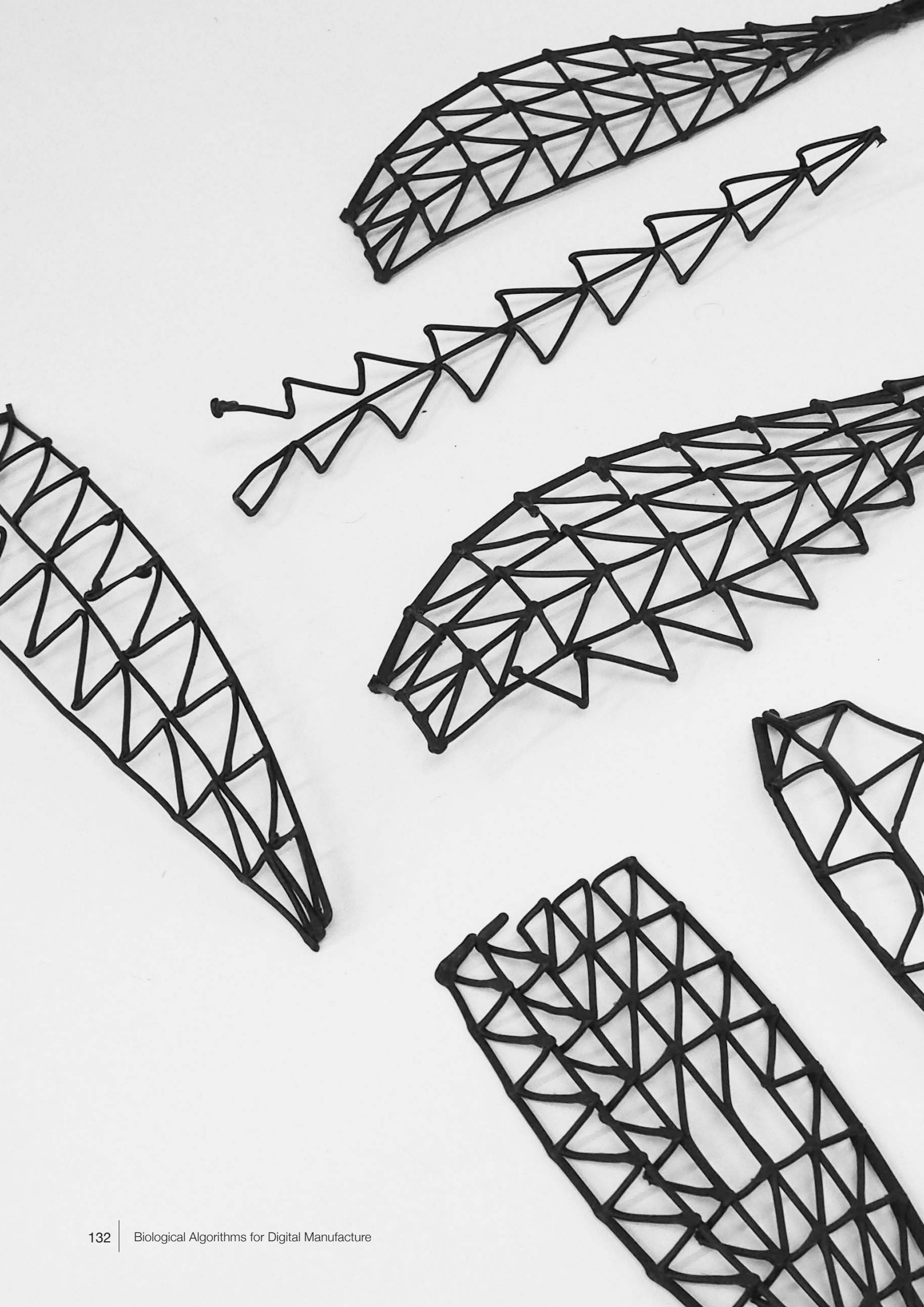


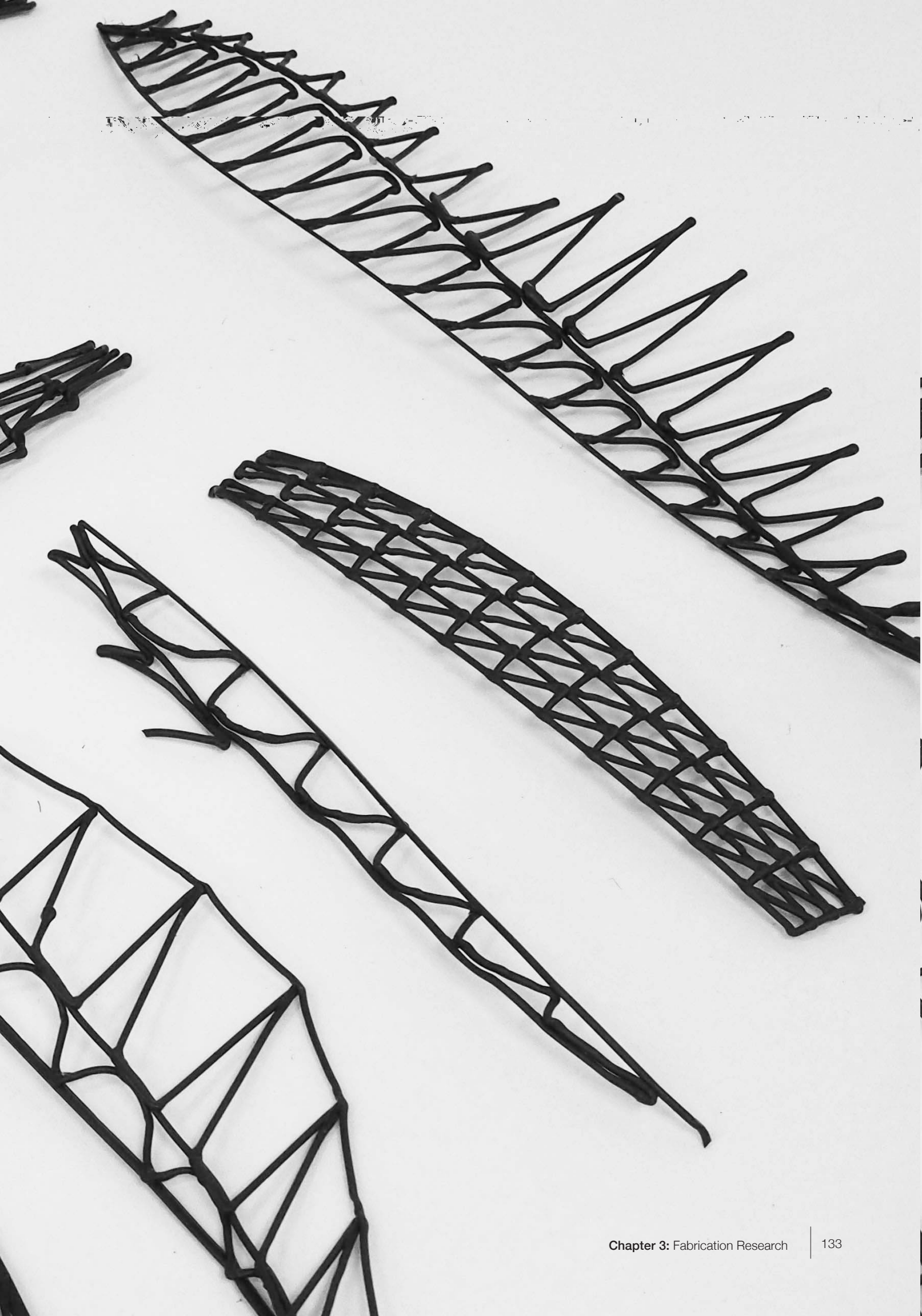
print number	3.07
intention	full wing shape back to the base
outcome	printed
reflection	having the curves form a full wing shape means that the printer is able to potentially build an arch shape - one small error in the scripting resulted in a few lattices being missed - but doesn't affect the overall print order.

3.08



print number	3.08
intention	large version of wing shape
outcome	failed
reflection	with an attempt to push the limits of both the script and the printer - a model twice the length was printed, however the scaling resulted in a greater distance between each curve - which the print wasn't able to connect due to sagging and offset issues.





PRINT ERRORS

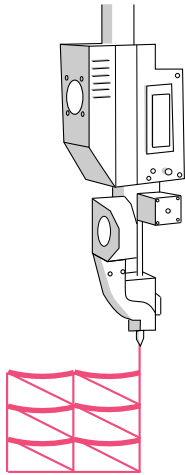
The following diagrams demonstrate errors that arose in the printing of the test models using the robotic arm. The prints themselves have various parameters that are applied and are able to be adjusted to suit the print.

However, during the printing of plastics into a complex spatial lattice, an issue arises with accumulation of tolerances, resulting in imprecisions or even collapse of the structures.

This is caused by the nonlinear behaviours of materials being printed, and a high number of connection points that very quickly accumulate errors—particularly in the case of extruding 3D lattices through the air where the material is only supported and connected through the nodes, while being suspended in the air the rest of the time.

This affects both the vertical tolerances and the transverse connections. The print ordering involves preventing areas from duplicating over top of one another, this needs to be overcome before printing successful models. There are limitations in both the programming of the robot and the printer itself that have caused errors.

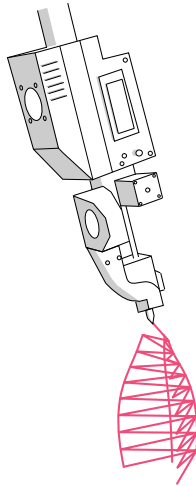
Of the parameters able to be customised through the RAPID Code, the speed is vital to have correct. For each model the robot speed was set around 1.2mm per second. Where each of the prints took over 40 minutes each. Resulting in a long time period to form a feedback to model improvements for future models. While these problems have an ability to be solved through basic script edits and changes in the parameters of the print and robot behaviours, to print the swarm structures further development of the ordering of each line will need to be explored at a variety of scales.



VERTICAL TOLERANCES

As the print adds layers, the effect of gravity accumulates and if the plastic isn't solidified it begins to sag. The result of this is a print that has weaknesses in the horizontals as the plastic has become stretched.

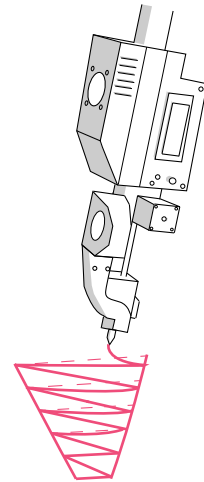
Offsetting each layer incrementally will accommodate the moving of the plastic at each layer.



PRINT ORDERING

The script ordering and print order of the curves can become out of place - if the printer interrupts itself it is likely to destruct a section of the print. With crossing lines this becomes an issue.

Using route finding algorithms and double checking section of the script will minimise the risk of print failure.



TRANSVERSE CONNECTIONS

When the printer is building longer horizontal members in a print there is a risk of the connection drooping slightly and not joining the print at the correct point. This causes an accumulative error the taller the print.

Controlling the maximum length of horizontal members will limit the chances of horizontal members failing at connections.

3.15

AGENTS BASED STRUCTURES PRINTED

Taking the knowledge from the small scale lattice system, it is possible to apply it to a large scale print in the form of an agent-based structure. However with this latticing system, the complex makeup of the agent systems will struggle when translated into a printed system. The nature of the printer printing in one continuous line results in many designs unable to be printed correctly without causing errors. The errors that were discussed resulted in cosmetic and structural issues, whereas attempting to print images such as in Figure 60, the network of lines is very complex and will become out of order.

Route inspections problems could potentially be applied to find the shortest path or closed circuit – resulting in a route that the 3D printer visits every line of the design. Also known as Dijkstra's Algorithm, it is used to calculate the shortest path between one vertex, and every other vertex in the model. The programming of this would require specific conditions – an even amount of lines are required for the graph (3D print model) to be a Eulerian circuit. Each vertex must be connected to an even amount of other vertices's.

Enforcing rules is beyond the scope of this project – it would require Python scripting to develop an algorithm to build networks between agent paths. If this were to be achieved, further programming would be required to overcome the ordering and Rapid Code.

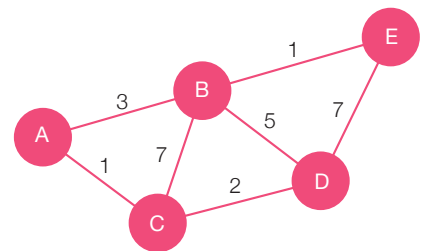


Figure 59. Dijkstra's Algorithm (Coding Game, 2018)

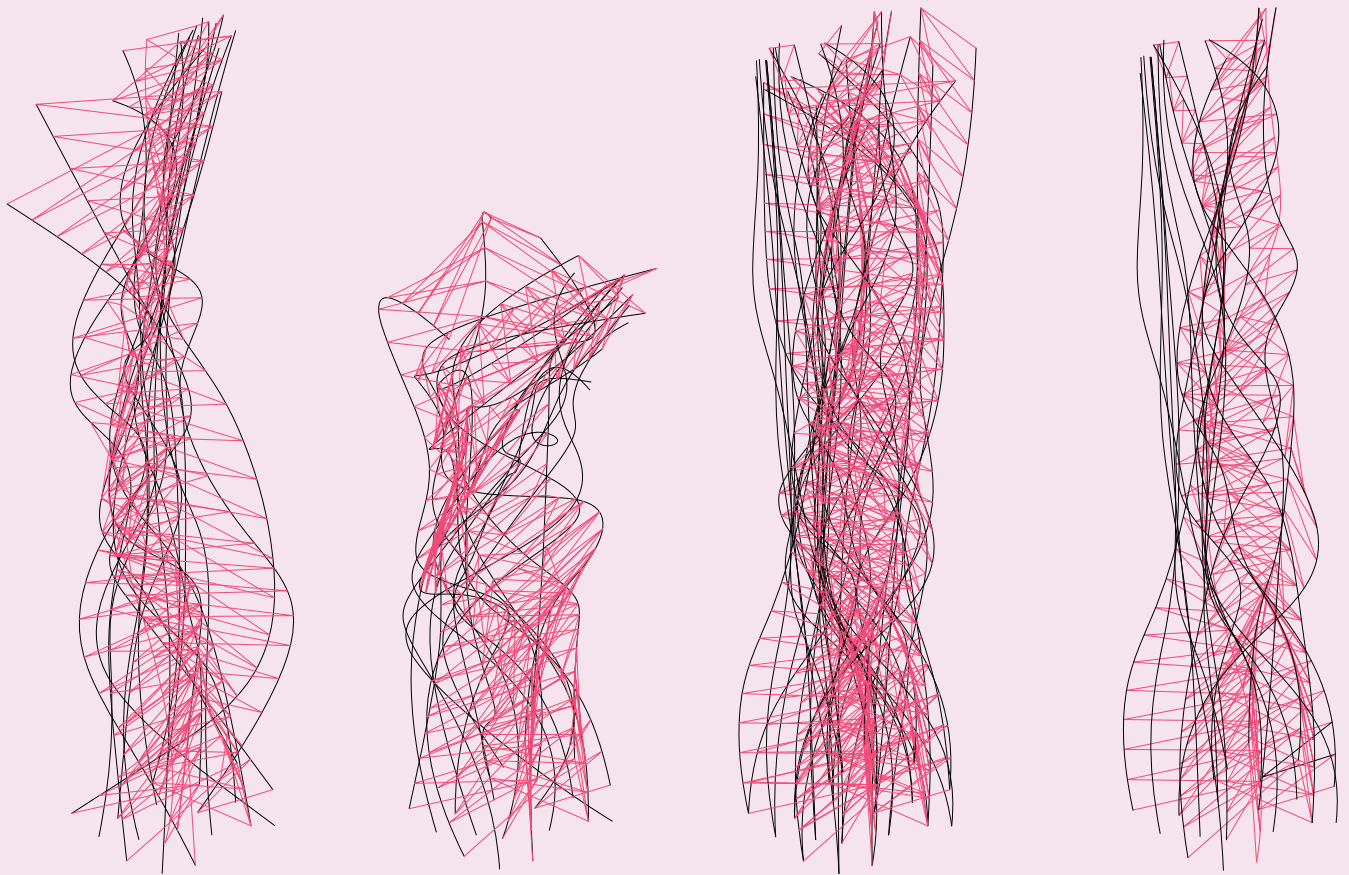


Figure 60. Latticed pattern of agent-based structures

3.16

FABRICATION FEEDBACK LOOP

Through previous findings of the prints that the current extruder produces, it has been found that the printer has many inputs and controls that need to be correctly programmed for the print to be successful. Using a continuous feedback loop is the quickest and most effective way of ensuring this process, through the testing and evaluation of both digital and physical models. Initially the 3D print models are generated in Grasshopper, and taken through to a second phase of the script which translates these into tool-paths. A cluster of components are used to translate a series of curves into planes and commands for the robot to read as RAPID Code. This is then exported out as a file which is loaded into the robot arm.

Generally the scripts are run in manual mode, this is not fully automated but it allows the user to instantly stop the robot if there is an issue with the extruder. Once the tool-path is completed by the robot. The arm is moved away from the print so no filament continues to extrude onto it. The 3D print is then removed from the workspace.

If these prints involve errors, these are then resolved and parameters of the print altered and a new script exported for fabrication. The extruder is able to be left running in this time, therefore the entire setup process isn't required to be repeated.

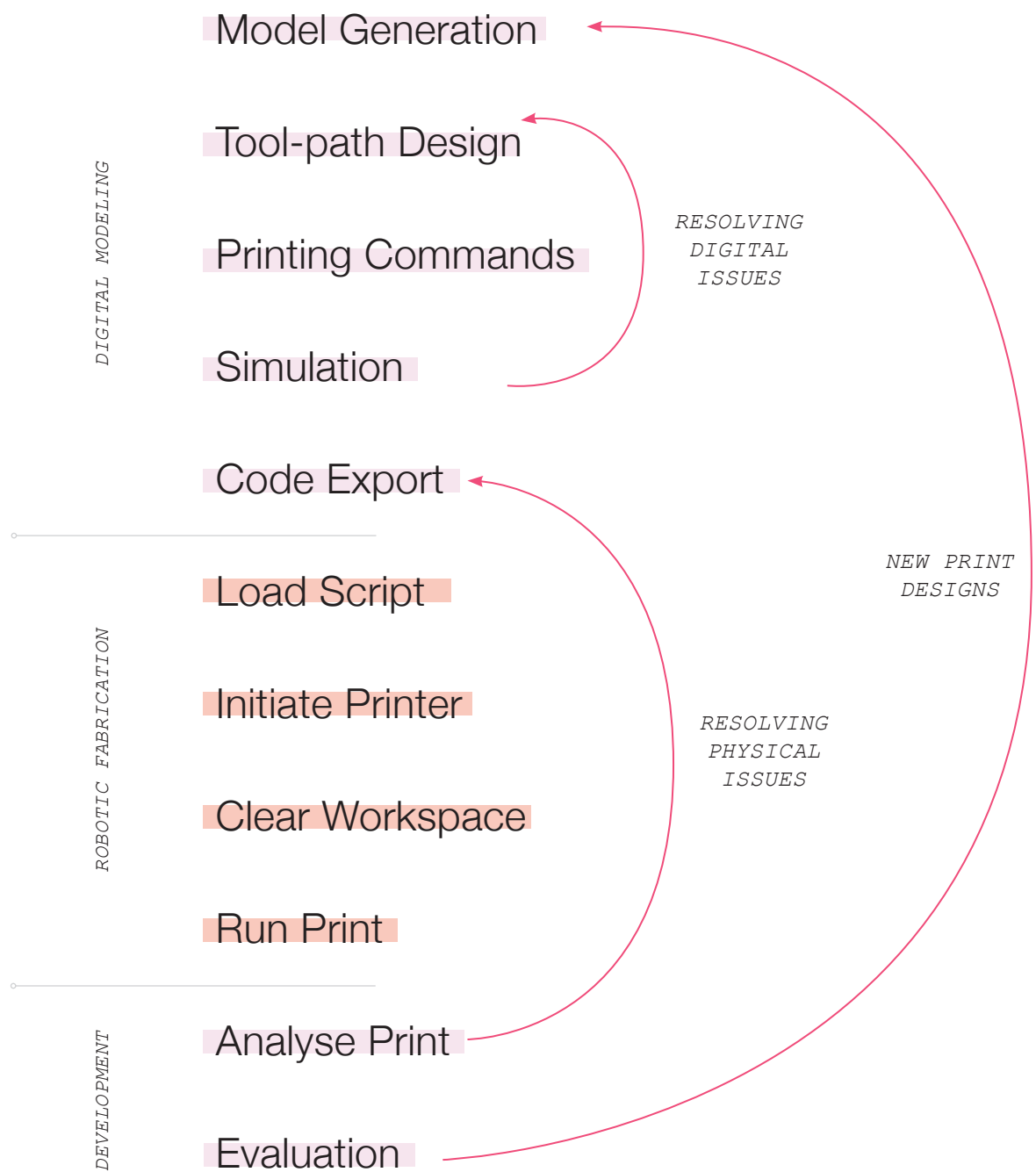


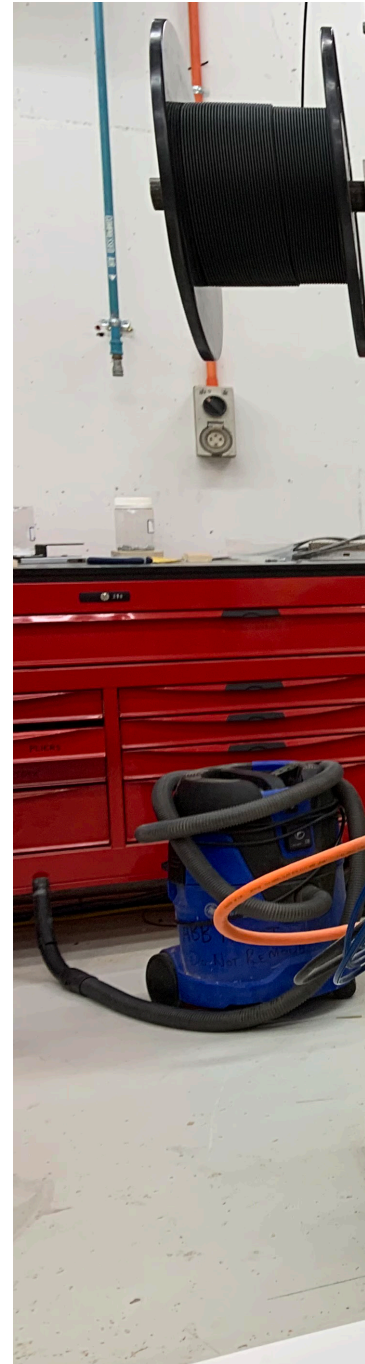
Figure 61. Feedback Loop method

3.17

EXTRUDER DEVELOPMENT

Each of the precedents described in the review above have a different end effector for an industrial robotic arm. Each are unique, in the way that either the educational institution, or company have developed and manufactured their own extruder. The extruder available for this research has been explained, but the possibility in this research can stem far beyond the scope with a further developed printer. The ability for the print to start and stop between objects, as well as a stronger cooling system of the thermoplastic, would allow for the scope of this project to greatly increase. However, the digital design and project methodology remains the same, but limits the output for this research. If it were possible, a second extruder would be used to undertake the next phase of this thesis.

A second extruder has been manufactured by a VUW design student for the robot. This is still in an experimental stage, so using this would involve its own set of limitations and parameters, but may bring about possibility to work around route inspection problems. The printer has the ability to stop extruding plastic and restart at a different point, taking away the problems with a single line programming, but it is not able to be used at this point in the research.



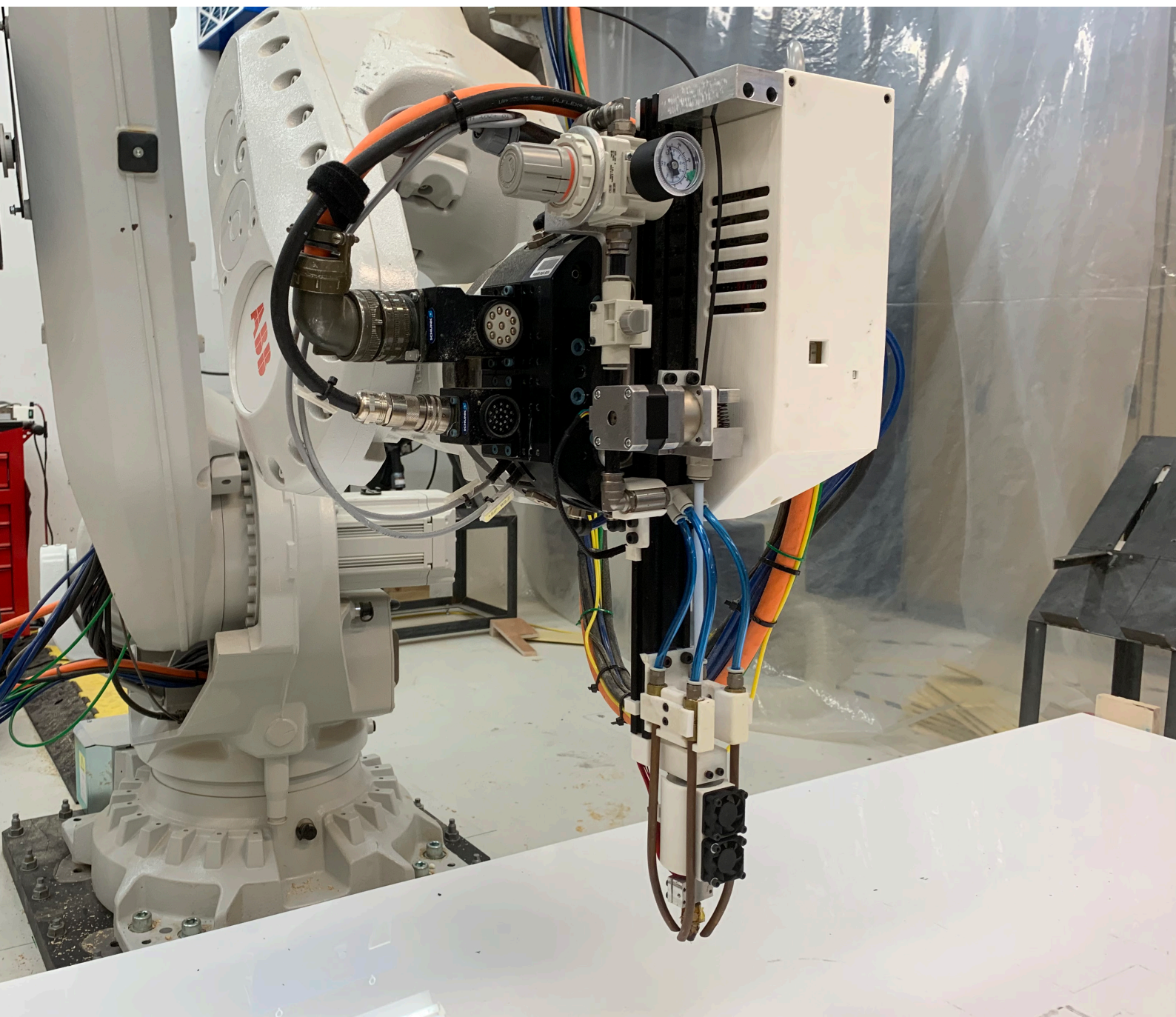


Figure 62. Second extruder model

3.18

PRINTING METHODS

These four images depict a series of different techniques outlined in Chapter 3 used for achieving different elements of a free-form print.

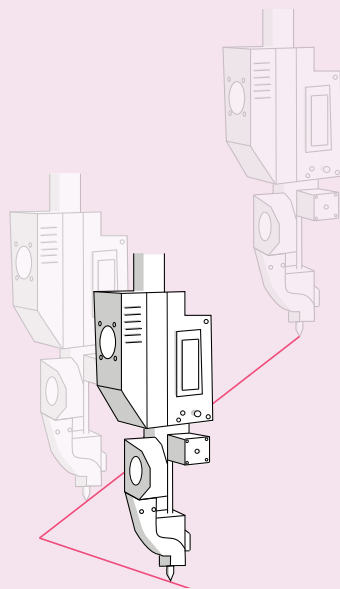


Figure 63. Complete model on XY planes

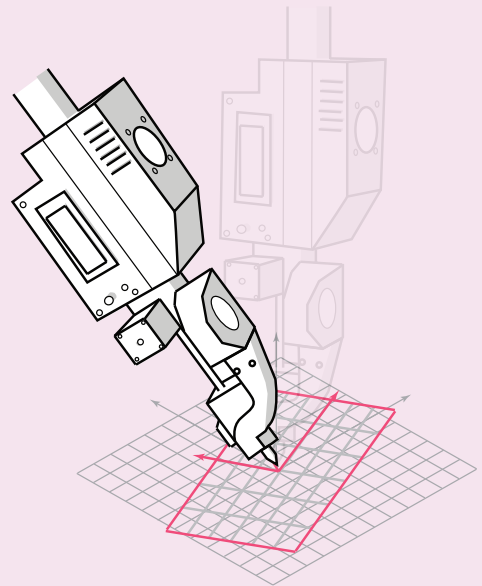


Figure 64. Plane reoriented at point

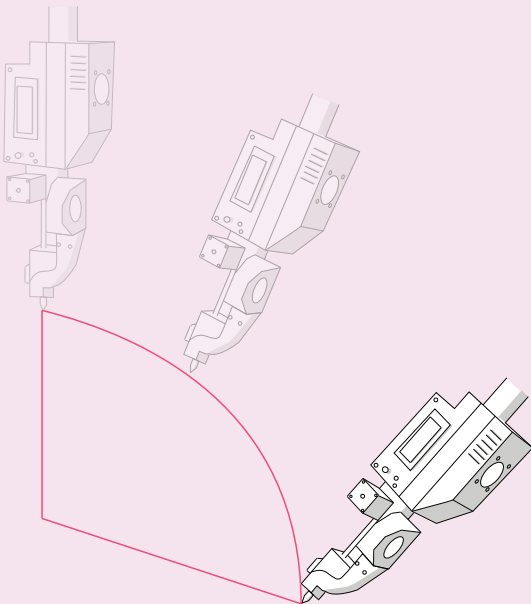


Figure 65. Reorienting along curve

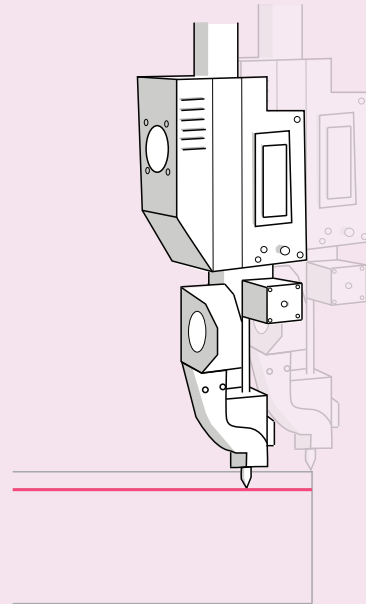


Figure 66. Offset at above level

CONCLUSION

Experimentation and development in the fabrication process has highlighted both the ability that agent-based systems have to improve free-form 3D printing, but also its necessity to be included in developing a system to be fabricated.

Each aspect that was identified and altered during the test prints were the result of using a feedback loop to edit previous faults and modify corresponding parameters. Although the modifications that were made were very minor and unaffected the form of the models, each made an imperative step towards creating the most efficient setup for extrusion. The result of this chapter is a fine tuned method and parameter that are able to be applied to a range of scales in the next phases of the research.

- **Reorientation of planes** – ensuring robot is orientated so printer does not interfere with print
- **Calculating offset** – allowing for gravitational sag on the print
- **Surface calibration** – correct height of extruder for sufficient adhesion with surface
- **Print speed** – ideal print speed to allow material to cool
- **Wait times** – set wait times to allow material to connect at junctions

Improvement in the scripting methods and understanding of the robots ability also enabled the ordering and layout of the movements to be streamlined. Interference of the robot and the cooling vent surrounding the nozzle decreased over the course of the initial experiments with this being taken into consideration in the digital model, removing the need for physical testing.

CHAPTER 4:

DIGITAL PROTOTYPE + FABRICATION

/ Biological Algorithms for Digital Manufacture

RESEARCH REFLECTION

Through the series of digital and physical tests, I have found that it is near impossible to get the models to work using a scripting process or a route-finding solution. From here I referred back to the review of precedent projects earlier in the research, where a key element is a voxel-based system.

voxel; a volume constituting of notional three dimensional space

3D printing historically is a layer based project – a simple foolproof system of building incrementally, this avoids any possibility of the print nozzle interfering with previously printed plastic. To avoid taking this research in an opposite direction, a voxel-based approach will give a similar ordered approach without going back to an incremental or grid based fabrication system.

Voxelisation is a way of streamlining messy volumetric data into a fragmented state. The overall composition remains. Subdividing continuous curves into individual data develops an unknown form by using the power of computation and the discrete ability of simulating natural phenomena.

Physically the agent simulations lie out of the scope of this project to completely program and print, but through data manipulation these can be broken into discrete pieces, using pixels or voxels.

The next phase of this research instead moves in a separate direction and works towards building a voxel-based fabrication system. This system will be able to interpret the agent simulations and also be driven by the digital application in the form of environmental controls.

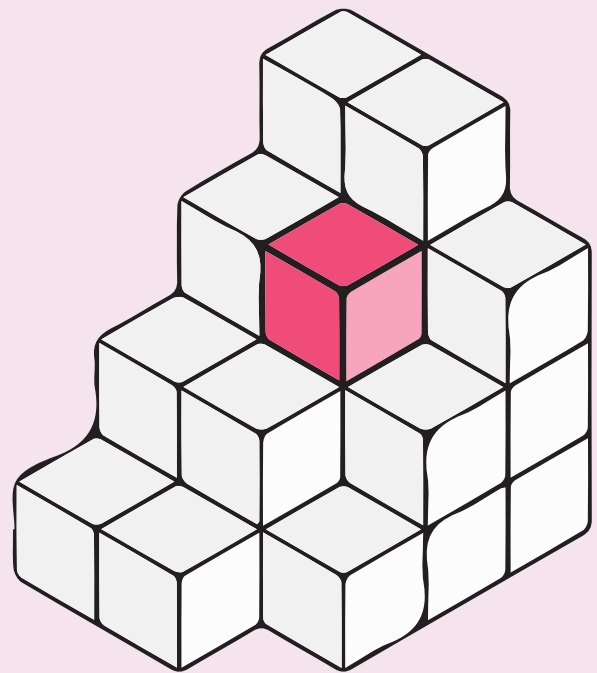


Figure 67. Array of Voxels

VOXELS

Through both a literary review of similar projects, and physical testing of the extruder at Victoria, it is key that the simulations or digital models take into the account that the print must be executed in one continuous line, with no interruptions to material already extruded. At this point in the research the focus for a fabrication system turns towards a voxel-based system.

A voxel, is a form of 3D data, essentially a 3D pixel. In computer programming a voxel is able to contain any amount of data and is able to be customised abundantly and arrayed many times.

While using a voxel-based data structure loses the organic elements of the design, it is likely that this is the only way to begin to fabricate these systems. Moving to a voxel-based structure does allow for customisation within each voxel, this is able to be informed by the agents themselves, but give a grid-like structure that is able to be built incrementally layer by layer. Using a layering approach, the printing process is far less likely to be interrupted by collisions in the print, while there will still be areas of collision, the amount of errors will be greatly decreased. The nature of a voxel-based system allows for scale-ability, a 4x4 grid of voxels is able to be scaled to a 400x400 grid. Working with a fabrication method such as this, allows for the physical testing to remain at a small scale, giving quick feedback and turnaround during the testing phase. Once a digital model succeeds at a small scale, it is also proven to succeed at any larger scale.

The move to a voxel-based structure is an alternative to allow for swarm structures to be fabricated, it will act as a secondary system independent of the agent-based simulations.

4.02

VOXEL DESIGN

Using pixels as an example in 2D, each pixel has the same basic form - a square, but any pixel is able to contain unique data. Pixels are not read one by one, but as an overall composition to form an image. 3D voxels work in the same way, with each voxel containing unique 3D data, that together builds a 3D form.

Each individual voxel in a 3D print is able to have a unique set of data within it, this data is digitally designed and then input into a series of voxel spaces. Each 3D print must have an element of continuity, with an order through the overall form with each voxel able to be connected as one composition.

The individual design possibilities of voxels are almost infinite. As digital models, they are required to have an entry and exit point for the robot arm to move through the voxel to the next, but each are able to be customised. Further considerations must also be made in regards to the structure and stability of the voxel, and building second layers upon a series of voxels requires a series of supports for the layer.

Each of these limitations that have been found through the previous experiments with the extruder are able to be worked into both the individual voxel design and the overall composition. Shown in Figure 68 are a series of designs built for initial testing with the printer, each of these have an entry or an exit point as well as building height for the next layer. The next phase of this research looks into both the design of the voxel and its printing ability.

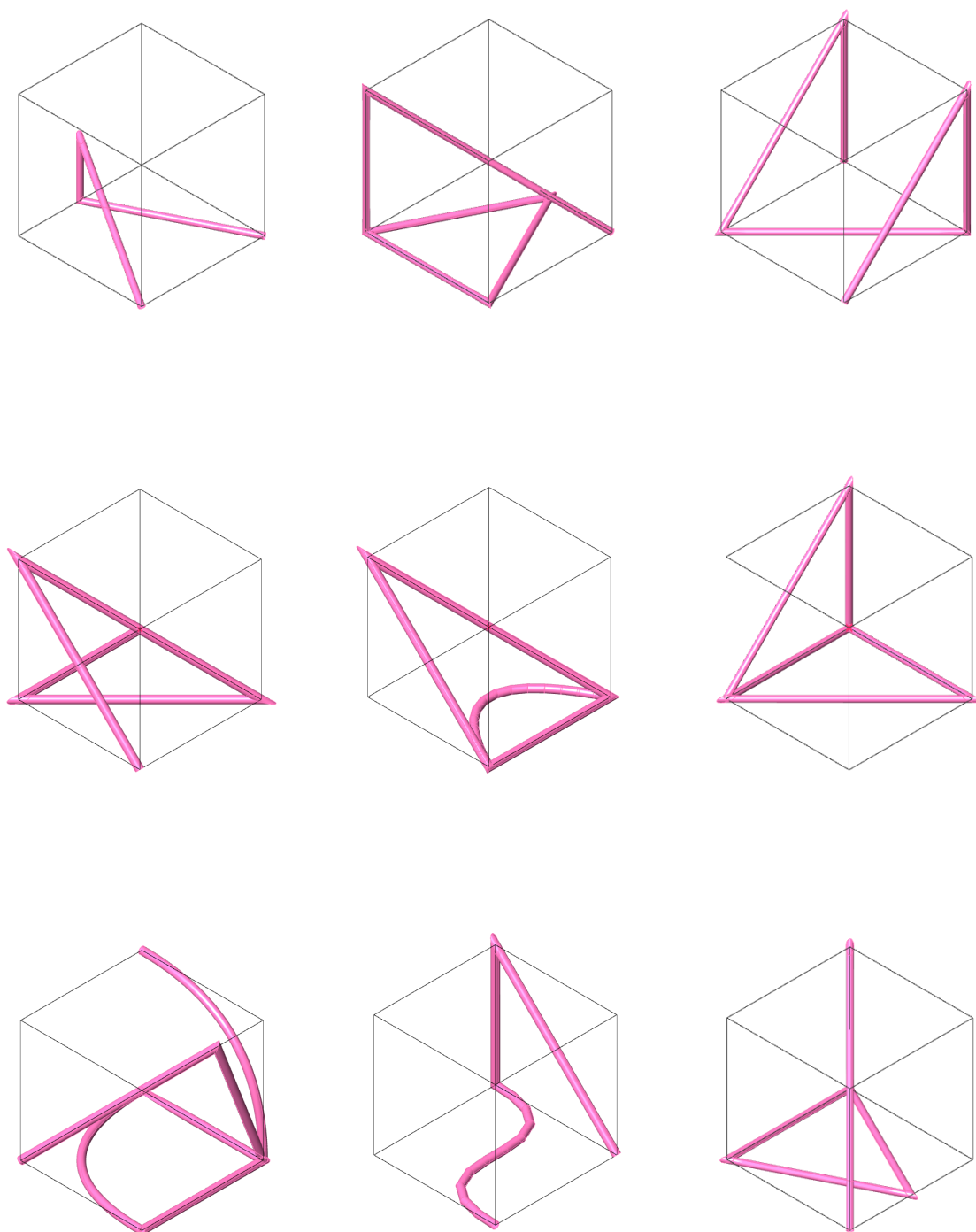


Figure 68. Designed selection of voxels with considerations of 3D printed order

4.03

VOXEL COMBINATION DESIGN

Using parametric and generative software for the programming and the fabrication process of this research gives opportunity to make use of integrated generative solvers in Grasshopper. Galapagos is an evolutionary solver integrated into Grasshopper that is capable of applying evolutionary logic to solve specific problems. This was used to build a series of voxel designs connecting an amount of the vertices's together. Each simulation is then analysed in terms of their curve length, with the longer curves being output as potential voxel designs.

The setup for the simulations are based of the corners of a voxel box, where a variable number are selected and interpolated between. Each simulation has a test factor of the length of the curve in the voxel. Measuring against the curve length gives a higher chance of each voxel having more corners to support surrounding voxels.

The following images describe a series of outputs generated using Galapagos, with a different amount of anchor points for each group of simulations. The models will then be input into digital models at different ranges, with the density of the voxel varying depending on the desired density of the 3D print.

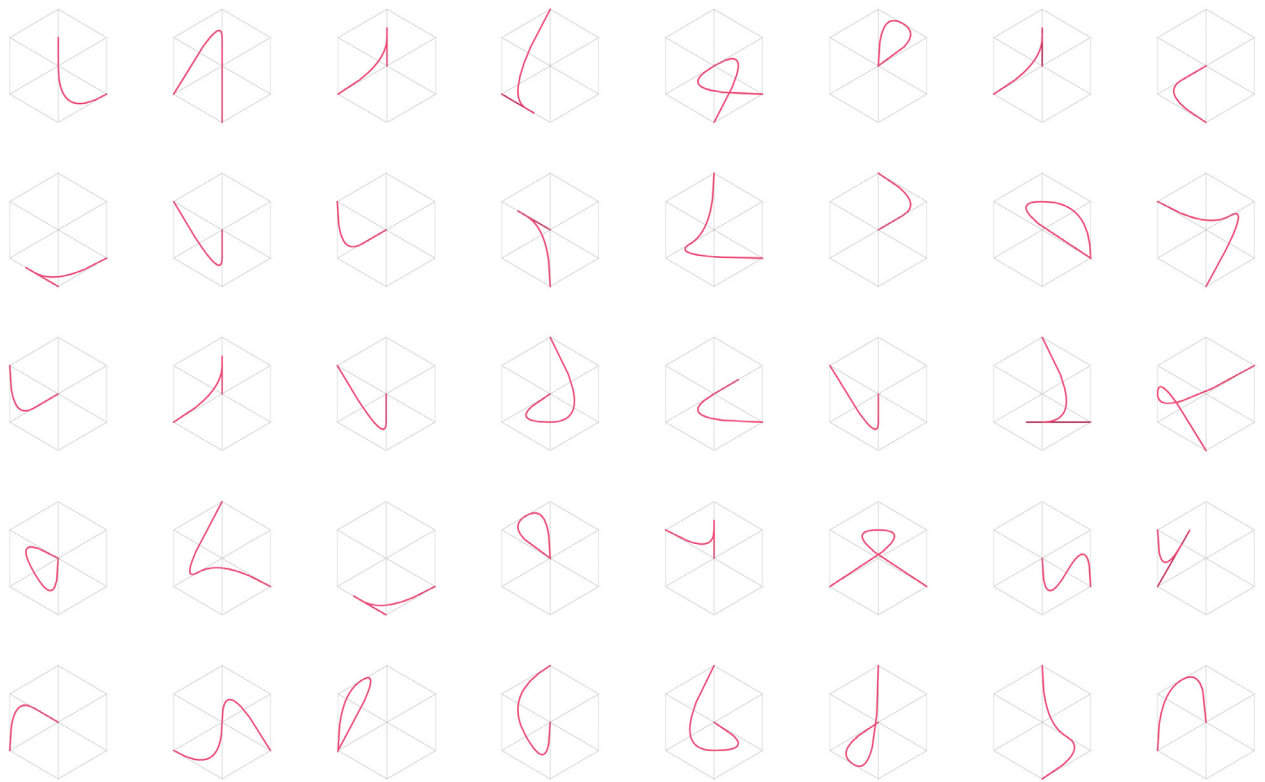


Figure 69. 4 Points



Figure 70. 5 points

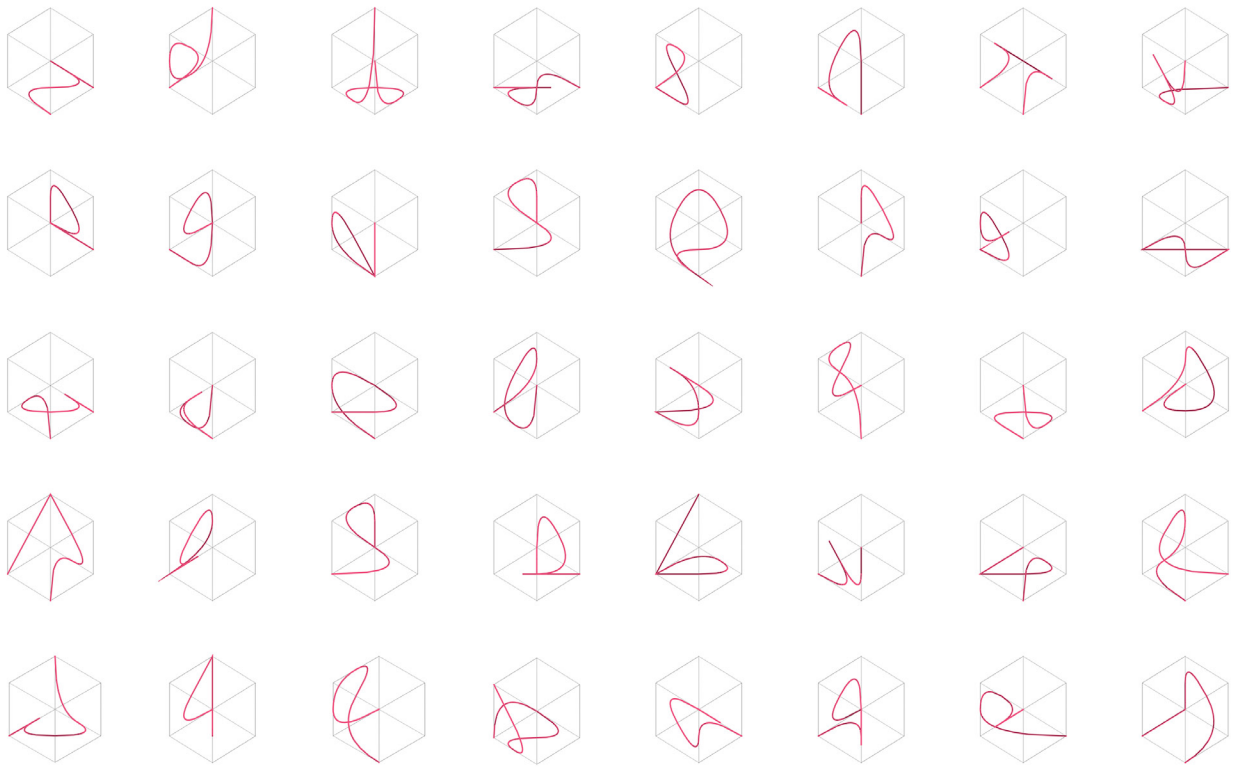


Figure 71. 6 points

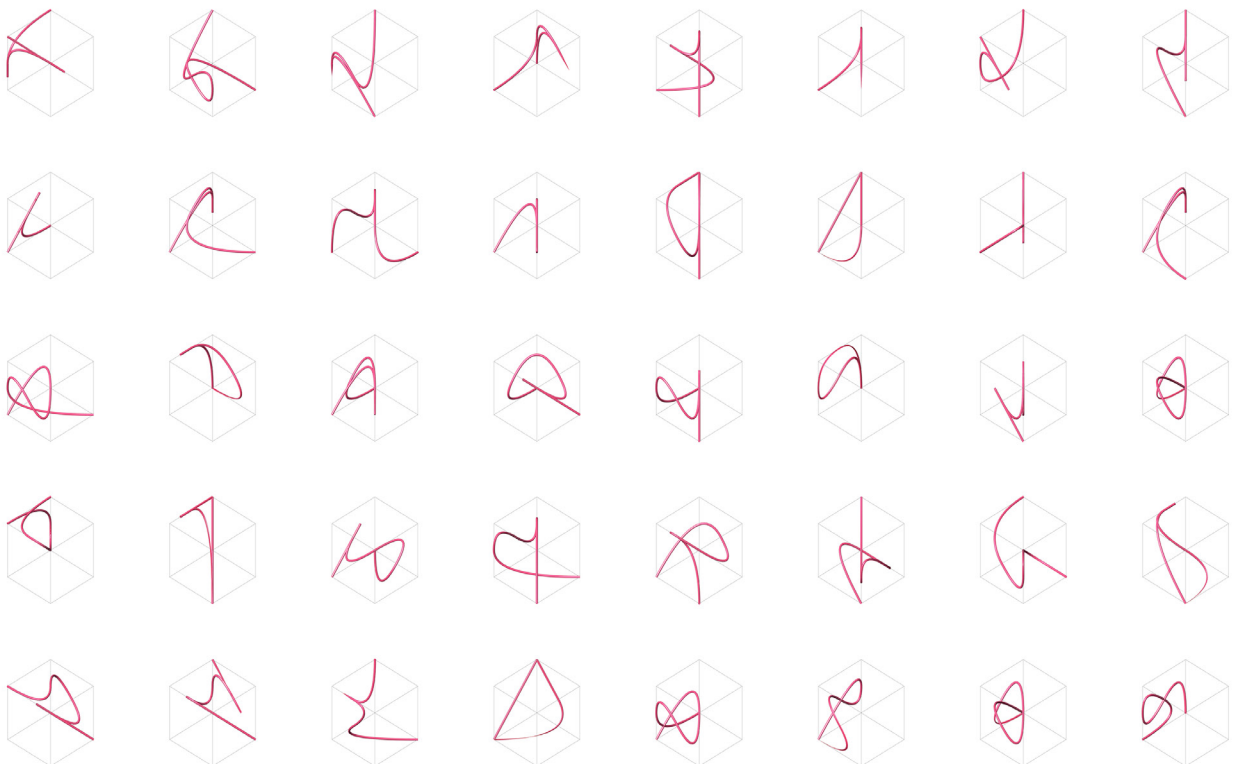


Figure 72. 7 Points

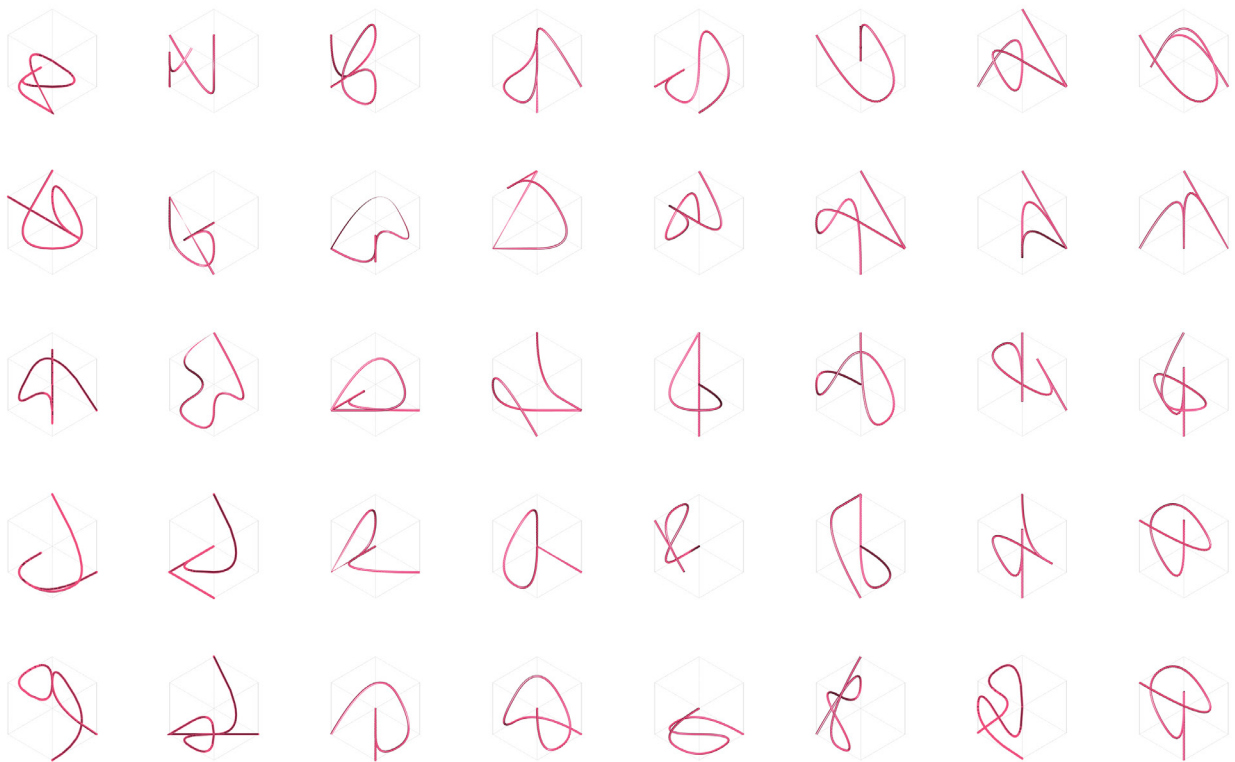


Figure 73. 8 Points

4.04

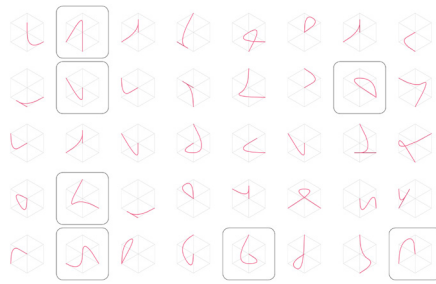
OPTIMISING VOXEL DESIGN

The output of the simulations will then be taken into an analysis process, and separated dependent on their print-ability and density. Density has been included as it gives a selection that will be able to be implemented as controls for sun shading in the Digital Application section of this research for the National Science Challenge.

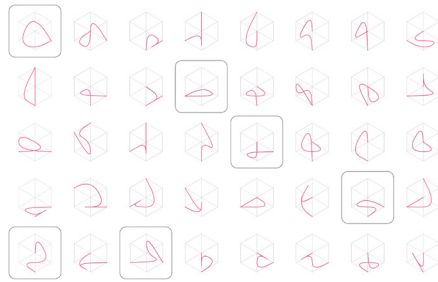
This the voxels have also been categorised showing their exit and entry points – this information is key when determining the voxel design for a certain section of the composition and how it will interact with other sections of the model.

GENERATED DESIGNS

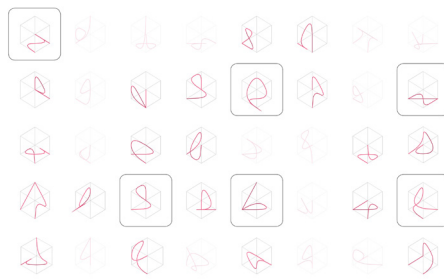
PRINTABLE



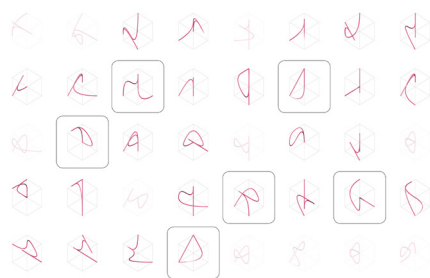
4 points



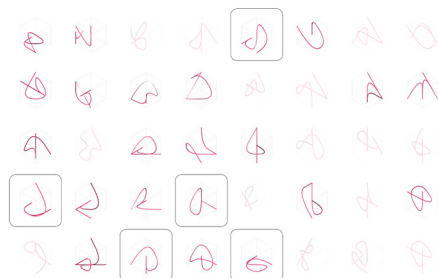
5 points



6 points



7 points



8 points



VARIABLE DENSITY

% KEY

START/EXIT

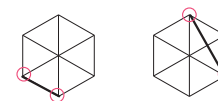
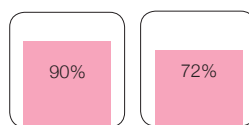
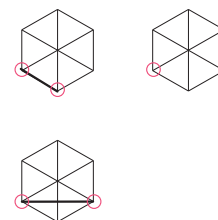
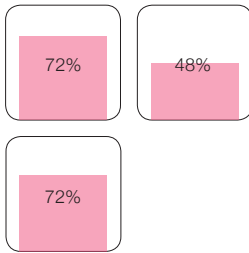
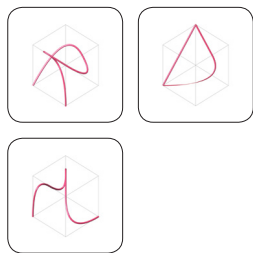
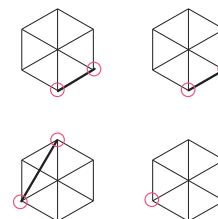
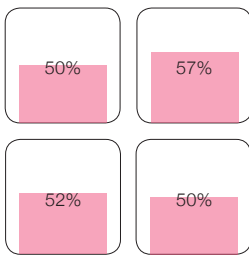
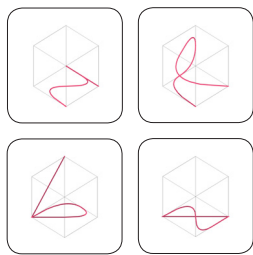
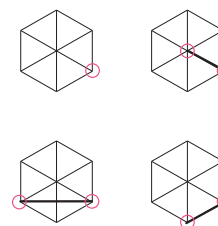
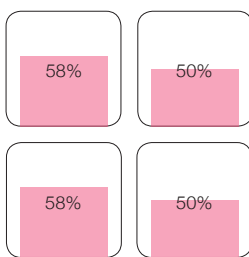
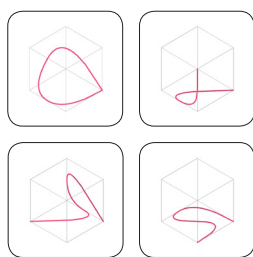
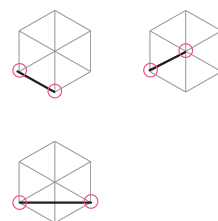
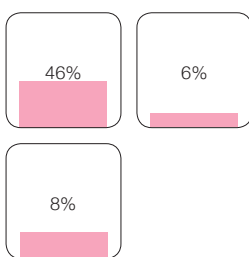
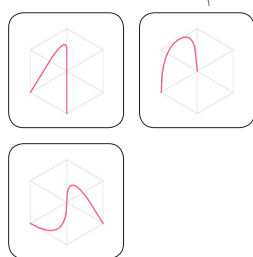
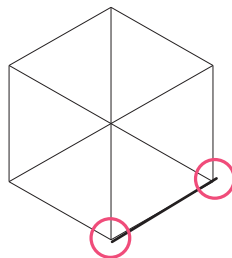
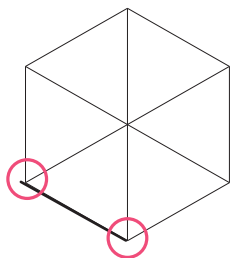
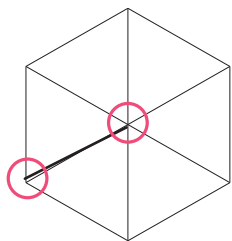
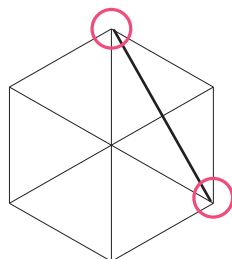
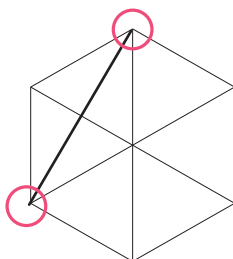


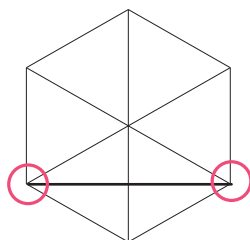
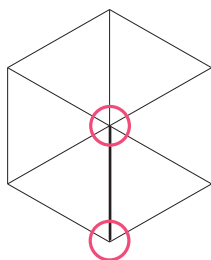
Figure 74. Analysis work-flow of voxel designs



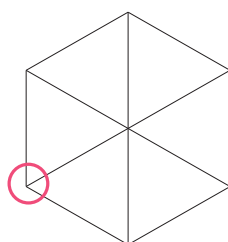
PASSING THROUGH VOXEL



LEVEL OF PRINT



*CROSSING SECTION OF
PRINT*



IN AREA OF PRINT

VOXEL ORDERING

The next stage of the voxel development is to instantiate them within the agent-based system design. For this to be achieved a system that controls the connectivity of one voxel to the next is required. As shown in Figure 75 , the series of voxel designs have a variable amount of movements to pass through the voxel. In categorising potential voxel designs, the voxel voids themselves will be split up into categories in the digital scripting.

Shown in Figure 76, moving through a staggered line print, most of the voxels are able to be similar as the 3D printer will move through the voxel space and out the other side, however when the model turns a corner, a different style voxel is applied in these sections. This is when a crossing or filling design will be used. This print would be relatively simple to implement using code, however as the designs become more complex, thorough data management and route placement will be required to ensure correct printing paths.

When these systems are developed and a potential route inspection algorithm is built, the system will be able to be applied over a range of scales and sizes.

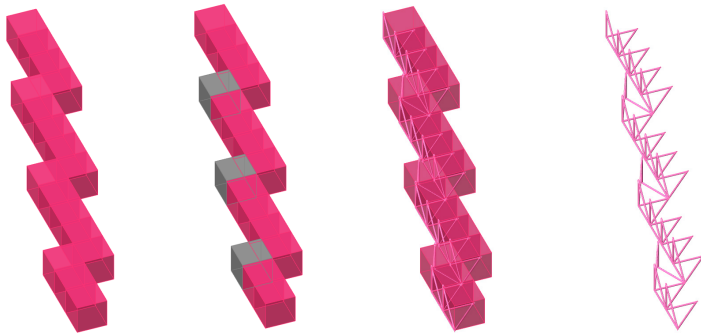


Figure 76. Application of designs to catergorised voxels

4.06

PRINT ITERATIONS

The following images document a comprehensive series of testing multiple voxel designs within a larger print. Testing both the ability of the extruder to print complex designs as well as the connectivity networks between voxels.

Consideration must be given to the print order and the connectivity of one part of a digital model to the next, while previously voxels have been an ideal form structure, the design of the voxel must be able to be printed, and a consecutive design next to it mustn't interrupt or collide with existing voxels.

Increasing the print in the Z direction has a great impact on the accuracy of the print. This has been discussed and is able to be rectified through adjusting the height of the next layer to account for the sag in the print, however this adjustment needs to be varied dependent on the print design. The movement of the robot at the point where it builds support for the consecutive layer determines the amount of pressure causing the print to sag, these movements and reorientations need to be tested and then the second layer of the print adjusted according.

e.g. - changing both orientation and direction of the print results in more plastic being extruded over a longer period of time compared to the robot arm staying in the same orientation and moving to the next voxel.

1.01



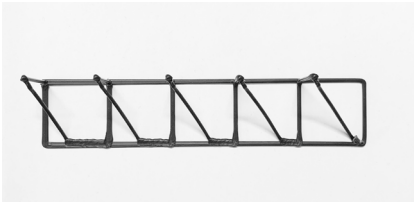
voxel types: 1
intention: Test repetitive motion of simple voxel
outcome: printed
reflection: parts of the model may unstick from the acrylic base when printed upwards - check Z calibration of printer

1.02



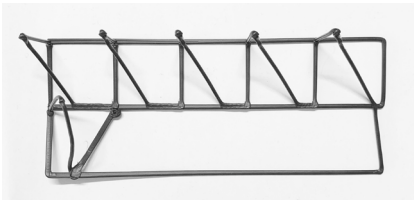
voxel types: 1
intention: Test repetitive motion of simple voxel
outcome: printed
reflection: Having a solid base to stick to gives model stronger overall build

1.03



voxel types: 1-5
intention: Introduce variable voxel parameters
outcome: printed
reflection: having each voxel with variable designs - different lengths of plastic along bottom edge works but repeating same plastic should be avoided

1.04



voxel types: 2
intention: Add second layer to print
outcome: fail
reflection: having each voxel with variable double check angles of model to avoid printer running into own prints

1.05



voxel types: 2
intention: Add second layer to print
outcome: printed
reflection: check for areas where nozzle interference may happen and test print

1.06



voxel types: 3
intention: Add second level
outcome: fail
reflection: account for drooping of plastic - prints to not provide sufficient support for second level

2.01



voxel types: 1
intention: Introduce curved voxel
outcome: printed
reflection: wait times in-between each node need to be removed - this is to ensure a smooth print over a curve - as a curve is made up of multiple points which the robot arm moves between

2.02



voxel types: 1
intention: multiple curved voxels
outcome: fail
reflection: check printer interferences when printing multiple voxels side by side - new voxel may interrupt previous print

2.03



voxel types: 1
intention: multiple curved voxels
outcome: fail
reflection: as previous experiment - 2.02, account printer interferences

2.04



voxel types: 1
intention: multiple curved voxels
outcome: printed
reflection: with the tolerances and interferences accounted for each voxel will print successfully. The ordering of this has to be accounted for in two directions - positive x and y to build a grid

2.05



voxel types: 2
intention: Add second layer to print
outcome: fail
reflection: as experiment 1.06 -account for drooping of plastic - prints to not provide sufficient support for second level

2.06



voxel types: 2
intention: Add second layer to print
outcome: fail
reflection: as experiment 2.05 -account for drooping of plastic - prints to not provide sufficient support for second level

2.07



voxel types: 2
intention: Add second level
outcome: fail
reflection: with accounting for drooping of plastic - prints still not building solid base for second layer - future experiments into printing multiple anchor points for second layer to resolve problem.

3.01



voxel types: 1
intention: replace curves with straight lines to allow for wait times to solidify second layer
outcome: fail
reflection: as previous experiment - 2.02, account printer interferences

3.02



voxel types: 4
intention: Add second layer to print
outcome: printed
reflection: introducing wait times improved the stability of the second layer of the print

3.03



voxel types: 4
intention: Add second layer to print
outcome: printed
reflection: as experiment 3.02 - improving the wait times ensures the second layer prints

3.04



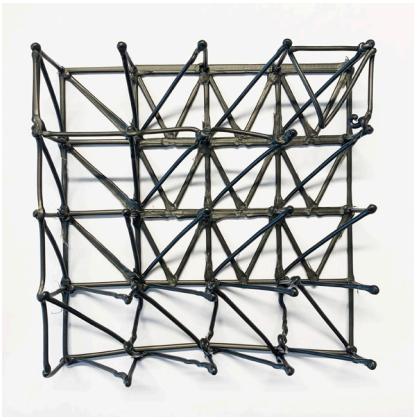
voxel types: 4
intention: and large second layer to print
outcome: printed
reflection: with the fundamentals and the voxel connections working - the design is able to be scaleable in the x and y direction without having any impact on the print model - proving the scaleability of a voxel-based model

4.01



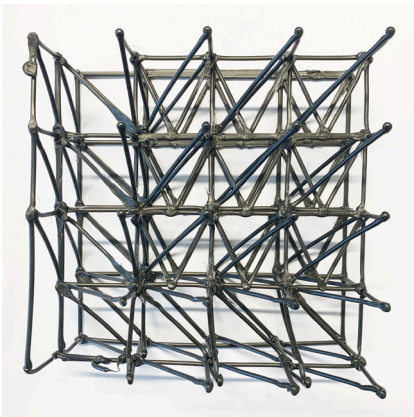
voxel types: 5
intention: build supports for third layer of print
outcome: rough print
reflection: with the tolerances and interferences accounted for each voxel will print successfully. The ordering of this has to be accounted for in two directions - positive x and y to build a grid

4.02



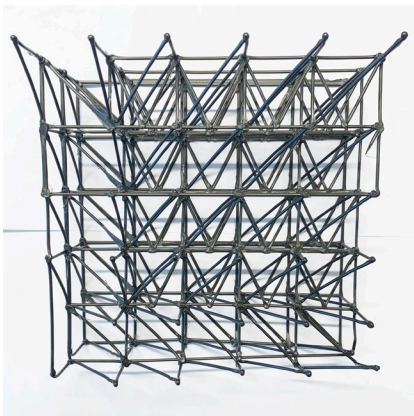
voxel types: 5
intention: test options for building stronger prints that are able to print higher and wider
outcome: printed
reflection: the connections and wait times ensure a solid base - issues arise when building multiple supports on the end of the voxel design

4.03



voxel types: 5
intention: adding further layers to print
outcome: printed
reflection: scaling up the print design works when considering the connections for levels above

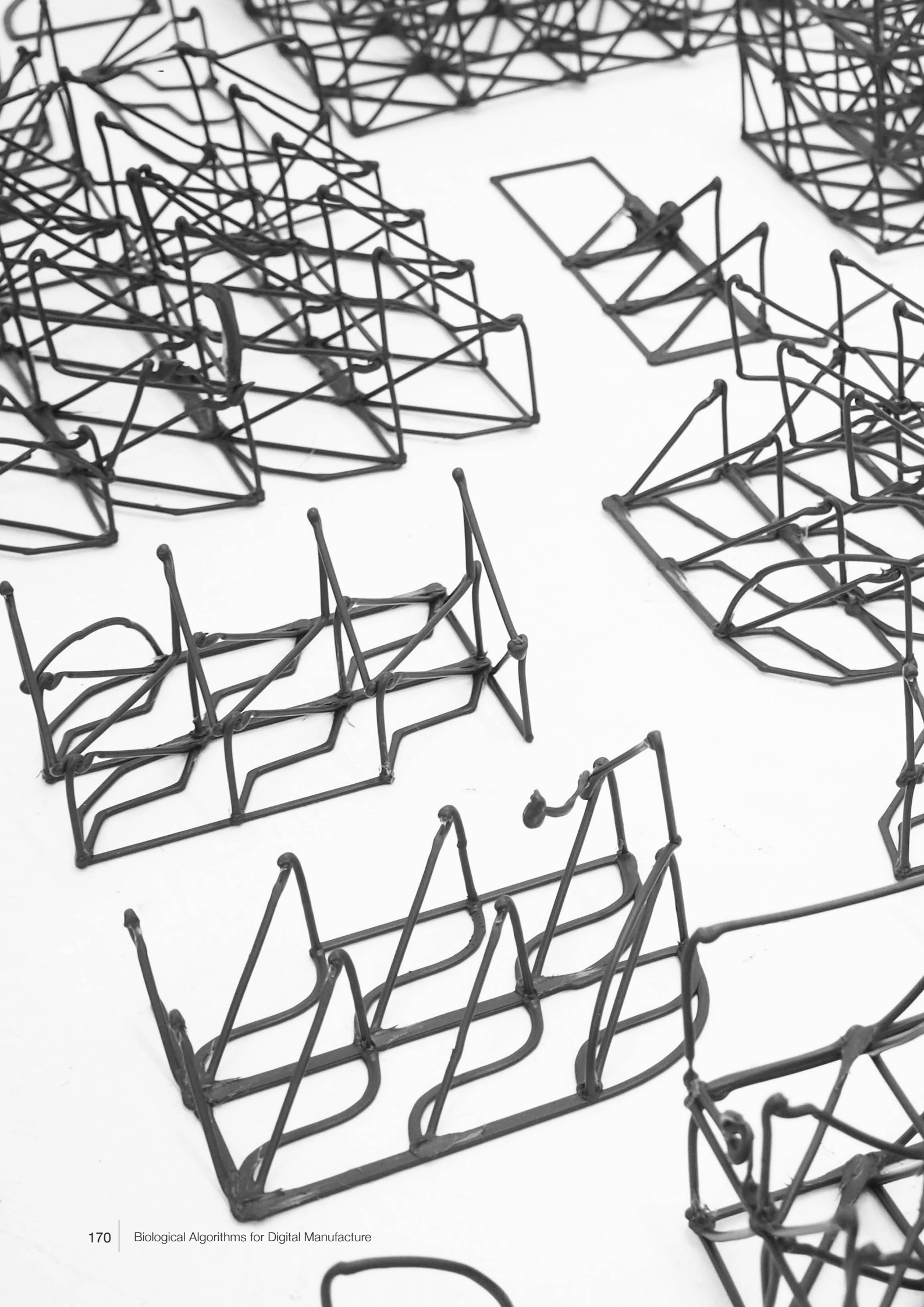
4.04

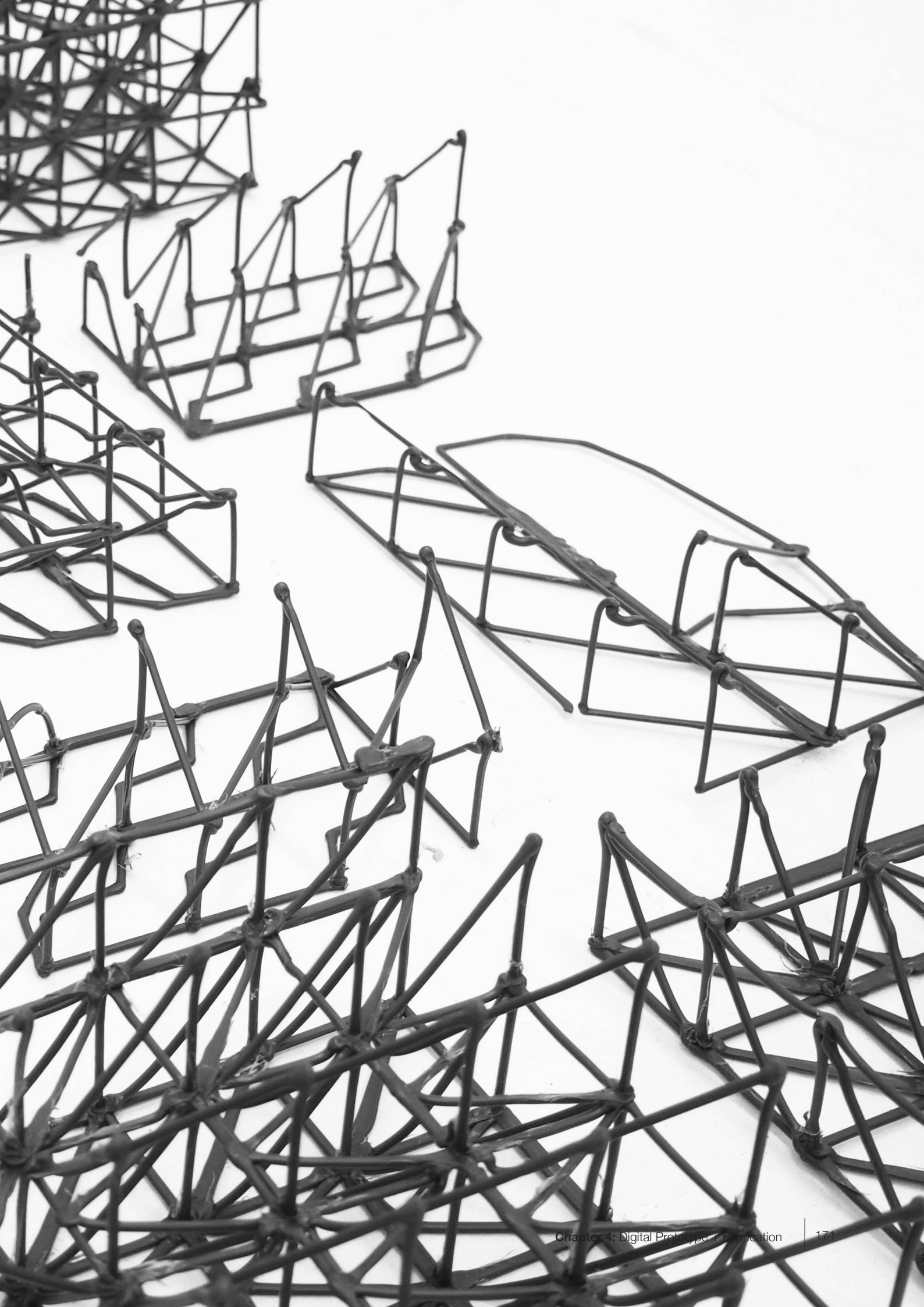


voxel types: 5
intention: increase size of print to 5 x 5 x 5 voxels
outcome: printed - however not to full height
reflection: printer stopped extruding PLA plastic during print so form stopped building - plastic is not coiled up or caught and reprint









VOXEL-BASED SWARM TECTONICS

Using the methodology of both digital and physical testing the previous models, this built a knowledge of the limitations and abilities of voxel-based printing. Data is able to be transferred to a range of voxel designs and then interpreted and programmed into large models ready for print. The next section outlines the development of a physical print informed by agent-based systems and translated into voxel-based models.

Beginning with a basic agent-based model at a medium scale. These designs take a simple simulation and build this as a box-based model. A series of Grasshopper clusters were built to take the curves and build voxels. Using parametric design the box sizes are able to be altered to allow for more or less boxes and variable voxel size.

The networking between the voxels is determined based on both the relationship in the X and Y direction, and then the Z direction, this results in a print that will be built from the bottom up and from one side to the other, avoiding issues with print collisions.

A positive Z relationship between the voxels is key in models such as Figure 77, where the base of the model will need to be printed first. In the case of voids and gaps, it is likely that extra material will have to be built between each if route inspection algorithms aren't able to connect sections of the model. This would be similar in common 3D printing, where 3D printer software accounts for sections of a digital model and fills in unsupported sections with low-resolution, porous sections of plastic, designed to be broken off post-printing.

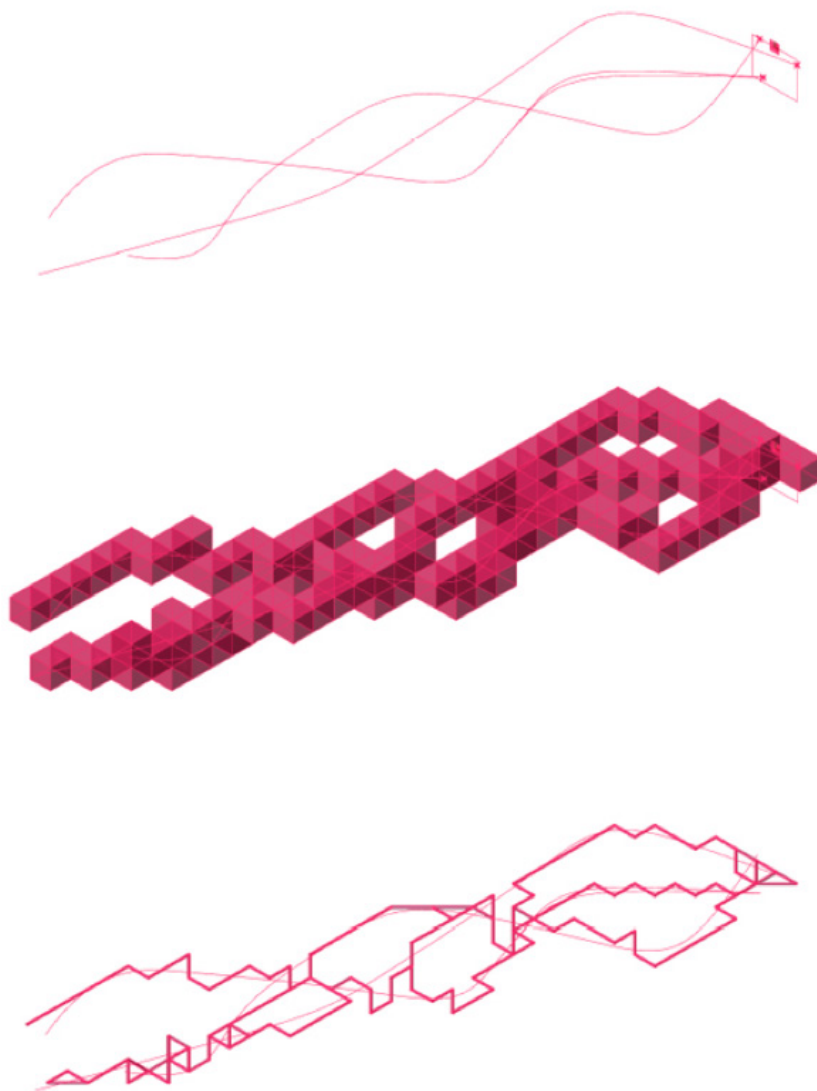
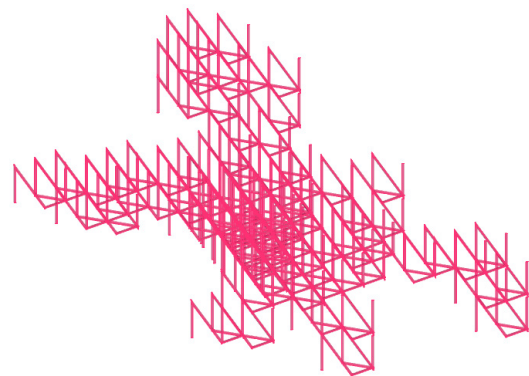
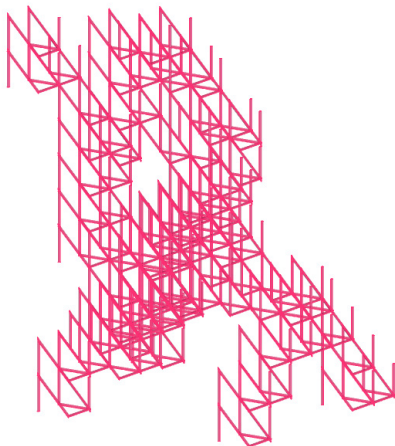
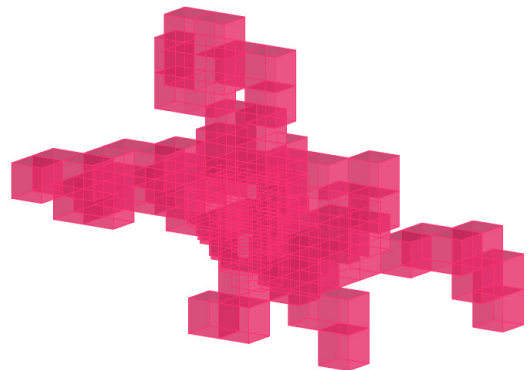
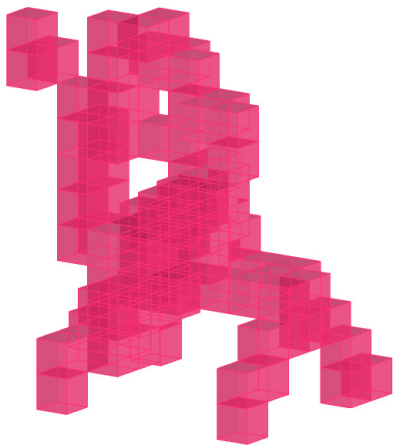
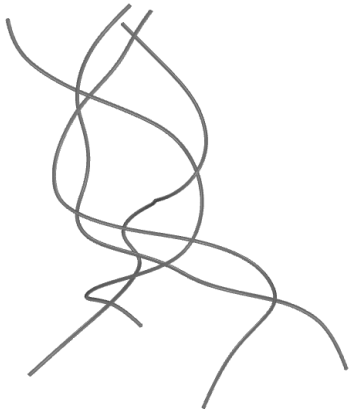


Figure 77. Agent models translates into voxel compositions and print network

Figure 78. Medium scale examples of voxels translated to print models



4.08

VOXEL SIMULATIONS

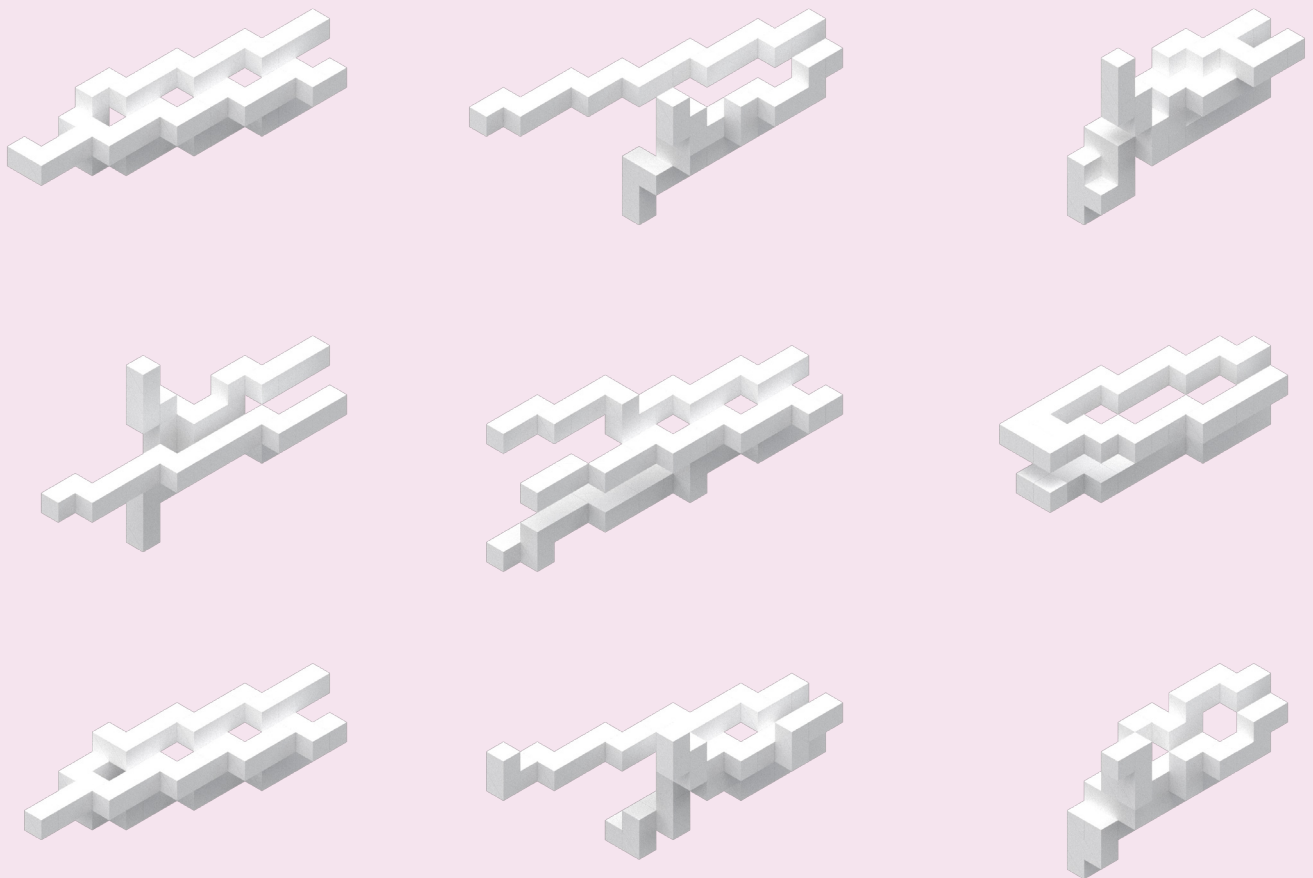


Figure 79. Voxels of Variable swarms

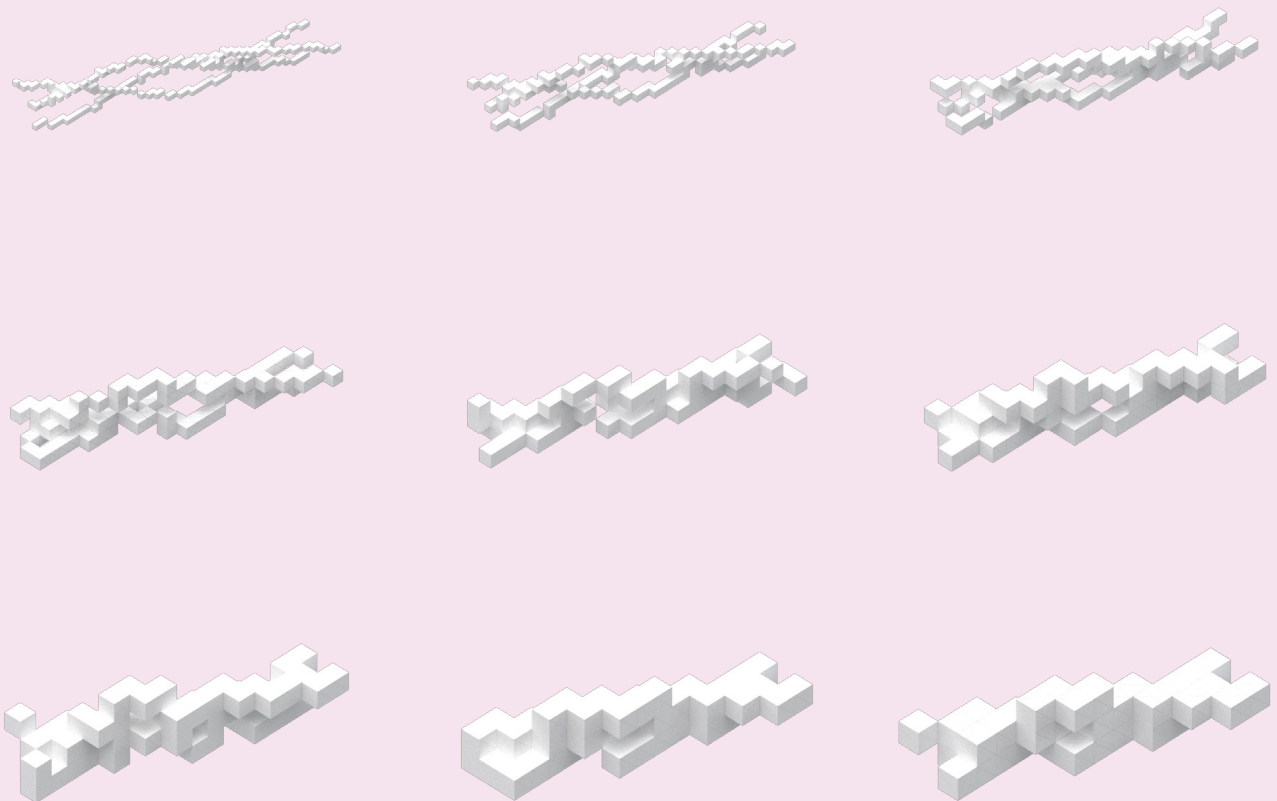


Figure 80. Variable Voxel size

Figure 78 demonstrates the output the Grasshopper cluster builds when given a series of curves. Figure 79 is a series of different curves input into the same cluster with the same voxel size parameter. Figure 80 demonstrates the control the cluster has changing the size of the voxel. This allows for the resolution of the model to be altered - changing the print time and multiple factors of the printing process.

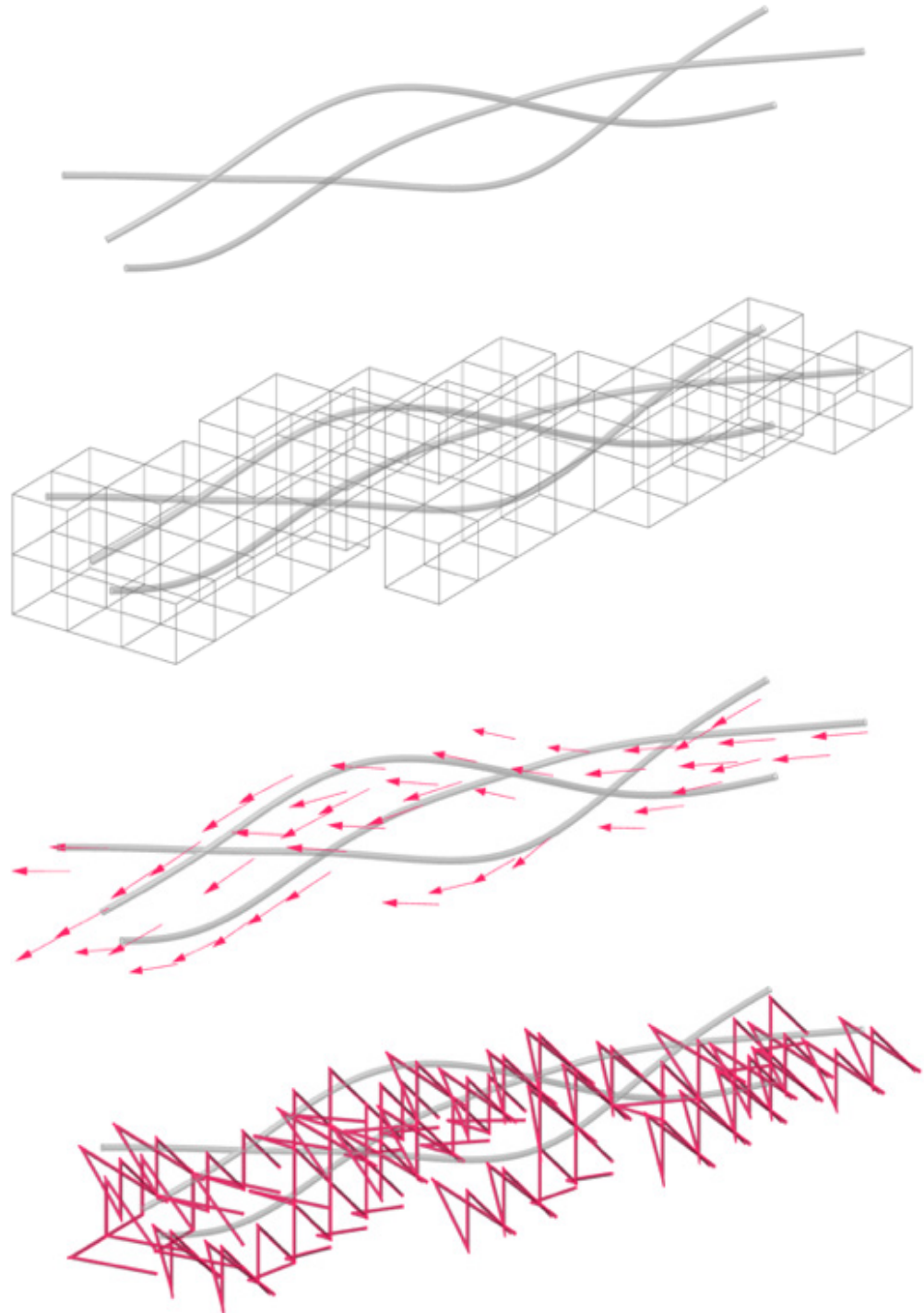


Figure 81. Agent simulated curves translated into vector informed voxel design

4.09

APPLICATION OF SWARM TECTONICS TO VOXEL DESIGN

With a robust system to build a translation of agent-based structures as a voxel system, the next phase of the research focuses on adding detail to the designs informed by the agent-based designs. Each agent-based simulation in this thesis is built on the basis of vectors and points in space, this data is stored and translated into curves for physical application.

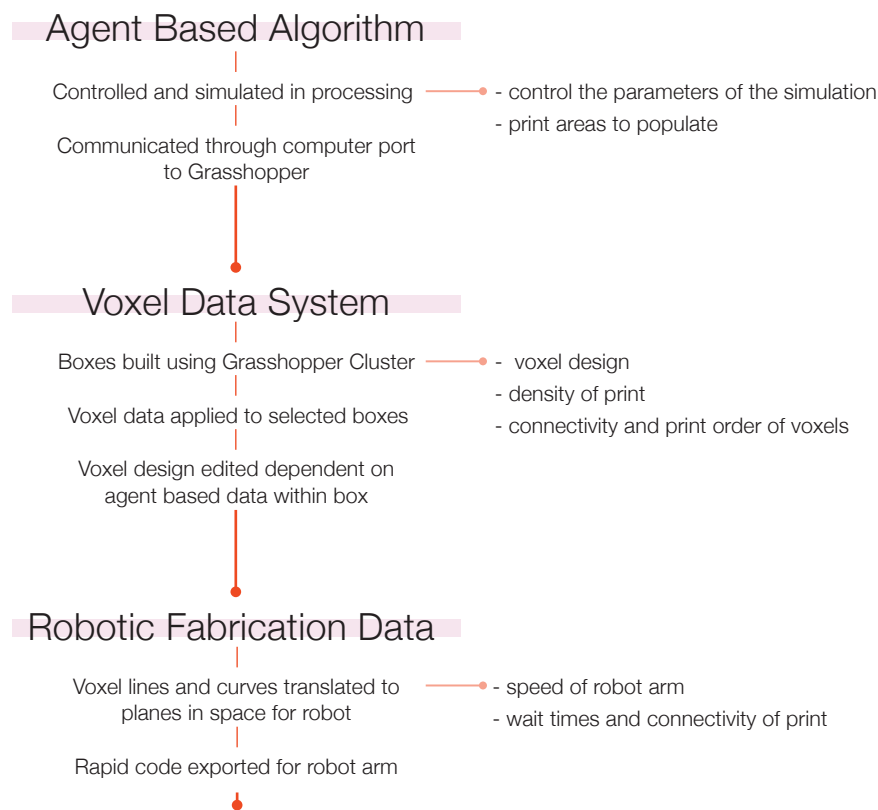
Taking this data from the simulations, it is able to be applied to the design of the individual voxels as parameters. Similarly in the way that vectors were used in Chapter 1: Digital Research to inform the agent paths, the vectors are able to be used to inform the design of the individual voxels. Shown in Figure 81, each box is assigned an overall vector determining the angle of the print diagonal in each voxel.

This translation is purely information being translated for aesthetic reasons, it has no impact on the form finding of the print, instead changes the secondary structure. Using the vectors does however allow the overall agent patterns to emerge.

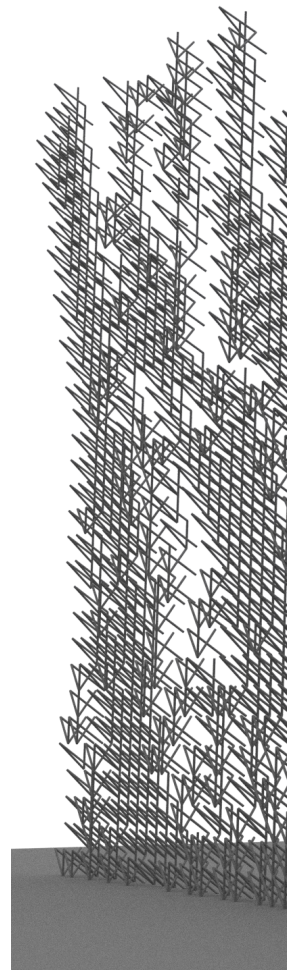
4.10

VECTOR INFORMED 3D PRINT

Figure 82 depicts a model that represents the system that takes the data from the agent-based simulations and uses it to both aesthetically and structurally inform a 3D print model. The diagram below describes the communication between the different algorithms and softwares.



Through this chapter the overall work-flow takes an agent-based algorithm output and builds a digital model with a voxel-based fabrication system ready for 3D printing. The next phase of the research aims to take this same process and apply it using a driven intent. For the purpose of this research aligning with the National Science Challenge it will be environmentally based in the form of a sun shade.



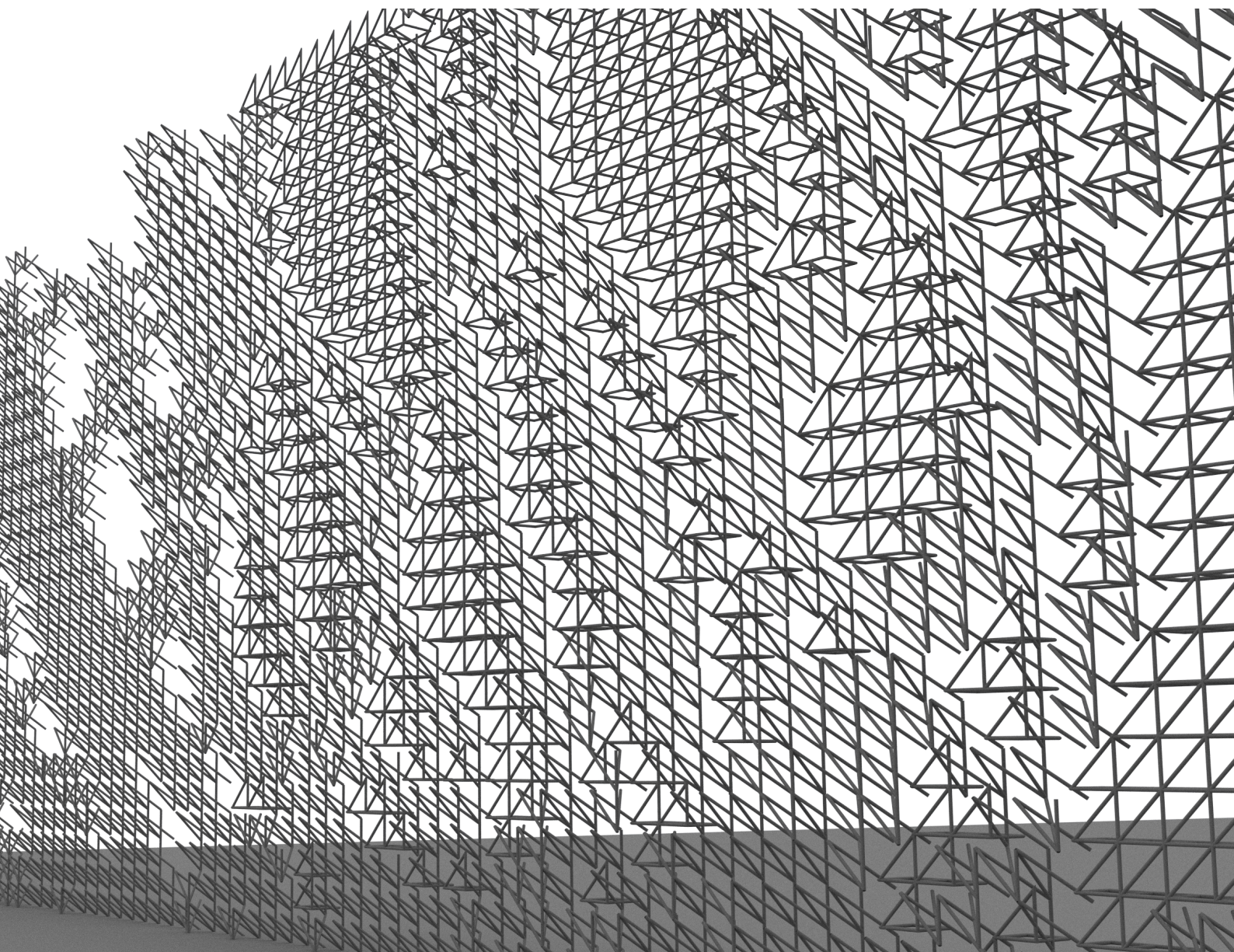
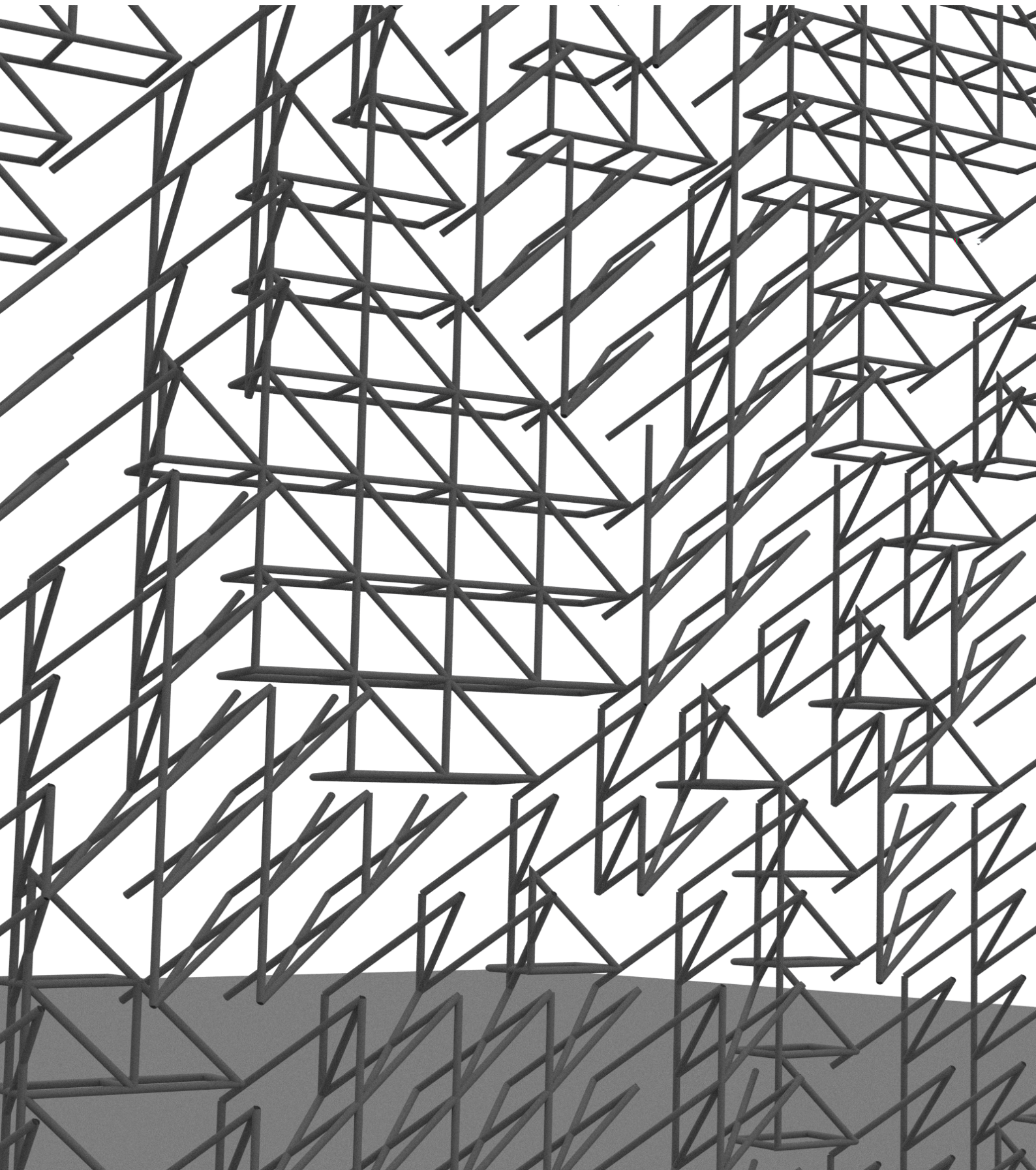


Figure 82. Large-scale 3D print model



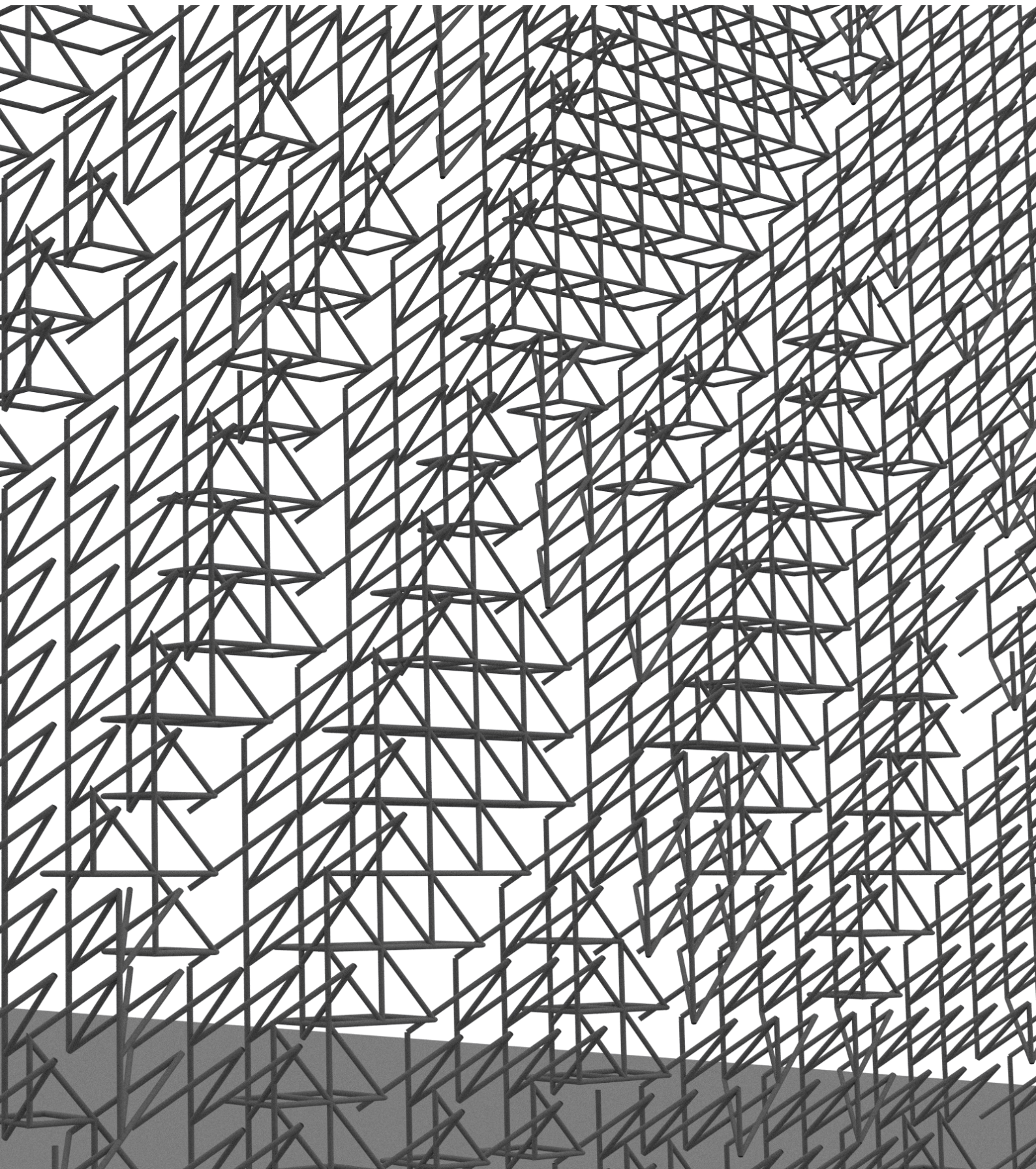


Figure 83. *Large-scale 3D print model*

CHAPTER 5:

DIGITAL APPLICATION 2.0

/ Biological Algorithms for Digital Manufacture

Analysis of a potential space needing
daylight shade // areas where shade is
required //

Voxel design dependent on level of
shade need // density studies and
analysis of model //

5.01

INTRODUCTION

This phase of the research focuses on reintroducing the environmental aspect of sun-shading. As with Chapter 2: Digital Application, this phase is aligned with the National Science Challenge. The aim is to use environmentally informed design as a test case as how these agent-based systems can both simulate form and a fabrication system. The following pages depict both an agent-based environmental control and analysis of a voxel-based system as a sun shade.

The output for this section will be a series of digital models for the NSC, that are each optimised and able to be 3D printed using the extruder. Taking the same process that the previous section did, but integrating another layer of information.

Agent Based Algorithm

Controlled and simulated in processing

Communicated through computer port
to Grasshopper

- control the parameters of the simulation
- print areas to populate

Control of agent structure
based on architectural
shading techniques

Voxel Data System

Boxes built using Grasshopper Cluster

Voxel data applied to selected boxes

Voxel design edited dependent on
agent based data within box

- voxel design
- density of print
- connectivity and print order of voxels

Robotic Fabrication Data

Voxel lines and curves translated to
planes in space for robot

Rapid code exported for robot arm

- speed of robot arm
- wait times and connectivity of print

Digital Model to
be fabricated with
environmental design
intent integrated

SUN SHADING & SIMULATION TECHNIQUES

When simulating agent-based systems with architectural intent it is key to first outline the techniques to apply to the simulation, and the method to implement them. The images below depict a series of techniques to apply to the following simulations of potential sun shades.

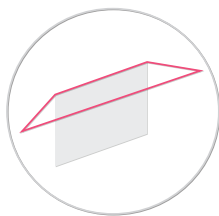


Figure 84.
Use shading designer to determine area

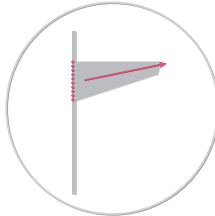


Figure 85.
Use design intent with agent-based systems to propel agents to areas needed

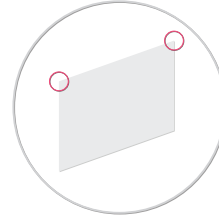


Figure 86.
Have variable start points to ensure form is spread across entire window area

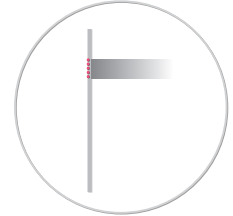


Figure 87.
Increase separation of agents further away from window to intensify shade closer to window

As with the previous agent-based simulations, a series of testing and manipulation of the input data is required to optimise and understand the output, Figure 88 shows an initial series of agent simulations from a single starting point populating a sunshade.

The simulations have various parameters to modify that impact the form, these have all been tested through the digital research.

The core of these simulations is a loop function, where a certain amount of iterations are run before the simulation is terminated. The amount of loops in a simulation is able to act as a digital gate, in the simulations shown in Figure 89, each was run with a series of digital gates that altered the control vector and areas to populate.

A digital gate added a level of control, and is able to be input into each of the parameters that are tested in this research. A digital gate controls the data that is being fed into the simulation. After a certain amount of iterations, the data changes, giving the agents a new vector informing their movement. The gate gives control to the user, dedicating a certain percentage to a specific area or movement.

LOOP COUNT STEERING VECTOR OF AGENTS

Iteration 0	Initial Vector
Iteration 9	Steering Vector to Point 1
Iteration 19	Steering Vector to Point 2
Iteration 29	Steering Vector to Point 3
Iteration 39	Steering Vector to Point 4
Iteration 49	Simulation Terminated

Amount of iterations - 50

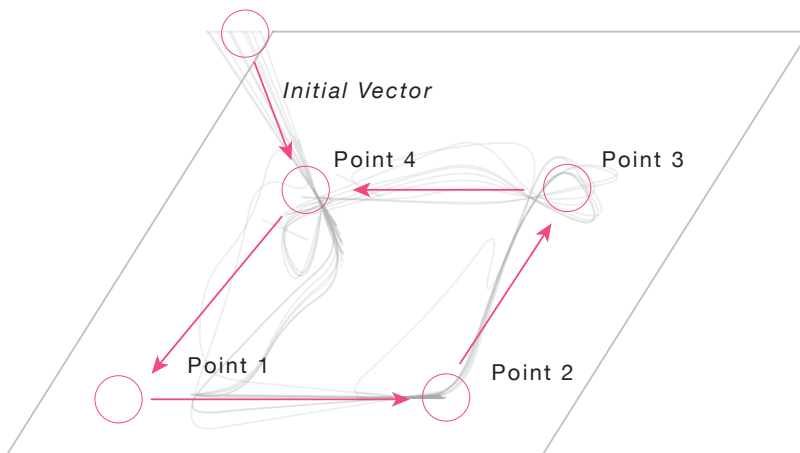
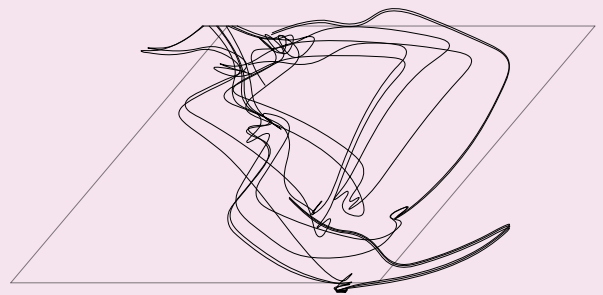
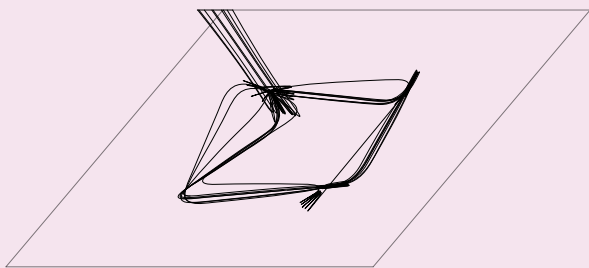
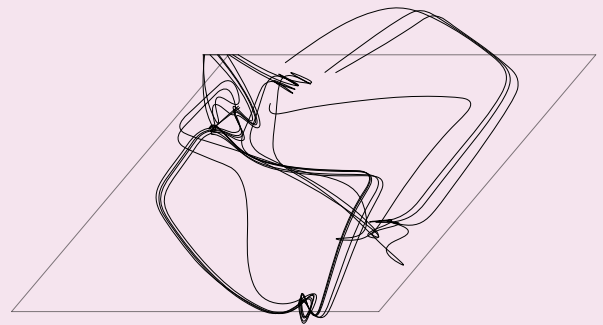
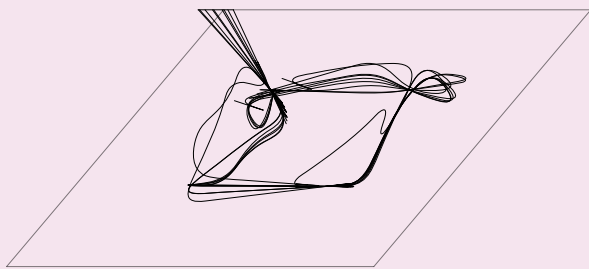
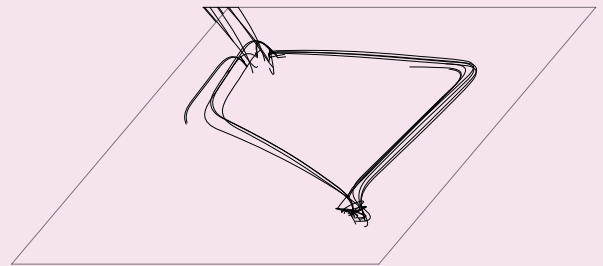
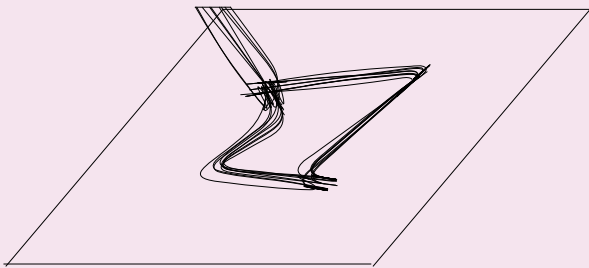


Figure 88.



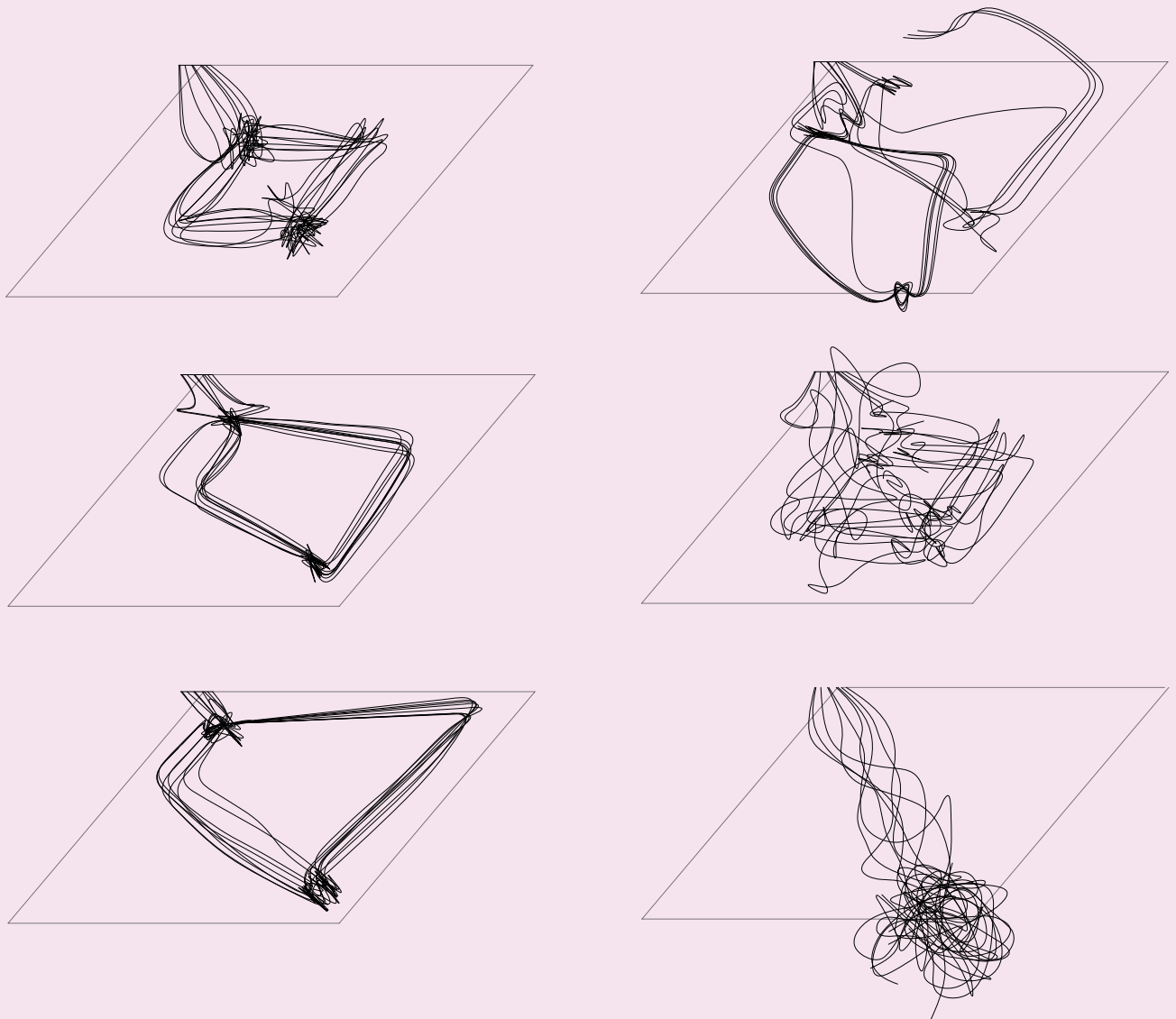


Figure 89. Series of simulations run from section 5.02

SWARM TESTS

Further experimentation using agent-based simulations and Ladybug's Sun Shade designer produced a range of results. The images shown below were produced by altering a number of parameters. A portion of the simulations were run with agents purely interacting with each other, and the rest of the simulations the agents had an initial vector. This vector has been designed to direct the agents around the sun shade area.



Direct | Adhere - 0.06
Repulse - 0.34 | Length - 33m **40**



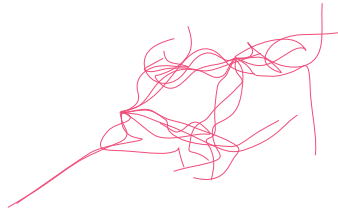
Aligned | Adhere - 0.06
Repulse - 0.34 | Length - 27m **40**



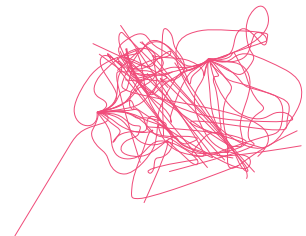
Aligned | Adhere - 0.06
Repulse - 0.34 | Length - 55m **80**



Direct | Adhere - 0.06
Repulse - 0.34 | Length - 102m **80**



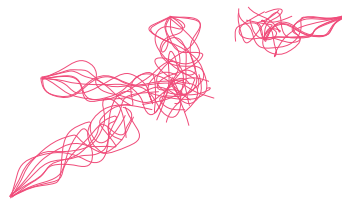
Direct | Adhere - 0.06
Repulse - 0.34 | Length - 13m **20**



Direct | Adhere - 0.06
Repulse - 0.34 | Length - 43m **40**



Aligned | Adhere - 0.06
Repulse - 0.34 | Length - 27m **40**



Direct | Adhere - 0.27
Repulse - 0.34 | Length - 27m **40**



Direct Increase | Adhere - 0.27
Repulse - 0.15 | Length - 28m **40**



Direct Increase | Adhere - 0.27
Repulse - 0.15 | Length - 30m **40**



Direct Decrease | Adhere - 0.27
Repulse - 0.15 | Length - 24m **40**



Direct Increase | Adhere - 0.27
Repulse - 0.15 | Length - 57m **80**

Four control starting points (all others two) -



Type of Initial Vector | Adhere Strength
Repulse Strength | Length of Material Required XX - Amount of Agents



Direct | Adhere - 0.27
Repulse - 0.3 | Length - 54m 80



Direct Increase | Adhere - 0.27
Repulse - 0.3 | Length - 56m 80



Random Vector Increase | Adhere - 0.27
Repulse - 0.3 | Length - 56m 80



Direct | Adhere - 0.27
Repulse - 0.3 | Length - 14m 20



Direct | Adhere - 0.27
Repulse - 0.7 | Length - 13m 20



Direct | Adhere length - 0.27
Repulse - 0.3 | Length - 13m 20



Align | Direct | Adhere length - 0.27
Repulse - 0.3 | Length - 14m 20



Align Increase | Direct | Adhere length - 0.27
Repulse - 0.3 | Length - 14m 20



Align | Direct | Adhere length - 0.27
Repulse - 0.3 | Length - 14m 20



Align | Direct | Adhere length - 0.07
Repulse - 0.3 | Length - 14m 20



Align | Direct | Adhere length - 0.07
Repulse - 0.16 | Length - 14m 20



Align | Direct | Adhere length - 0.07
Repulse - 0.19 | Length - 14m 20

5.04

SHADING RESULTS

With the previous agent-based simulations in 5.04 the vector driven agents introduced an element of environmental intent. To analyse this a parameter of sunshade cover was recorded. The following results show the percentage of the sunshade area over a single window that has been used throughout both digital application sections.



Experiment 1:
72% vs 33m



Experiment 7:
36% vs 26m



Experiment 2:
72% vs 27m



Experiment 8:
56% vs 27m



Experiment 3:
86% vs 55m



Experiment 9:
52% vs 28m



Experiment 4:
95% vs 102m



Experiment 10:
61% vs 30m



Experiment 5:
36% vs 13m



Experiment 11:
30% vs 24m



Experiment 6:
63% vs 43m



Experiment 12:
78% vs 57m

The potential length of printing material has been recorded. This can be used to gauge the amount of material that could potentially be used. Ideally a large coverage of the sunshade area without having the simulation run for too long will produce too much geometry.



Experiment 13:
82% vs 54m



Experiment 19:
20% vs 33m



Experiment 14:
89% vs 56m



Experiment 20:
52% vs 14m



Experiment 15:
88% vs 56m



Experiment 21:
19% vs 14m



Experiment 16:
32% vs 13m



Experiment 22:
31% vs 14m



Experiment 17:
22% vs 13m



Experiment 23:
40% vs 14m



Experiment 18:
19% vs 13m



Experiment 24:
37% vs 14m

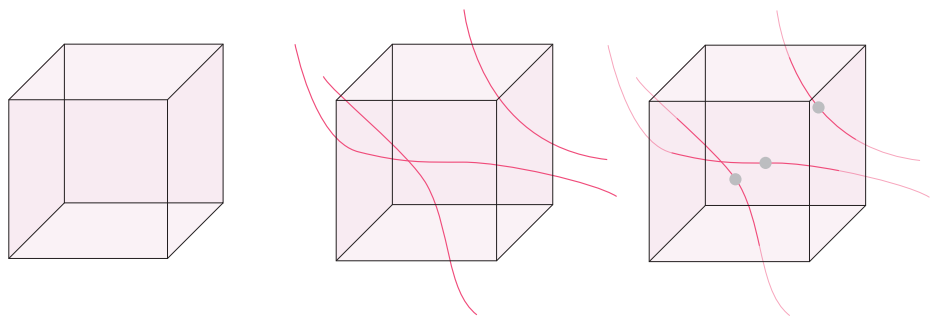


Figure 90 Varying amount of passes through a single voxel

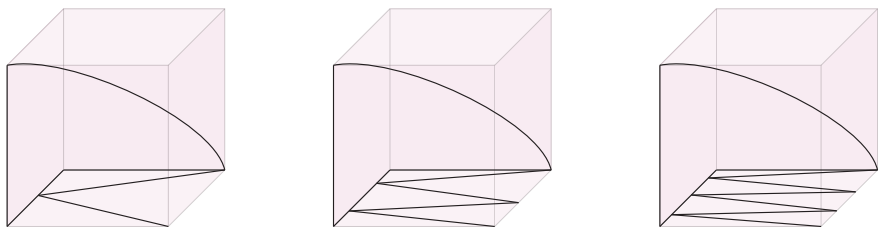


Figure 91 Variable amounts of material

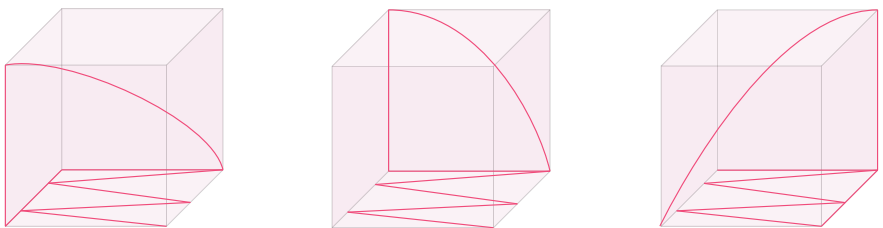


Figure 92 Alternative angles for path movement

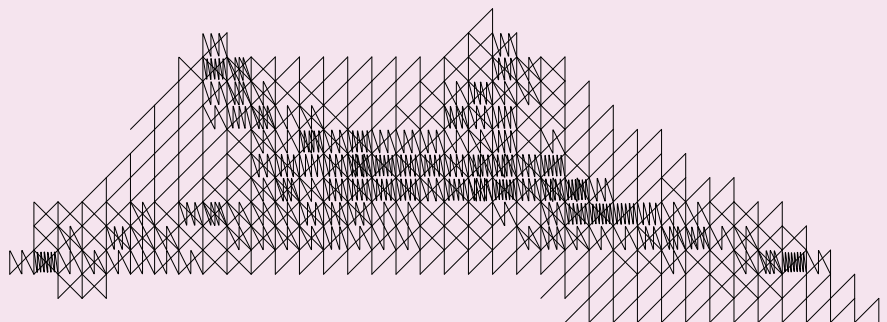
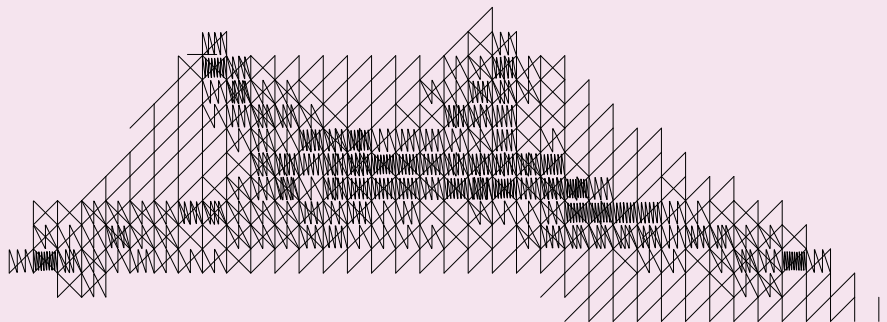
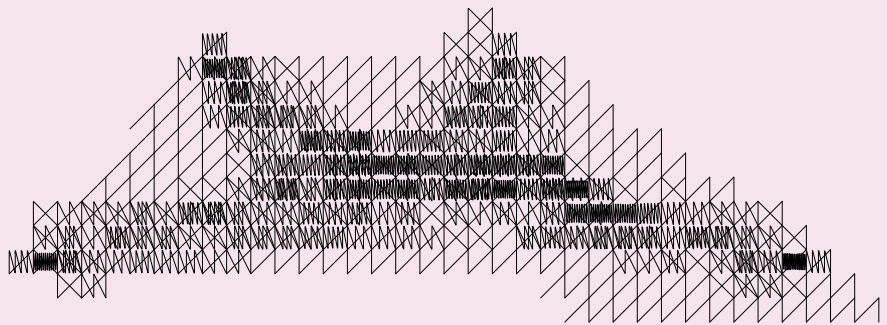
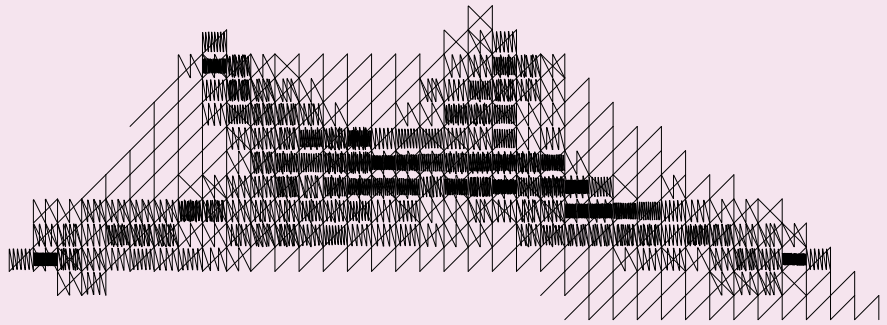
5.05

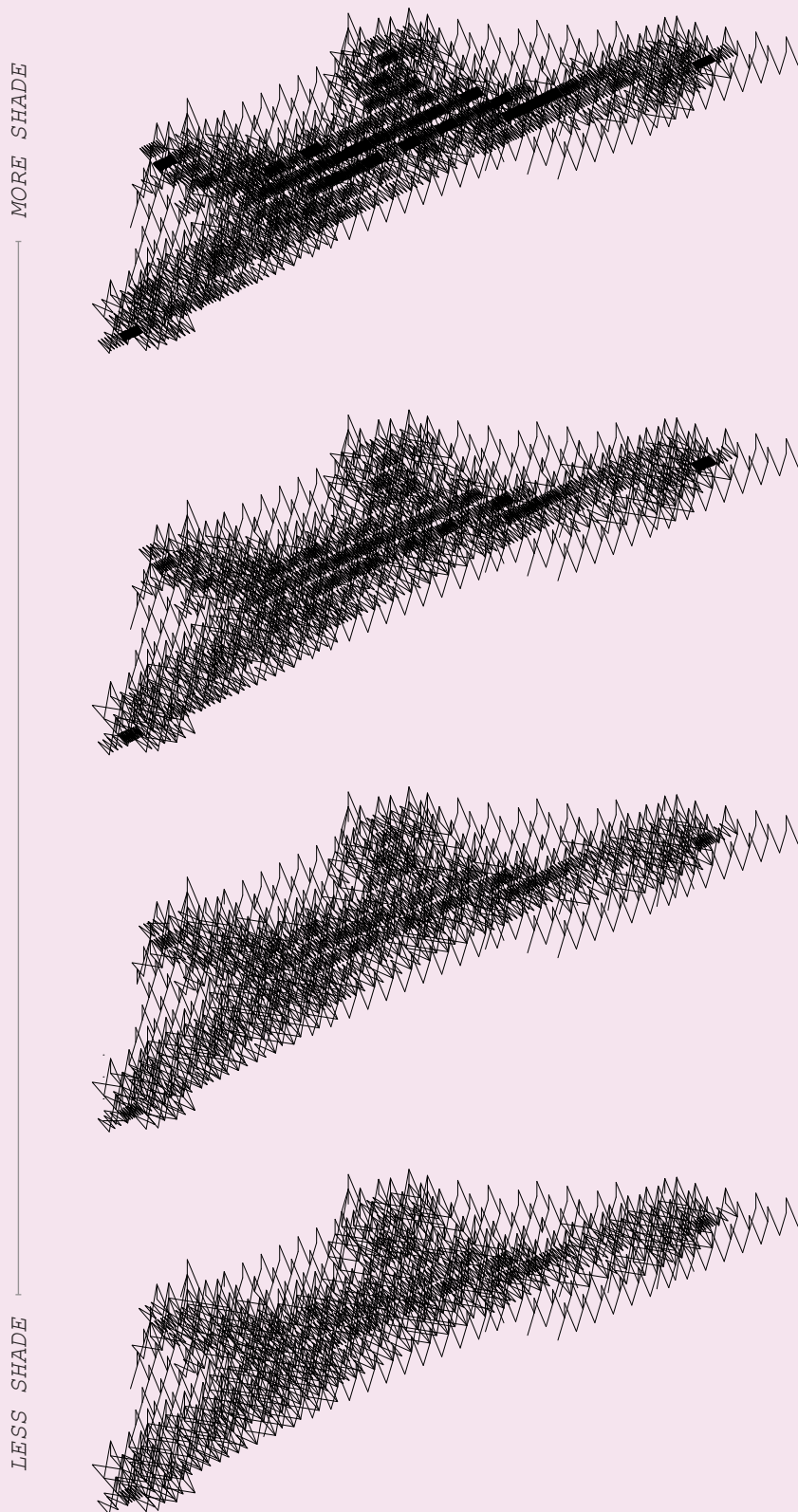
SUN SHADING DESIGN

Similarly to the last section where vectors were used to alter the design of the individual voxels in a specific area, the shading levels are also able to inform this. Taking data from the previous swarm tests populating the sunshade area, this is able to be translated into an integer that can drive the density of the voxel. For this example each of the voxel spaces are tested to determine the amount of agents that pass through each void. The greater amount passes through each void, the greater amount of material that will be printed within that voxel.

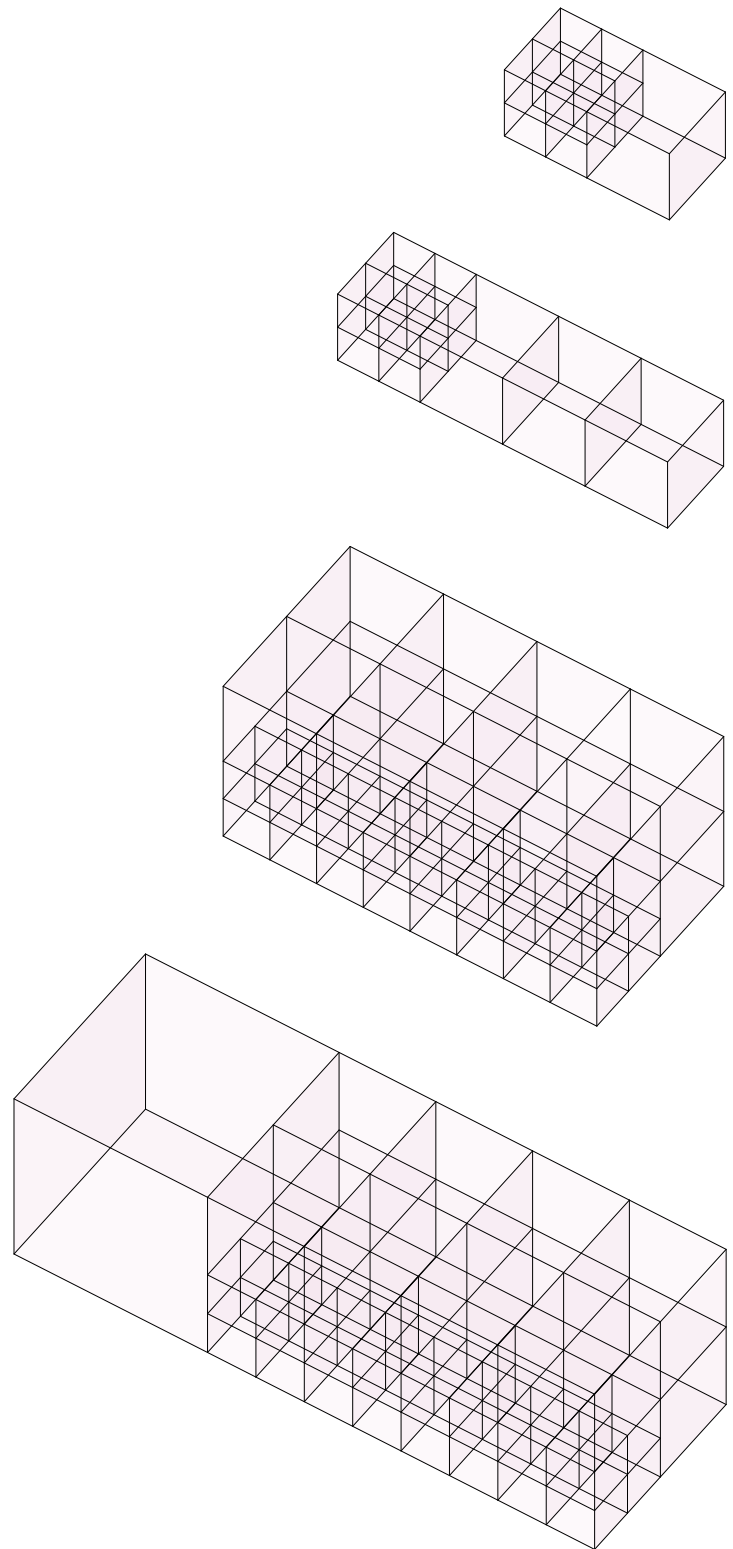
The entry and exit points of the individual voxel remain the same, it is the movements within the voxel that are increased for greater density to achieve shade or sunlight to pass through the space.

Introducing the customisability of each voxel design brings together all the elements in the last section but using an environmental design analysis. This example highlights the adaptability of agent-based systems and how they can be used with different parameters driving the design.





Each of these variations are built using the same sun parameters to populate a desired sun shade area. The entire desired area is filled with a simple voxel, and then the agent simulation in the area (Section 5.03) determines the density and individual design of the voxel. The tolerances with this are able to be manipulated dependent on the amount of sun still needed to be let through, allowing for heat and light to pass through but still acting as a sufficient sun shade.



5.06

DENSITY MANIPULATION

Adding elements of density for sun shading applications is also possible through the design of the voxels themselves. While the voxel shape must remain an equilateral cube, there is potential to have varying size voxels across a model.

The size variation is able to be controlled by the swarm data or the environmental data, and applied in a way that reflects the design intent of the fabricated model. Sections of a sunshade that require extra density would be able to have single voxels split into 8 smaller ones, resulting in more material in that space.

This technique would work in the same way as adding material in the voxel, but give more design variation and uniqueness. There could potentially be issues during the fabrication process as one of the limitations of free-form printing is the effect that gravity has on the print. There is a possibility that the top layers of a series of different sized voxels do not align, this would have to be tested and the desired offset for the amount of voxels adjusted accordingly.

Through the following print testing this technique will have a series of ideal print parameters set to it that will allow it to be inserted at any point in a voxel-based model and be printed successfully.

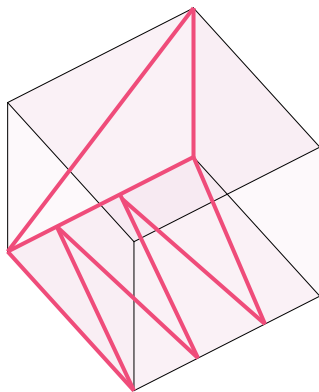


Figure 95. Single voxel design

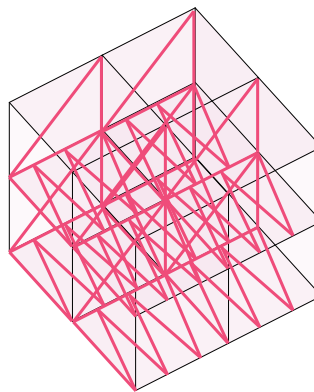


Figure 96. Multiple voxel design

CONCLUSION

Through experiments with digital environmental design and analysis tools, this section has developed a methodology to simulate environmentally driven agent-based simulations. This work-flow runs through a series of programs to produce a speculative design ready for 3D printing using the robotic arm. The overall model is representative of the environmental design which has driven the development of the model.

Figure 98 depicts a series of shading analysis for an outcome of the final methodology. The input shading design focussed on a restricting excessive light into an interior space at in the morning.

The highest level of shade given to the space is at 10am, and is able to provide partial shade across a single day. This analysis is able to form a feedback loop if required, with future simulations focussing on areas without material.

Figure 97 details the process of the digital methodology for this output. Initial environmental data is determined and used as driven design instructions to inform the simulation. Analysis is used to determine if the system is satisfactory, where parameters such as the levels of shade provided and the amount of material required is analysed before the digital model is turned into a series of voxels for fabrication. Multiple programs are used for this with data communication streamlined between each of these.

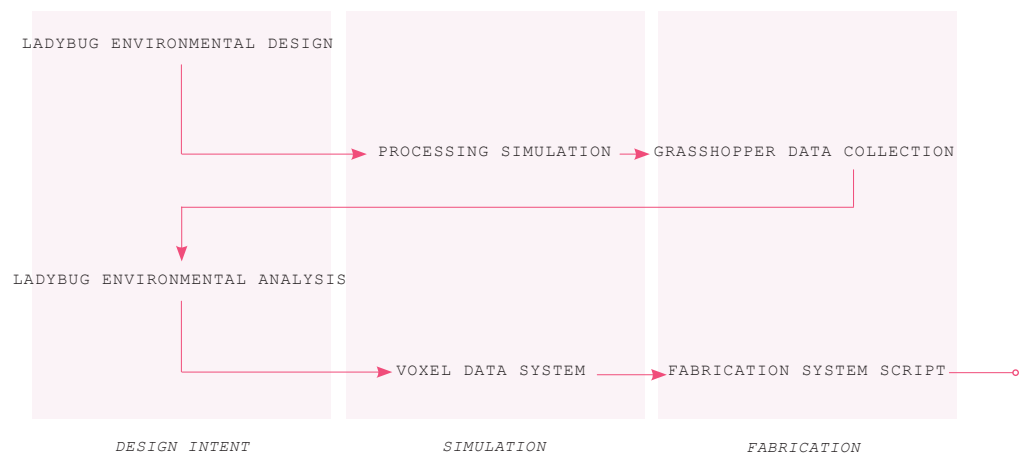


Figure 97. Digital methodology

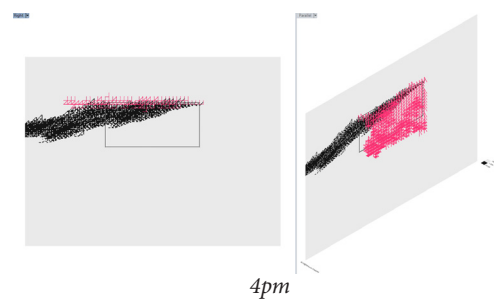
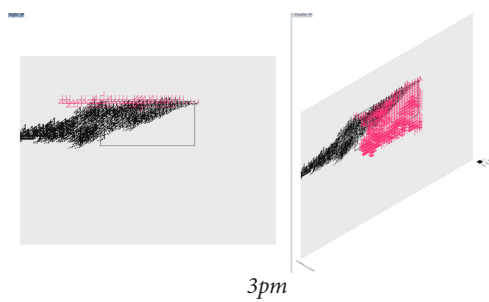
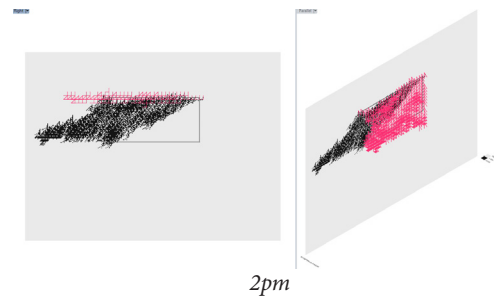
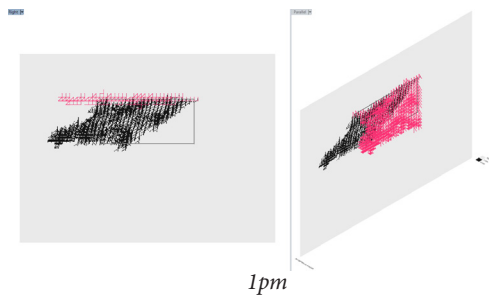
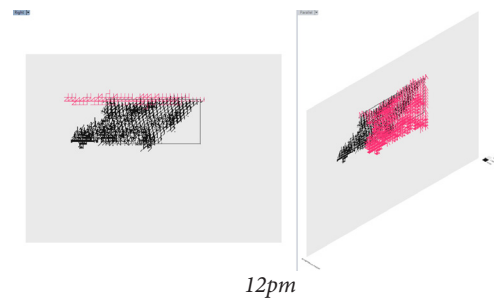
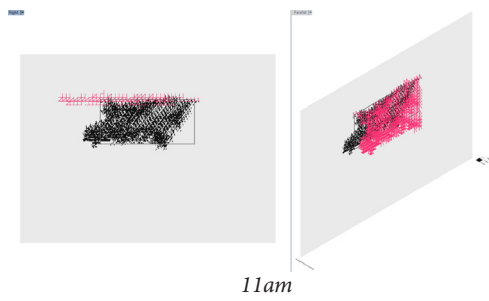
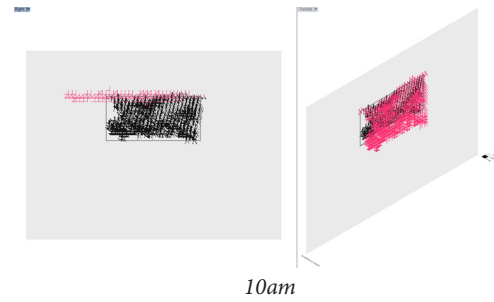
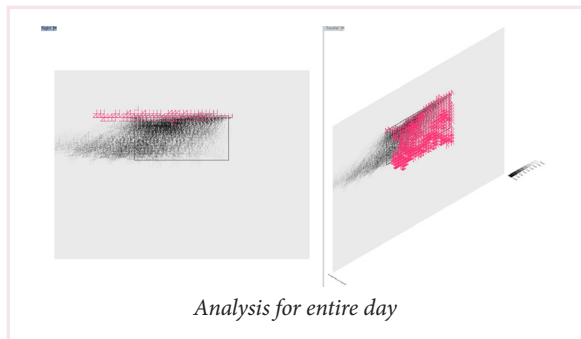
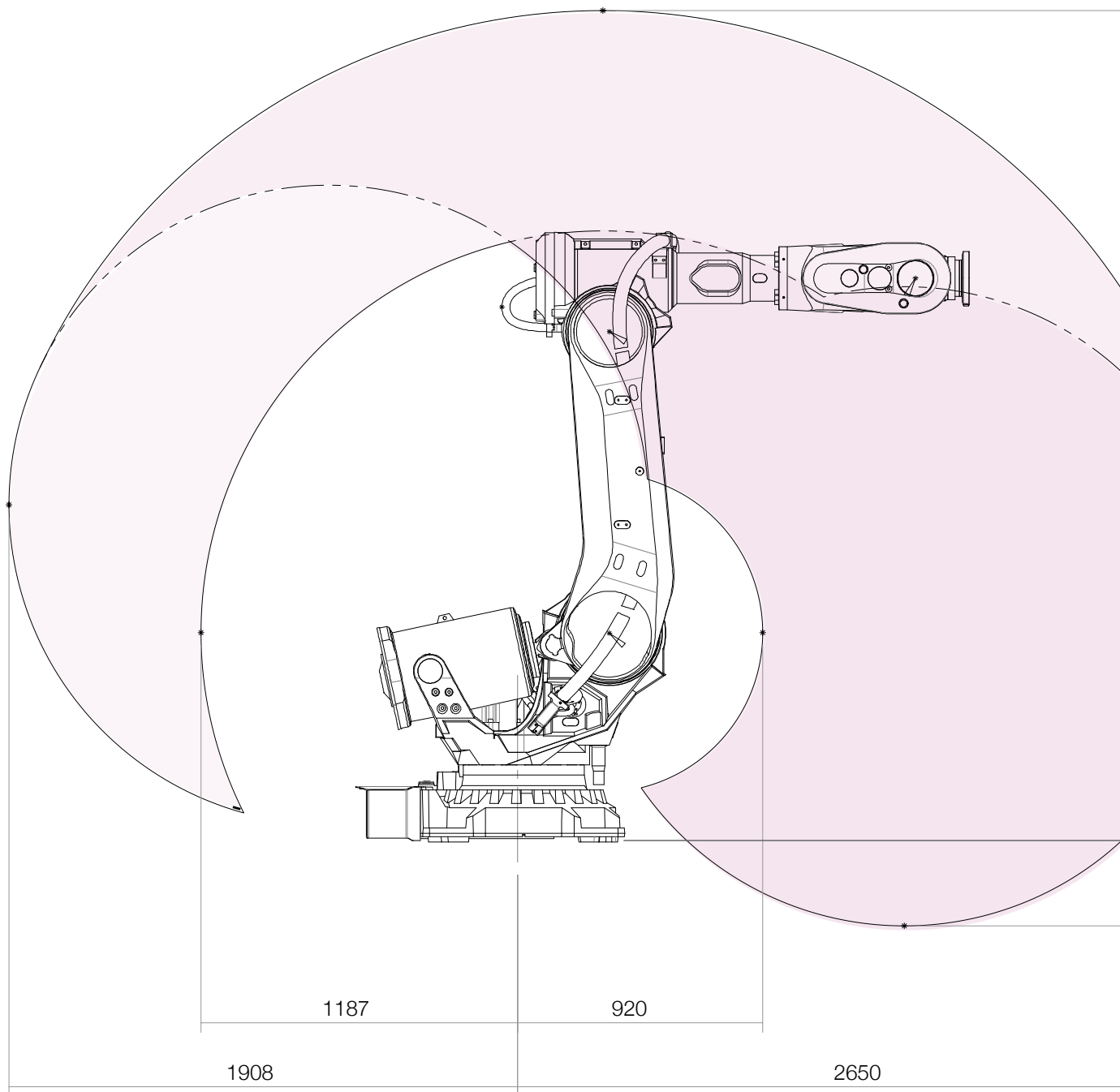


Figure 98. Sun analysis across a single working day

CHAPTER 6:

PROTOTYPE FABRICATION

/ Biological Algorithms for Digital Manufacture



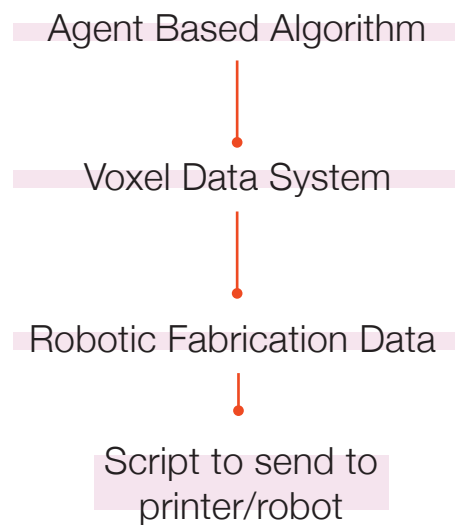
6.01

INTRODUCTION

The final phase of this research brings together each chapter to present both a final digital and physical model. Continuing to use an environmentally informed model as a proof of concept for the application of agent-based simulation, this section will aim to fabricate a scale model of a sunshade.

The sunshade parameters and design elements were developed in the previous chapter, these now need to take into consideration the attributes of the free-form 3D printer.

Taking the results and working print parameters, together these will be modelled into a digital model with print data attached, ready for fabrication.



6.02

USE OF SECOND EXTRUDER

During the final phase of this research the ability to use the second extruder came available. After a series of final developments and hardware improvements, the printer was functioning. The second extruder, which is controlled and be scripted through HAL, had numerous advantages in comparison to the original.

These advantages/differences include:

- ability to stop/start extrusion in Rapid Code
- improved cooling system using compressed air
- variable temperatures

This extruder comes with its own set of disadvantages, the largest one being that this involves the learning of an entire new tool and its controls. Each of which need to be scripted

The use of a voxel-based fabrication model was adapted for the purpose of working with the previous extruder which could only work along a single line. The availability of the second extruder suggests that this was possibly not needed, but this system will still be tested using the new extruder. The change in tool will allow for streamlining in the route inspection process, but does come with its own set of constraints.

The next pages depict the model development using the overall methodology built in this research. The final phases of this methodology will be tested and specified in a series of tests using the second extruder.

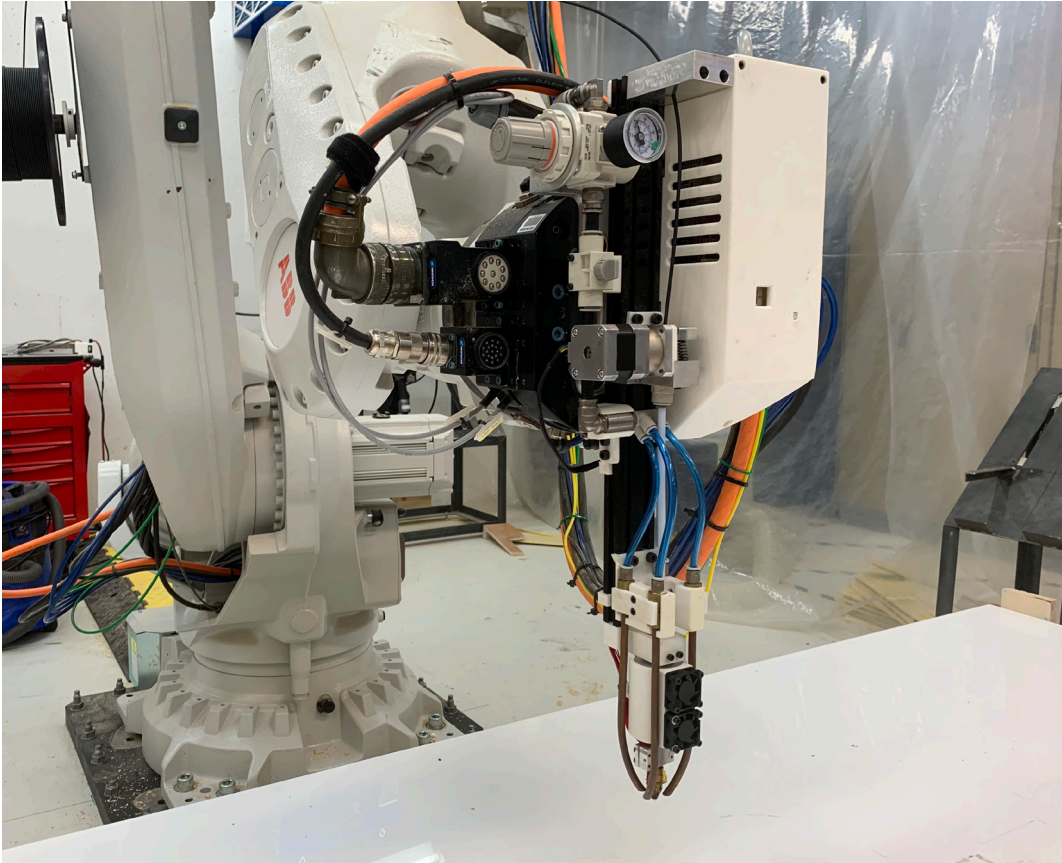


Figure 100. Second Extruder

6.03

MODEL DEVELOPMENT

Regardless of the availability of the second extruder, the first half of the work flow, which is completely digital is not needed to be adapted.

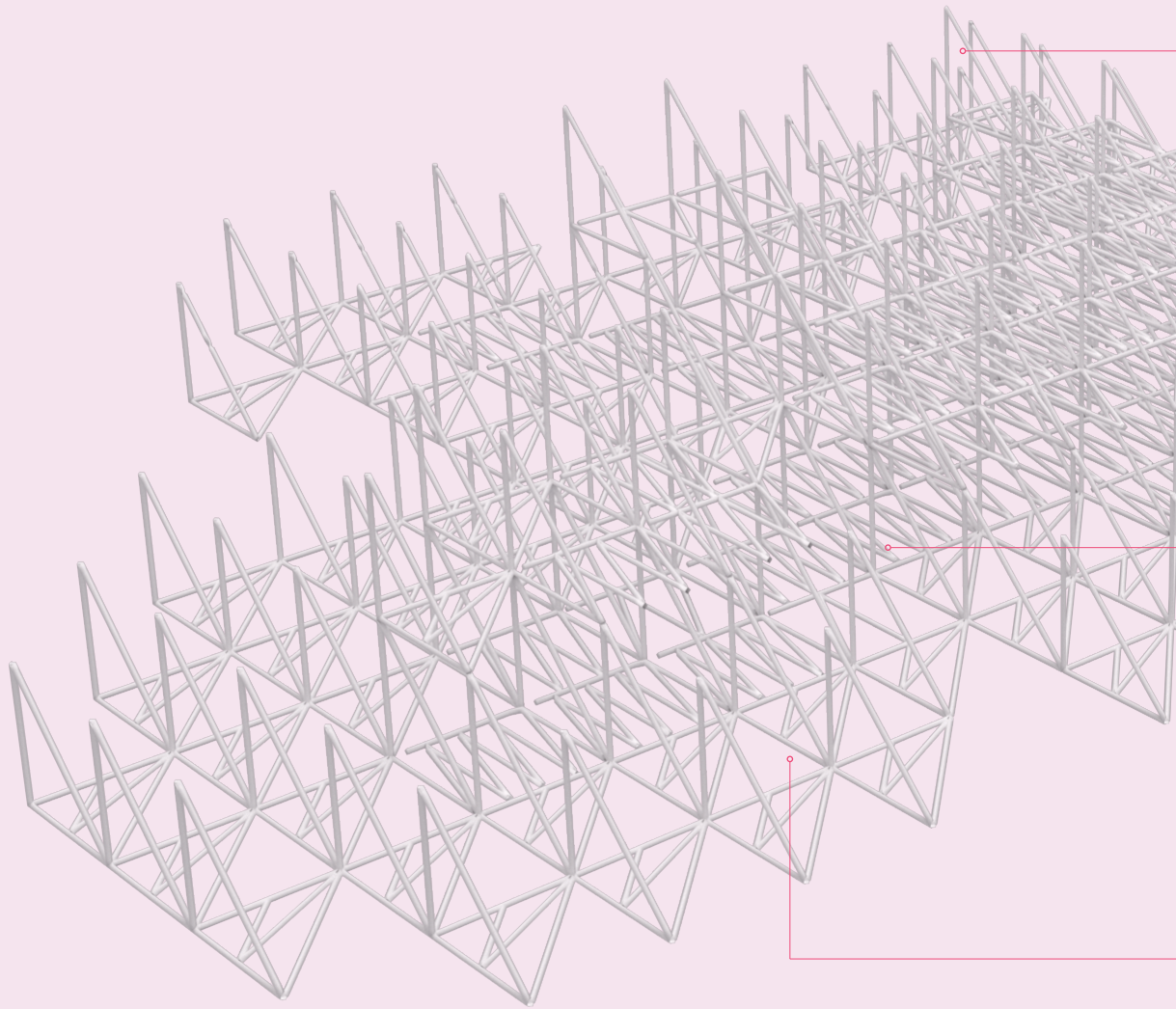
Multiple sections of digital simulations and data transfer are strung together. Shown in Figure 101, these form a continuous flow resulting in a digital model.

Working through these steps, a digital model is output, with a variety of parameters controlling its functionality and aesthetics that have been developed over the course of this research.

Digital Model Work-flow

1.
Environmental design setup
 2.
Environmental design parameters applied
 3.
Agent-based simulation scripted
 4.
Simulation run using environmental constraints
 5.
Output environmentally analysed
 6.
Voxel-based system applied to simulation output
 7.
Individual voxels designed
 8.
Voxel designs applied to simulation
-

6.04 DIGITAL SUNSHADE MODEL



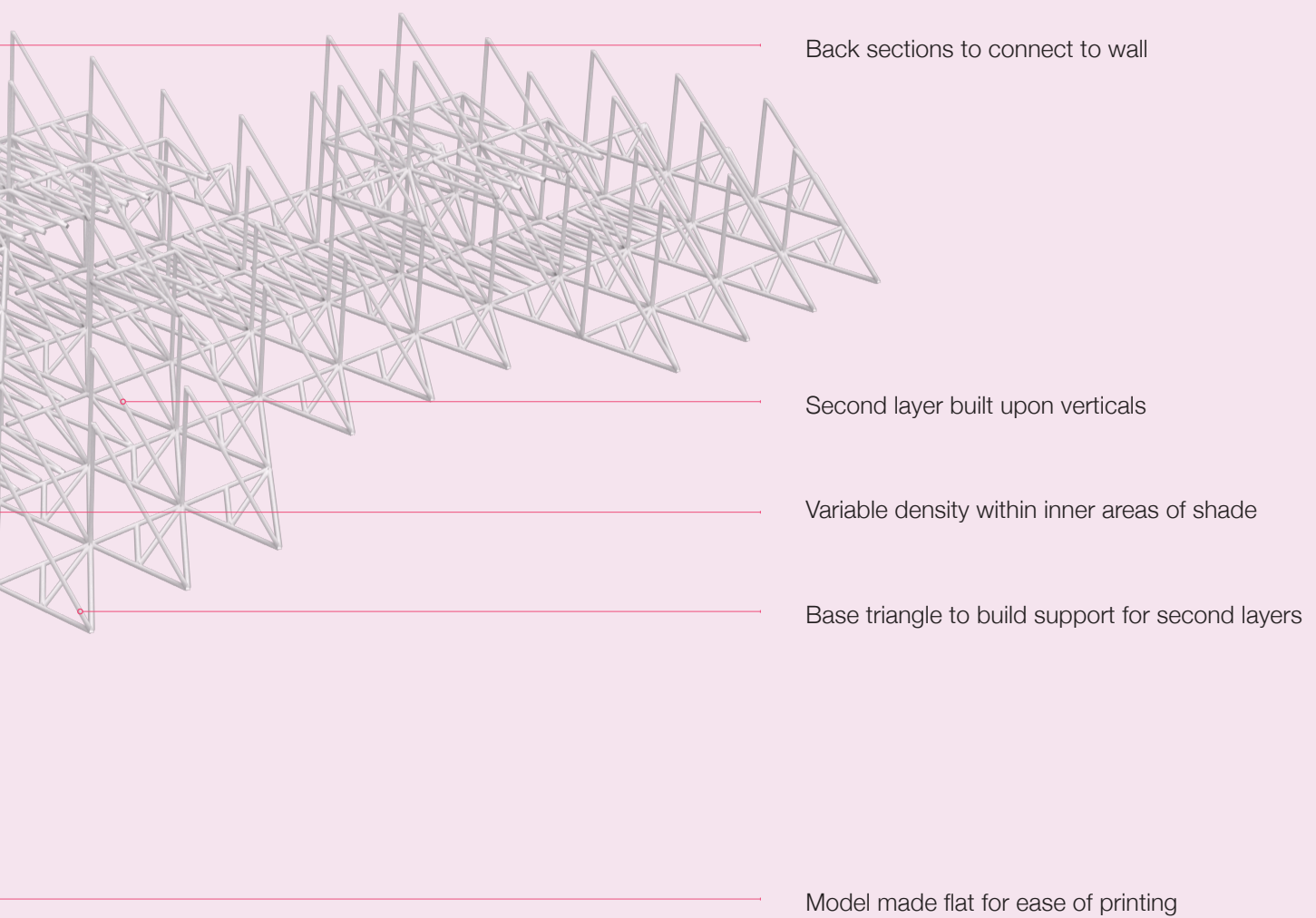


Figure 102. Digital model of sunshade

FABRICATION DEVELOPMENT

The use of the new extruder will greatly impact the work-flow of the fabrication system. The route inspection section will be more straightforward, however the control of the Rapid Code is a new element of this research, and will need to be considered carefully.

The printer is not new in this project, so the troubleshooting and print parameters will be an easier task, but still necessary to perfect.

Overall the fabrication work-flow remains the same as it has for the previous chapters of the research (Figure 103), but with the coding instead being embedded in the script rather than controlled remotely.

The following pages depict the first series of test prints with the new extruder building a sunshade design.

Fabrication Model Work-flow

1.
Model split into series of levels, X, Y, Z
 2.
Route inspection algorithms used to determine
pass through each level
 3.
Script used to adjust plane
 4.
Offsets adjusted for each level
 5.
Wait times input at each turn/junction
 6.
Fabrication simulated for error checking
 7.
Rapid Code exported to robot
-

6.06

EXTRUDER TESTING

To begin to fabricate the digital model it is necessary to derive a set of print parameters that the extruder works best with. Similarly to the first series of print tests, this needs to be undertaken at a base level before building a more complex model.

The second extruder is controlled using Rapid Code, this requires the data that was previously transmitted through the desktop computer to be encoded into the robot routine.

The same data is required to be programmed, but it now has to fit within the script of the robot routine. A series of initial commands are sent to the robot, with other commands able to turn on/off during the print.

INITIAL COMMANDS

- Heating on
- Temperature set
- Extrusion off
- Fan On

COMMANDS WITHIN SCRIPT

- Extrusion on / off
- Wait times
- Printer speeds

Through a series of test prints these will be resolved with a series of parameters that are capable of printing a medium to large scale model.

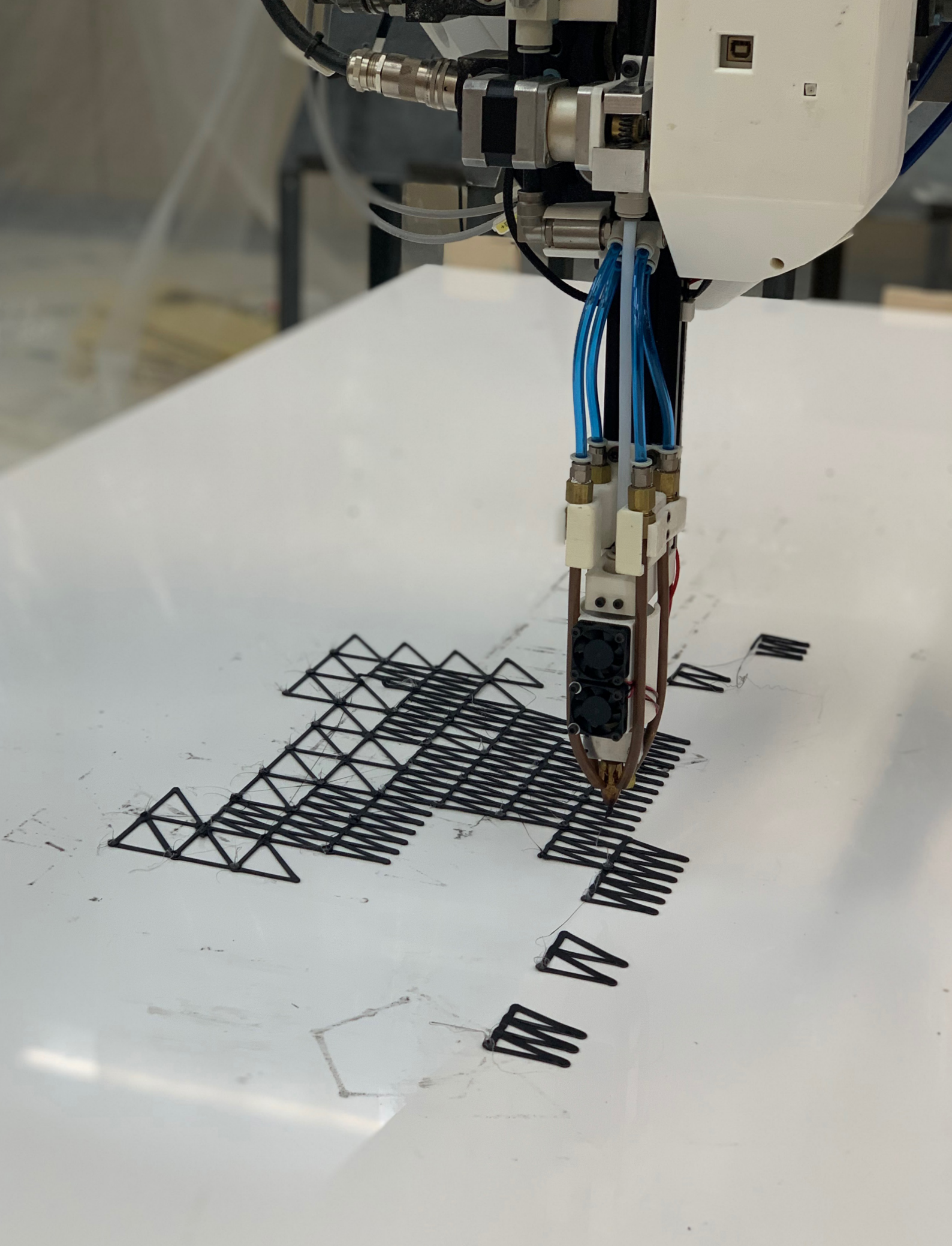
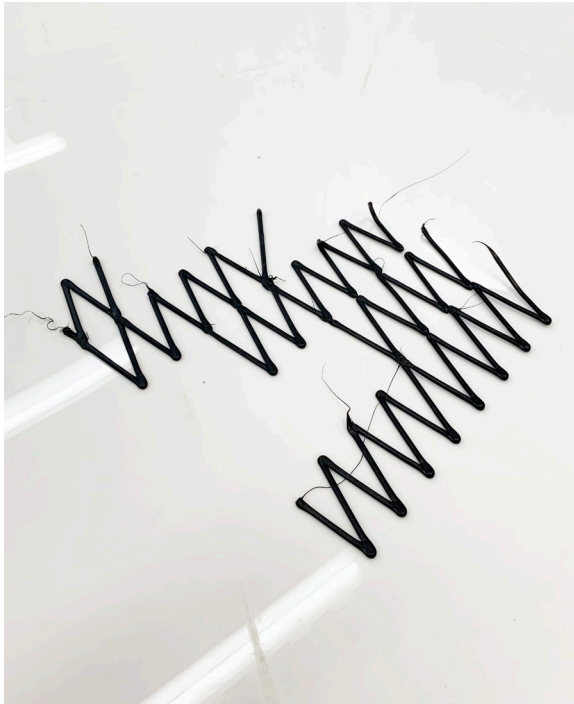


Figure 104. Printing of test 5.04

5.01



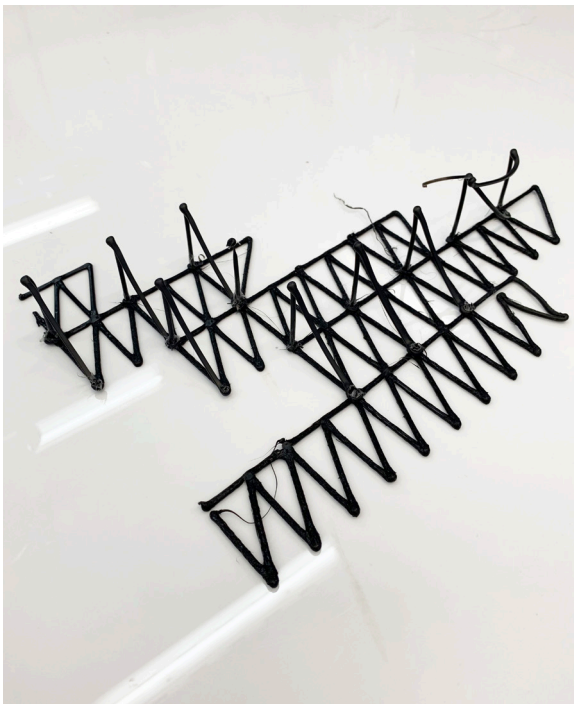
intention: Test the capabilities of printer turning on/off and order of script to achieve a routine where print doesn't collide.

outcome: rough print

reflection: The ordering of the boxes in the Grasshopper script were based of the order of the swarm, so they were ordered from the outside in. This was not a problem at a base level but would cause issues when verticals are built as the extruder would collide with previous print sections.

To avoid this, the boxes were first ordered in the X, then Y, then Z direction to ensure that the print would build in a set order.

5.02



intention: print a second layer atop foundations

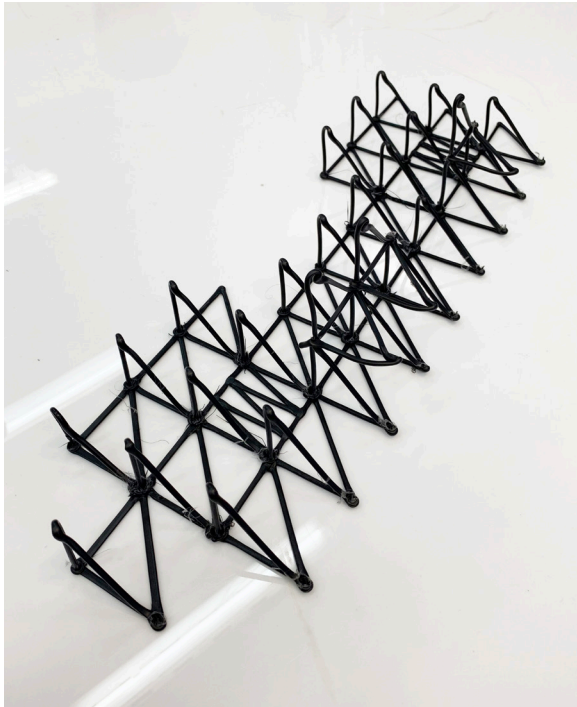
outcome: fail

reflection: Previous print tests have been in a strict order or the print has collided with itself. Despite the re-ordering of the boxes, a small number of verticals were found to interrupt the robot at later points in the print.

On inspection of the script, it was found that through the density manipulation, the order of the print was edited to allow for a number of voxels to include more material.

From this error, it is clear that data management and ordering of parts of the print needs to be concise.

5.03



intention: print a second layer atop foundations

outcome: minor errors

reflection: With the adjustments to the print voxel order for a second time the print successfully finished.

The shading structure itself was much larger in the digital model but was sectioned down to test the printer without having too long a print time.

The second layer was only built in three sections, each of these was on the end of the voxels, so they didn't have four points supporting them however they still managed to stay upright. There were differences between the two layers, this was resolved by using a 2.5mm offset, accounting for the vertical tolerances caused by sagging of the print.

The next phase is to test a third and more layers on top of the print at a larger scale.

5.04



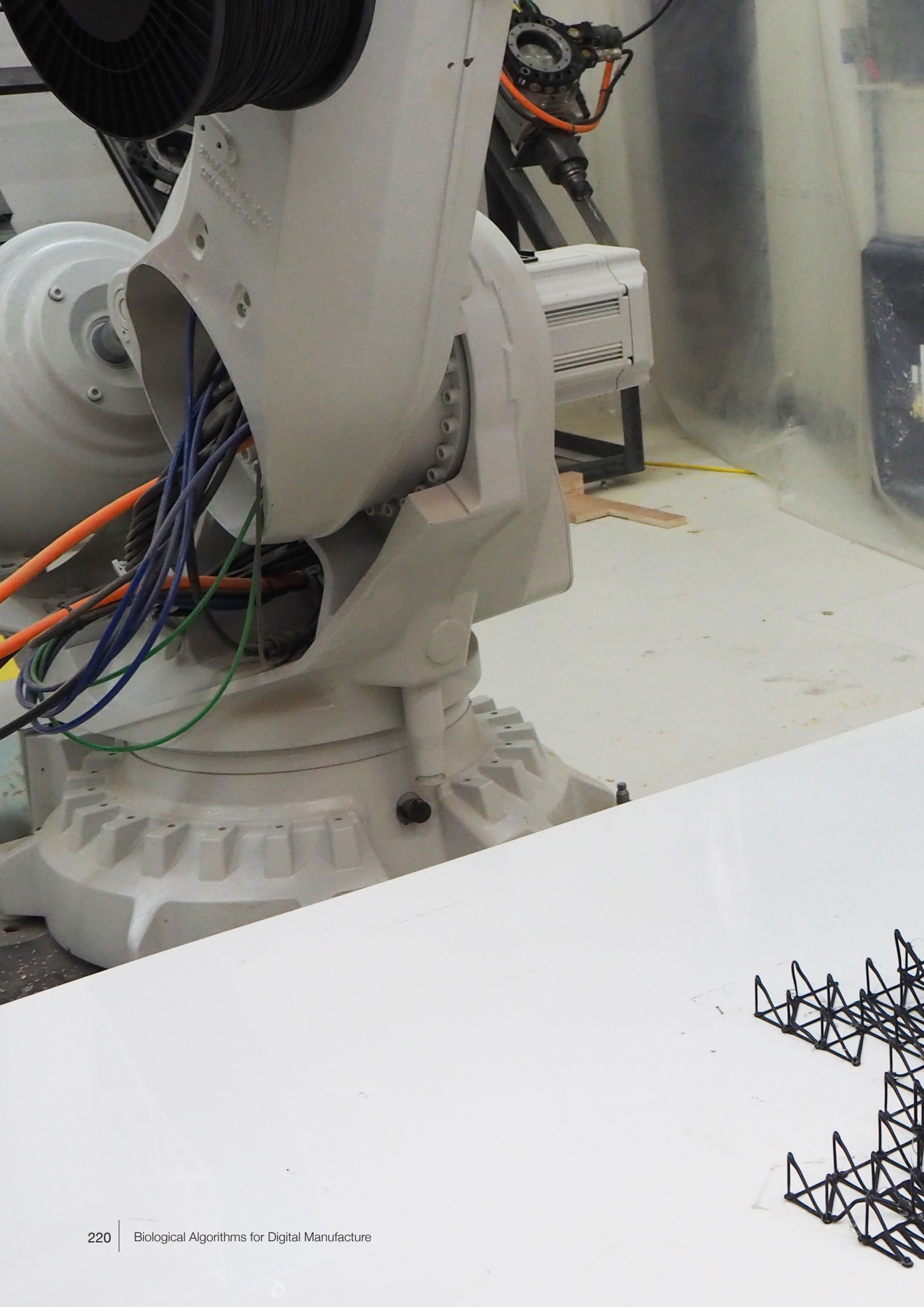
intention: print large scale model of sunshade

outcome: printed

reflection: With the previous print (5.03) printing correctly, the aim was to scale this up in both the X direction and Z. One of the draw-cards for switching to a voxel-based system was the scalability. This print aimed to test this theory.

This print was successful and printed with the same script as 5.03. There were minor issues with areas not being supported, changes could be made to the script to build extra supports where there is none to avoid this happening.

The far right section of this model did not print due to the table being uneven. The plastic was unable to adhere to the ground. This did not affect the remainder of the model however.



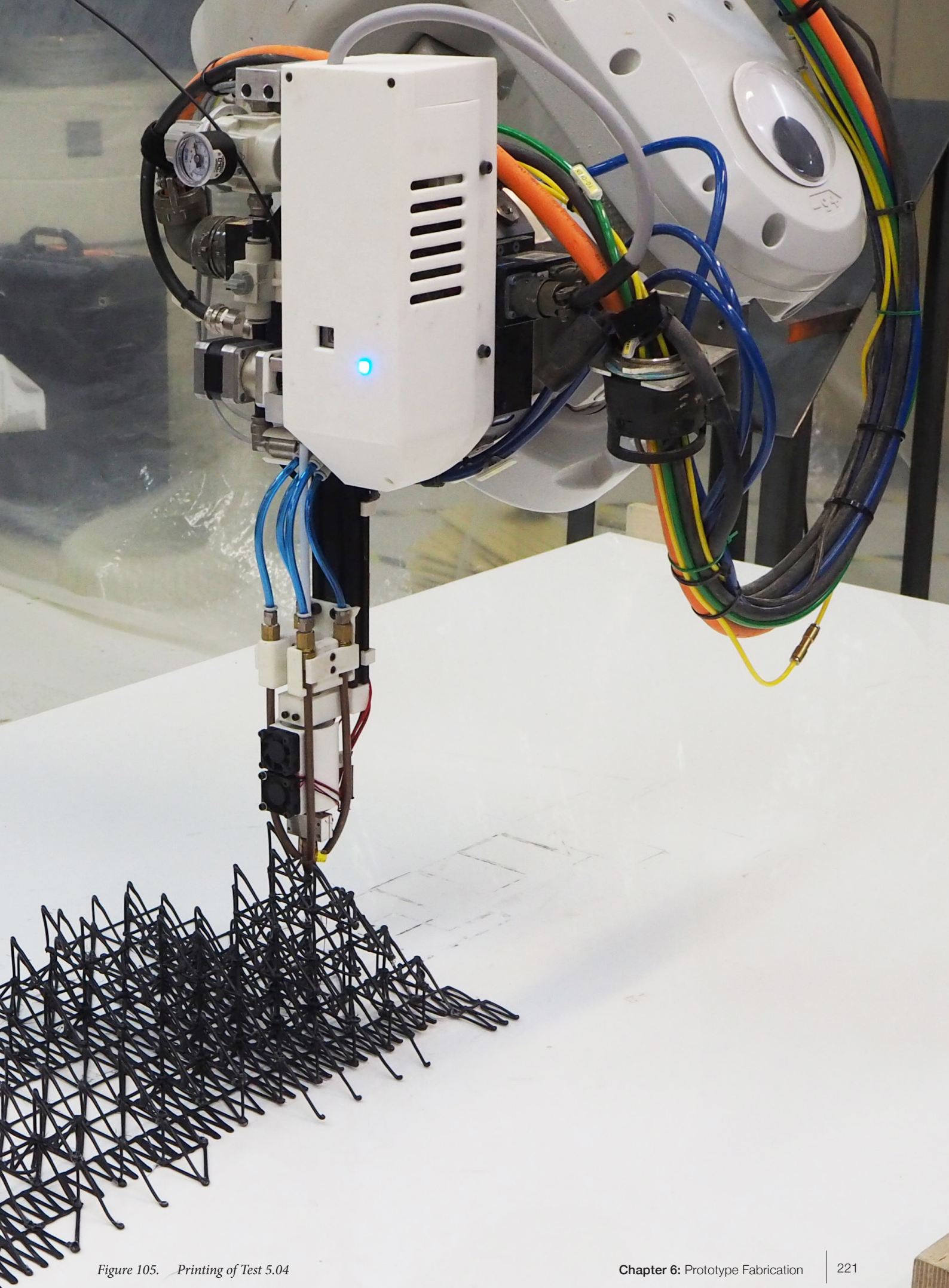


Figure 105. *Printing of Test 5.04*

6.07

PRINT ERRORS AND RESOLUTIONS

ERROR

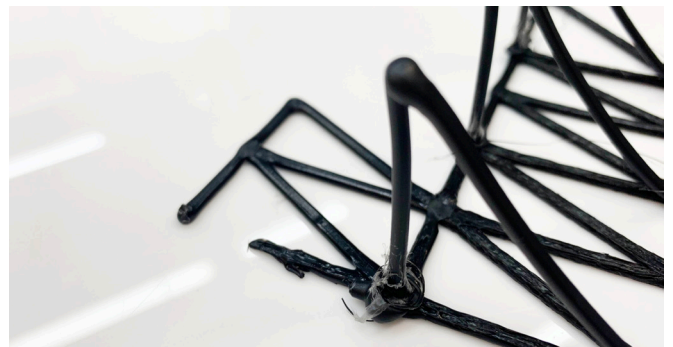
Thin vs thick prints

Sections of the print would on some occasions print very thick and bubbly, as if there was air within the filament. These sections had a very rough outside.



Wait times incorrect

At points in the print the printer would not print the entirety of the voxel - due to the fact the filament extrusion was not on early enough.



String filament

With the extruder moving around the base plane without filament extruding, the remains were dragged between points, leaving a series of strings across the print.

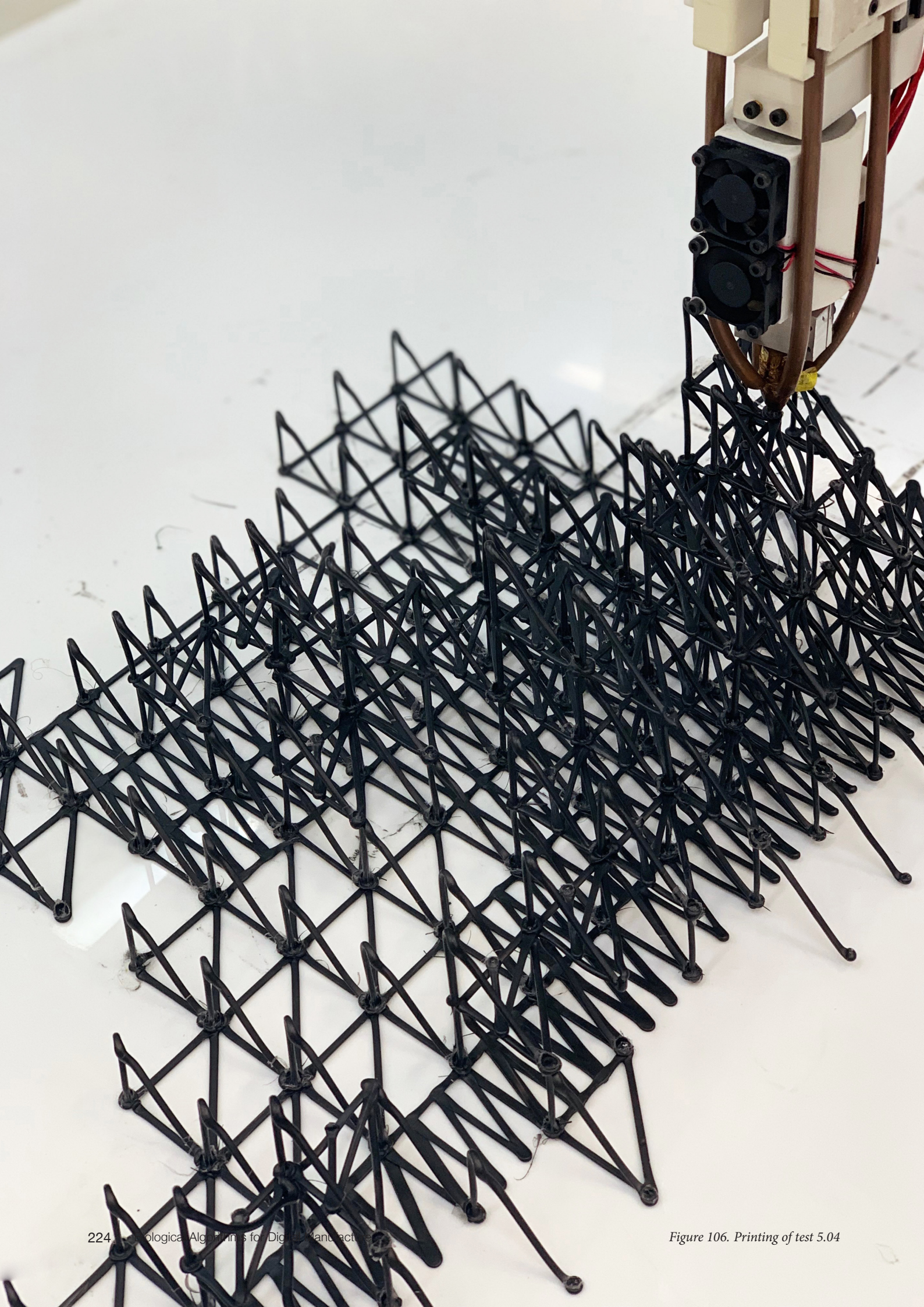


■ RESOLUTION ■

This effect was due to the filament being too hot. To combat this the output voltage of the heating system was set at the lowest. The fan was also set to run through all sections of the print, not just during the upright sections.

When the extruder turns on at an incorrect time this is due to either the extruder commands input into the Rapid Code incorrectly, but most likely due to the wait times before turning on not being long enough. With the wait times increased this ensure each print started at its scripted position.

With the extruder moving between points, it caused a small amount of material to be carried with it. To combat this, the wait time was increased to ensure it has stopped extruding when it reached the point to turn off. Increasing the speed between each of these points also stopped the print catching on the nozzle as the high speed would break it off.



6.08

PRINTER REFLECTION

The use of the new 3D printer bought many positives and negatives to the fabrication phase of the project. It has a lot of potential to be used to develop much more complex and intricate models than this research was attempting to. However due to time constraints there is no possibility to explore these potentials.

The use of the extruder in the final two weeks allowed for a quick fabrication of these final sets of models to test the digital output of the swarm and voxel systems. The script were less complex than expected without using it, once the commands section and the Rapid Code was set up.

The print errors discussed were able to be resolved and the final print in the previous set was able to be printed quicker, easier and removed the need to code in one continuous line.

The resolution of the print thickness has also increased from the previous model, with potentially a larger nozzle on the end. This is possibly having an impact of the Z offset and the print sagging due to more weight on the vertical members. To test this in future prints the extrusion rate will be lowered..

There are a range of differences between the two machines, and with time and experimentation the second model will far out perform the original printer.

	ORIGINAL PRINTER	NEW 3D PRINTER
PRINT SPEED	1.2 - 1.6mm/s	4mm/s
COOLING	Compressed air on extruder nozzle	Fan blowing down in direction of nozzle
CODE CONTROL	Controlled independently using Arduino commands	Controlled within Rapid Code
WAIT TIMES	Compulsory 1 second	0.3 - 1 seconds
PRINTING SYSTEM	Restrained by one line print	Able to stop printing and move to next section of model
PRINT TIMES	Small scale prints would take over an hour to complete	Quick testing at small scale results in fast feedback loop

CONCLUSION

Using the digital and fabrication methodology described at the beginning of this chapter, these sets of prints have been able to successfully be fabricated. A series of minor print issues arose when the change to the second extruder occurred. This however greatly advantaged the printing process and sped up both the script and print times.

The second set of prints within this chapter tested the density and variability possible when using a voxel-based fabrication method. The ability to print each of these prints at this scale, implies that the scale is able to be greatly increased and the print will remain successful.

The density section that was modelled in Chapter 5 would truly take advantage of the abilities of the new extruder. A series of digital models were developed with variable density by further populating a voxel space. Using the stop and start methods of the extruder it avoided the issues with interference that could possibly inhibit this concept from being printed. In future research this would be able to be implemented easily within a voxel design.

It is clear that the digital models that were simulated throughout this research were able to be fabricated using the methods described and the two extruders that were available. With further time and experimentation these models will be able to be scaled greatly, taking advantage of the reach of the robot arm, as these prints are very small compared to its capabilities.

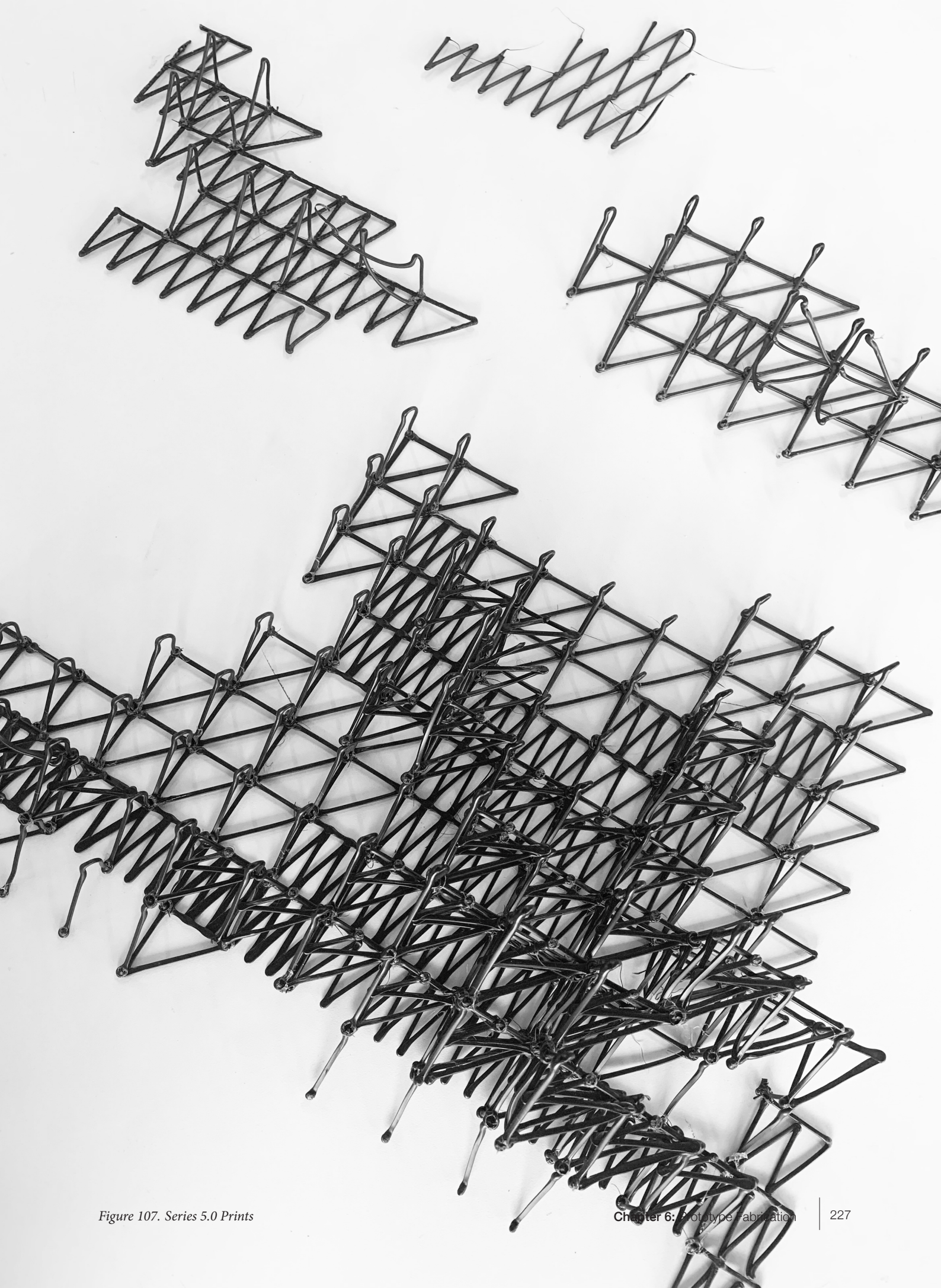


Figure 107. Series 5.0 Prints

CHAPTER 7:

DISCUSSION

/ Biological Algorithms for Digital Manufacture

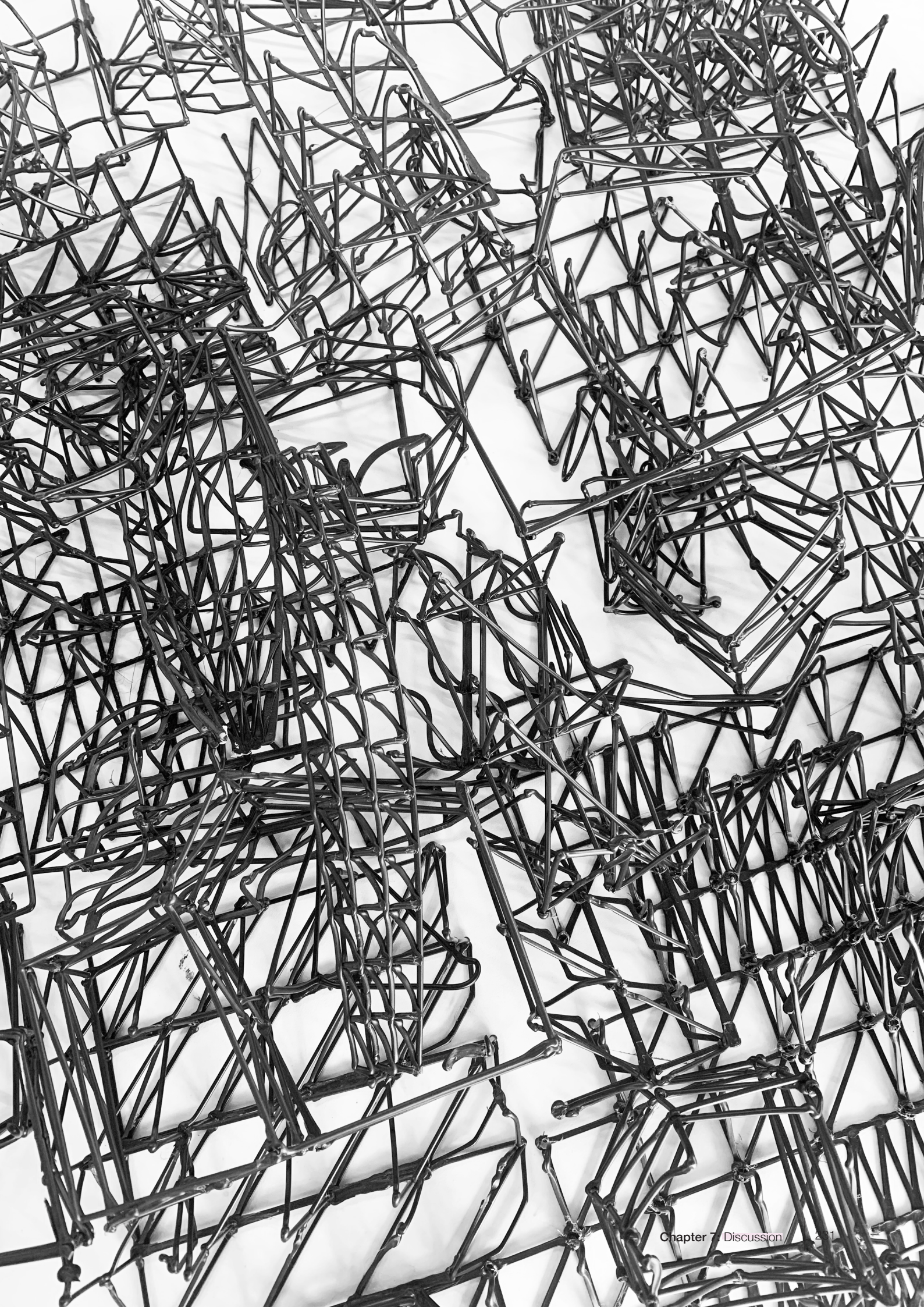
LIMITATIONS

Embarking on a digital project in a field that is unfamiliar, there were many limitations that were to be expected. While certain points in the research were one ended and were not continued in the direction described in the introduction, there were many issues at each level.

The unfamiliarity of agent-based systems and Processing coding in Java to resulted in, relying on on-line resources for guidance. This inhibited the growth of the Processing simulations as time was spent on learning the basics and not adding layers of detail and attributes to each simulation. With more time and experience each Processing simulation would be able to evolve into a layered data system holding multiple underlying design intents.

Limitations within the use of the robot also halter certain areas of the research. As discussed, issues arose with the use of both extruders and their capabilities. These errors were known and were tools rather than techniques, and overcome at various points during the research.

As a speculative design aimed to be fabricated, there is always issues with the unknown and the capabilities of the tools. This has been documented throughout the process, where issues arose at the introduction of each new software, concept or tool. However through rigorous testing, both the limitations and opportunities of these are able to be exposed and built on. The learnings from these limitations strengthened the project and built foundations for the following sections to be based on, rather than halting the research.



OPPORTUNITIES

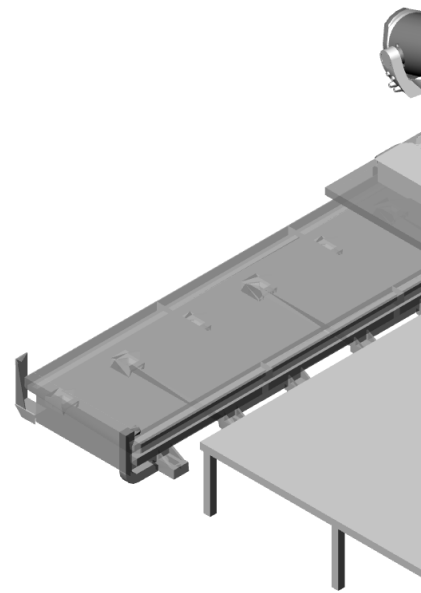
Many of the design limitations also turn into opportunities for this research to be extended further into the digital and fabrication realms.

Once the limitations in using the robot arm are solved, there is potential to extend the fabrication process and increase the space and tool capacity. The use of a larger workspace would allow for a second robot, or also a robot on rails. This would allow a much greater scale increase along the X-axis, where the robot could be used to build long wall-like structures such as the one modelled in Chapter 5. The use of a fabrication technique such as voxels, would suggest that 3D printing at this scale would have the same successful output that it does at the smaller prints in this thesis.

The availability period of the second extruder caused many problems in the research. There was a small test period time that involved getting used to the behaviour of the extruder. However this was at such a late point in the research, if this tool had of been made available five months earlier, there would have not been a need to transfer to a voxel-based fabrication system.

The use of voxels has its advantages, but the initial aim of this research was to test a free-form printer beyond that of an orthogonal grid. It was found that this is highly impossible without the use of an extruder that can stop and start printing. This change would not have been needed but did provide a thorough system to finish the research on.

With more time this research would be able to step back to the “Digital Research’ chapter and begin to focus on how to fabricated the agent-based systems thanks to the use of a more-advanced extruder.



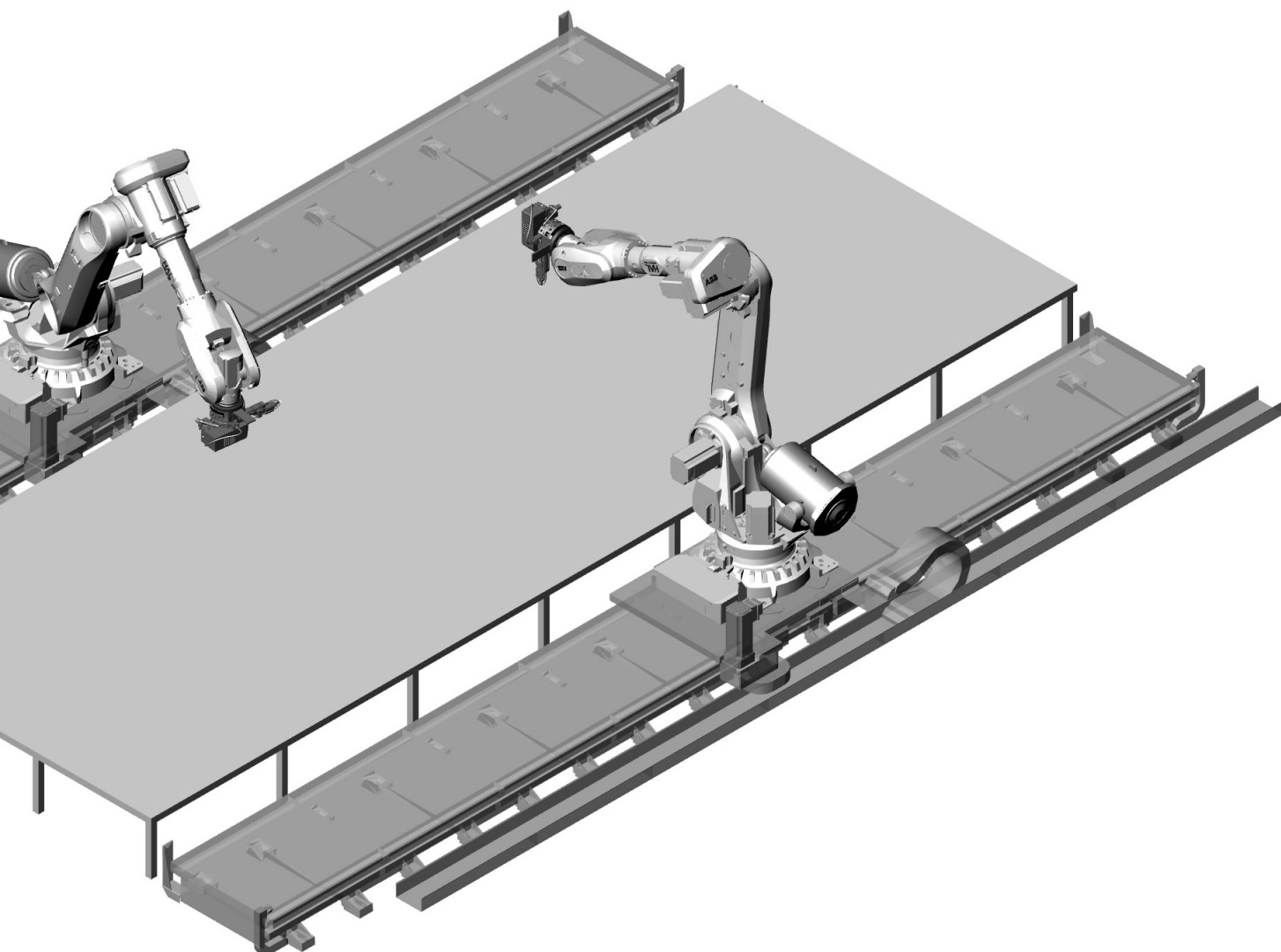


Figure 109. Two robotic arms on rails with 3D printers

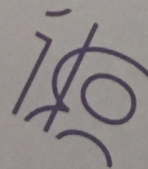


MASTERS THESIS

Louise Wotton
Masters of Architecture
Manufacture

In science, the word emergence refers to the production of forms and their behaviors, by systems of a collective nature. Emergent agent based simulations, are a tool used greatly within many areas of research as a speculative method to build form and space. The simulations in this thesis explore how an agent processes information from its environment and calculates an action input into these simulations are further parameters and variables are input to the ability and limitations of a robotic freeform 3d printer. Research and testing of a robotic freeform 3d printer arm is a key part of this research. PLA plastic, which is used for these models, has its own material properties, these are coded into the simulation logic, informing the agents movement, connectivity and density. Programming algorithms are used for this as a physical form once fabricated and evaluated accuracy and kinematics of robotic programming allows these agent based systems to be printed as intricate skeletal shapes. Code based modelling and mass customisation through printing, each time accounting for both limitations and capabilities of 3-D free-form printing.

Supervisor: Kevin Sweet



CONCLUSION

This thesis aimed to outline a methodology and digital work-flow which closes the gap between digital simulations and fabrication. Through the use of multiple software's and key concepts this has been achieved. However, fabrication issues have limited the development of this research, in working through these issues the work-flow has become stronger with fundamental fabrication issues being overcome and resolved.

When architectural, or any sort of design is interpreted in a computational manner, each element must be thought through as a piece of the entire system. Similar to nature, which can inherently be seen as discrete, where it is many parts which make up a whole, computational design must also be considered as part to whole relationships. Computation design, seen as the writing of code, makes up a large part of this body of work, is seen as a fast-increasing role in architecture. Mario Carpo's, *The Second Digital turn* focusses in on this, where 'digital architecture' was once mass customization, and CAD/CAM. Today however computation favors digitally intelligent architecture, where the unprecedented power of computation can be used for form finding and optimization. This will potentially lead to the introduction of machine thinking and learning, and potentially artificial intelligence. This research has built the bridge that can push computation in the direction of machinic processes. The nature of the parametric design in this research is based on small complex data interpreted as physical shapes, which is possible to be expressed and translated into advanced digital design such as machine thinking. This would branch this research to an area where the computation would be able to take the design intent and parameters and self-evaluate, removing the need for a user or programmer to monitor it.

The inclusion of environmental design with the National Science Challenge gave a specific design intent, clearly showing how agent-based simulations are capable of having intent. When design is thought about as 'discrete' parts, it is key that each part is able to have a whole relationship. Through this research the environmental aspects were drawn in at each point, connected to both the digital and the fabrication sides. Environmental data acted as a driver for a configurable system, an emergent design was built, the system was run without intent on a final image, but instead on the final functionality. The embedding of the performance and parameters into the design linked the model directly to a potential site, which its orientation, size, scale and location is all able to be manipulated and re-fed back into the system.

Emergent processes that draw on nature, were a critical explorative research element of this thesis. Although just as important, the digital fabrication methods explored in this thesis have the potentials to build frameworks to fabricate future semi-permanent and permanent architecture. With sustainability, construction methods, and innovation and important talking point in architecture and many other specialties, it is highly possible that 3D printing become more common in fabrication.

Analysing the outcomes of this thesis, I believe that future research would benefit by focusing on innovative fabrication and its link to other areas of architecture. One of the most difficult sections was the translation from a digital model to a built one. A stronger relationship from the computer to a built model, would increase the awareness for digitally-crafted design. The separation of architecture and construction is able to be closed through clearer communication and understanding of both parties. Construction or fabrication is not something that can be overtaken by robots, where people are no longer necessary. Robotics, or other forms of manufacturing will always need humans, to program, monitor and repair. The true element of craft and design should not be lost through a transition to a digital based architecture but retained and implemented in innovative ways that are also able to take advantage of the possibilities digital fabrication has to offer.

In conclusion, this body of work provides a depiction of a **work-flow and design approach outcome that takes a natural-based system and translates this into a fabricated model**. This aligns with the current progressive architectural movement, where the introduction and development of all things digital are becoming more and more prevalent in architecture.

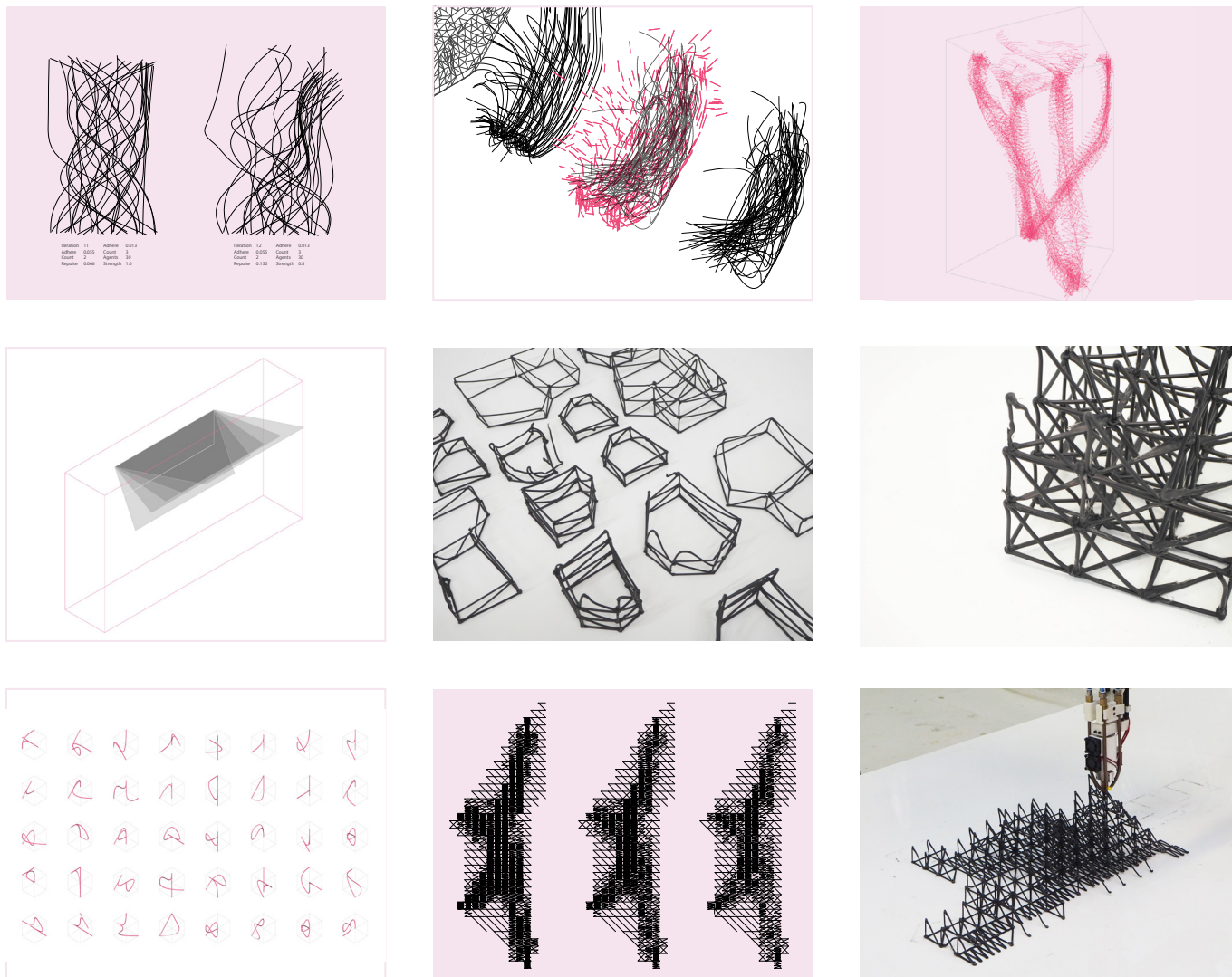
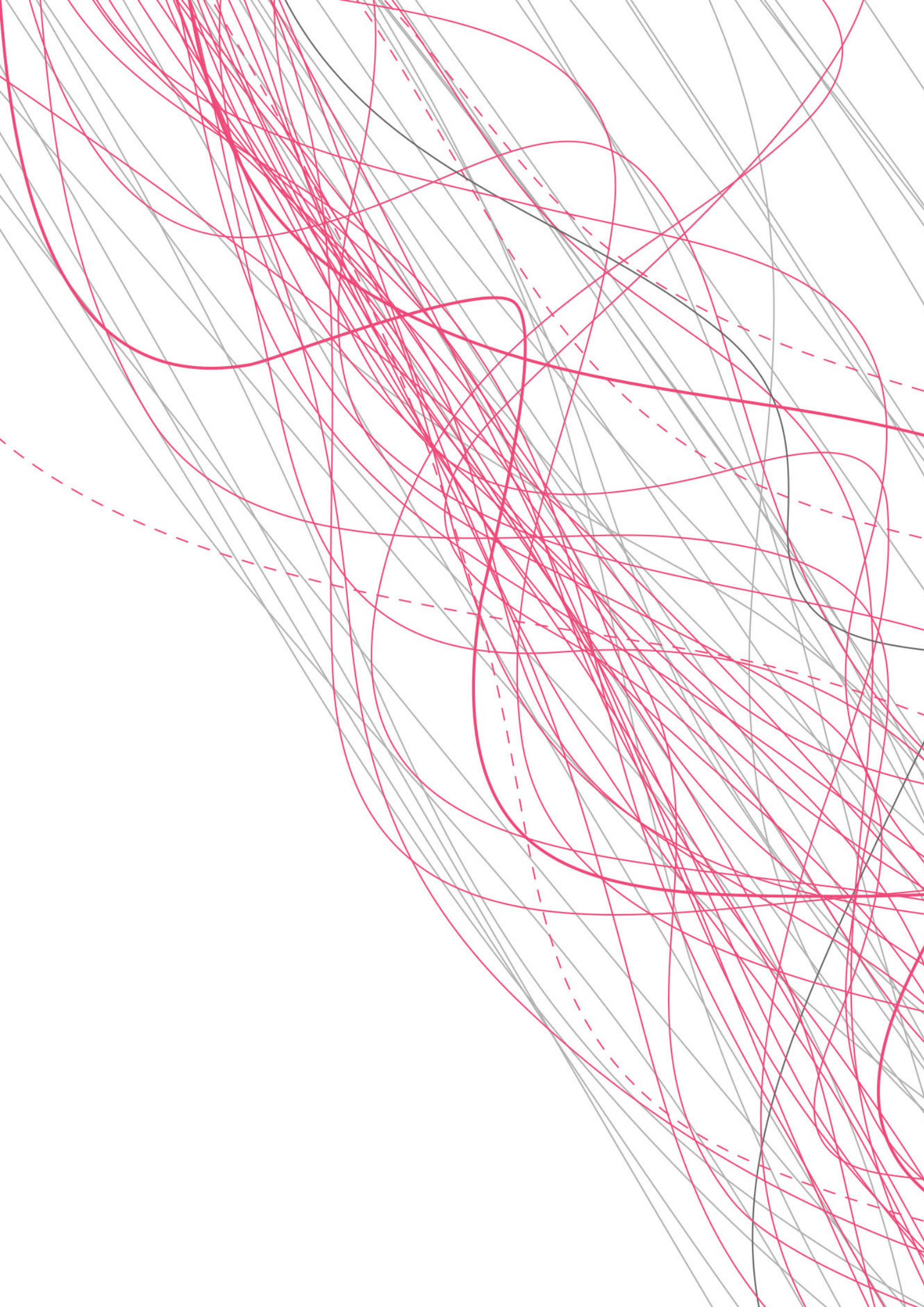


Figure 111. Series of Images of process through research project





WORKS CITED

/ Biological Algorithms for Digital Manufacture

WORKS CITED

- AI Build. (2016). Daedulus Pavilion. Retrieved February 3, 2019, from <https://ai-build.com/daedulus.html>
- Andrasek, A. (2018). Cloud Pergola. Retrieved February 3, 2019, from <https://www.alisaandrasek.com/projects/cloud-pergola>
- ASTM. “ASTM F2792-10 Standard Terminology for Additive Manufacturing Technologies.” American Society for Testing and Materials (ASTM), <http://www.astm.org/DATABASE.CART/HISTORICAL/F2792-10.htm>.
- Bazalo, F. (2014). Responsive Algorithms: An investigation of computational processes in early stage design. Architecture. Victoria University of Wellington, Wellington.
- BigRep. (2013). High Temperature Resistant 3D Printing Filament. Retrieved January 25, 2019, from <https://bigrep.com/matria/bigrep-pro-ht/>.
- Bonabeau, E. (1999). Definitions of Stigmergy. *Artificial Life on Stigmergy*, 5(2).
- Buš, P. (2016). Emergence in Urban Environments: Agent-based Simulation of Environment Reconfiguration. <https://doi.org/10.13140/RG.2.2.14814.13121>
- Carpo, M. (2011). *The Alphabet and the Algorithm* (1st ed.). The MIT Press.
- Carpo, M. (2017). *The Second Digital Turn: Design Beyond Intelligence*. London, England: The MIT Press.
- Coding Game. (2018). Dijkstra’s Algorithm. Retrieved December 20, 2018, from <https://www.codinggame.com/playgrounds/1608/shortest-paths-with-dijkstras-algorithm/dijkstras-algorithm>
- Greg Lynn, Greg Lynn Form, ed. M. Rappolt (Rizzoli Universe Int. Pub, 2008).
- Heylighen, F. (2016). Stigmergy as a universal coordination mechanism I: Definition and components. *Cognitive Systems Research*, 38, 4–13. <https://doi.org/https://doi.org/10.1016/j.cogsys.2015.12.002>
- ITKE, I. (2014). ICD ITKE Research Pavilion 2013-2014.

- Jiang, N., Wang, Y., Chen, Y., & Ahmed, Z. Y. (2014). *Spacewires - Filamentrics*. University College London.
- Jim. (2005). *Giant's Causeway, Co.* Retrieved February 15, 2019, from <https://www.flickr.com/photos/alphageek/>
- Karatay, M. \. (2006). *File:Safari ants.jpg*. Retrieved February 12, 2019, from https://commons.wikimedia.org/wiki/File:Safari_ants.jpg
- Kilkelly, M. (2016). *5 Ways Computational Design Will Change the Way You Work*. ArchSmarter. Retrieved from <https://archsmarter.com/computational-design/>
- Komar, M., Yilin, Y., Papadimitriou, A., & Castro, E. (2014). *Fibro.City*. University College of London.
- Kwon, H., Kaleel, A., & Li, X. (2015). *CurVoxels | Spatial Curves*. University College London.
- Li, M., Wong, O., Kim, D., & Homthong, S. (2016). *WireVoxels*. University College London.
- Lim, S., Buswell, R. A., Le, T. T., Austin, S. A., Gibb, A. G. F., & Thorpe, T. (2012). Developments in construction-scale additive manufacturing processes. *Automation in Construction*, 21, 262–268. <https://doi.org/https://doi.org/10.1016/j.autcon.2011.06.010>
- Loy, J., Canning, S., & Haskell, N. (2016). *3D Printing Sociocultural Sustainability* (pp. 51–72). https://doi.org/10.1007/978-981-10-0549-7_4
- Merholz, P. (2002). *Organized chaos: Emergence: The connected lives of ants, brains, cities, and software*. *New Architect*, 7(3), 56.
- Minar, N., Burkhart, R., Langton, C., & Askenazi, M. (1996). *The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations*. Santa Fe Institute Working Paper (Vol. 96-6–42).
- Ministry of Business Innovation and Employment. (2018). *National Science Challenges*. Retrieved February 3, 2019, from <https://www.mbie.govt.nz/science-and-technology/science-and-innovation/funding-information-and-opportunities/investment-funds/national-science-challenges/>
- Mitchell Waldrop, M. (1992). *Complexity; The Emerging Science of Order and Chaos*. New York: Simon and Schuster.

WORKS CITED

- MX3D. (2018). MX3D Bridge. Retrieved February 3, 2019, from <https://mx3d.com/projects/bridge/>
- O'Connor, J. (1997). Tips for Daylighting.
- Oxman, N. (2015). Design at the intersection of technology and biology. (TedTalks, Ed.).
- Papageorge, A. (2018). Free-form 3D Printing: A Sustainable, Efficient Construction Alternative. Victoria University of Wellington.
- Perony, N. (2013). Puppies! Now that I've got your attention, complexity theory. (TedxZurich, Ed.). TedTalks.
- Petrš, J. (2016). Application of Intelligence of Swarm in Architecture. Prague.
- ProgramArchitecture,. (2013). Programming Architecture - What is your profession?. Retrieved from <https://www.youtube.com/watch?v=jwRPQrxxNDA>
- Popov, N. (2010). Exploring Architectural Possibilities with Flocking Algorithms (pp. 385–398). Milan, Italy.
- R, O. R. & O. (2014). Theories of the Digital in Architecture (1st ed.).
- Ruy, D. (2016). Serving, Owning, Authoring. AD Magazine, (Digital Property : Open-Source Architecture), 141.
- Sanchez, J. (2016). Plethora Project. Retrieved February 3, 2019, from <https://www.plethora-project.com>
- Schiffman, D. (2012). The Nature of Code.
- Snooks, R. (2014a). Behavioral Formation | Multi-Agent Algorithmic Design Strategies. RMIT.
- Snooks, R. (2014b). Composite Wing. Retrieved February 3, 2019, from <http://www.rolandsnooks.com/#/compositewing/>
- Snooks, R. (2018). Kokkugia. Retrieved from <https://www.kokkugia.com>

- Sugihara, S. (2016). A(g)ntense. Retrieved from <https://vimeo.com/86229309>
- Taron, J. M., & Parker, M. (2014). Bounded Agency. In *Acadia 2014* (pp. 33–42).
- Terzidiz, K. (2006). *Algorithmic Architecture* (1st ed.). Oxford Architectural PRes.
- Tibbits, S. (2015). The emergence of “4D printing.” (TedTalks, Ed.).
- Webster, M. (n.d.). Volatile. Retrieved January 11, 2019, from <https://www.merriam-webster.com/dictionary/volatile>
- Wolfram, S. (2002). *A New Kind of Science*. Champaign, IL: Wolfram Media.
- Yap, B. V. Y. (2005). Termite Cathedral. Retrieved February 15, 2019, from https://en.wikipedia.org/wiki/File:Termite_Cathedral_DSC03570.jpg
- Yuliya, B. (2015). *Knitflatable Architecture: Pneumatically activated pre-programmed knitted textile Spaces*. University College London.

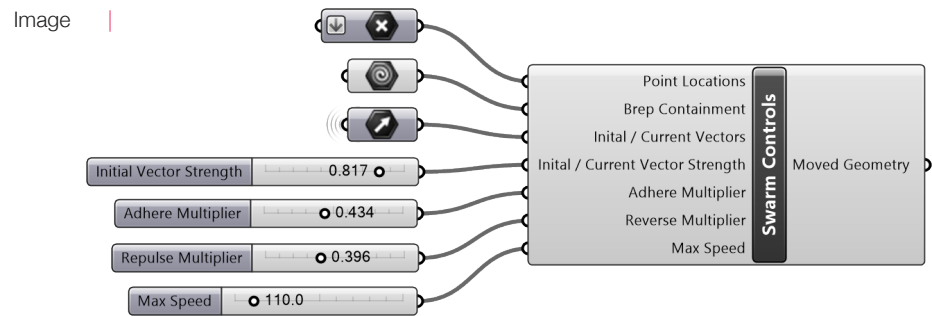
APPENDIX

/ Biological Algorithms for Digital Manufacture

GRASSHOPPER CLUSTERS

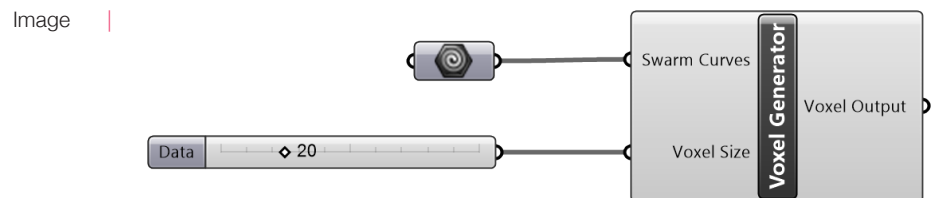
1. SWARM CONTROLS

- Location | Beginning of swarm simulation
- Purpose | Builds entirety of base swarm simulation using compounding vectors within looping script.
- Output | Geometry in the form of points, to be input into the looping script to build swarm simulation to be interpreted as curves



2. VOXEL GENERATOR

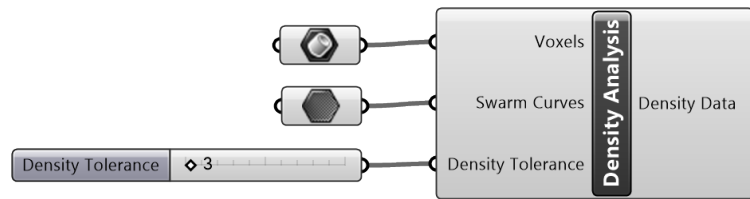
- Location | After swarm simulation
- Purpose | Builds series of voxels in the place of swarm curves
- Output | Box geometry



3. DENSITY ANALYSIS

Location | Before voxel instantiation
 Purpose | Analyses amount of points within voxel to determine the density of the print
 Output | Series of integers for density values

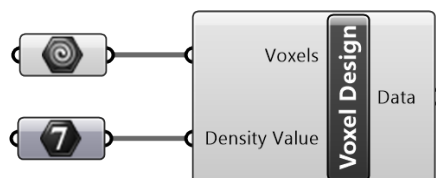
Image |



4. DENSITY VOXEL DESIGN

Location | After density data analysis
 Purpose | Instantiates print designs within a voxel. Print dependent on the output from the analysis. Individual design controlled within cluster by manipulating curves between box vertices's.
 Output | Print curves and points for HAL to interpret

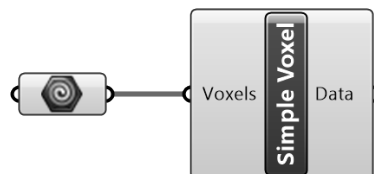
Image |



5. SIMPLE VOXEL DESIGN

Location | After voxel generator
 Purpose | Instantiates print designs within a voxel
 Output | Print curves and points for HAL to interpret

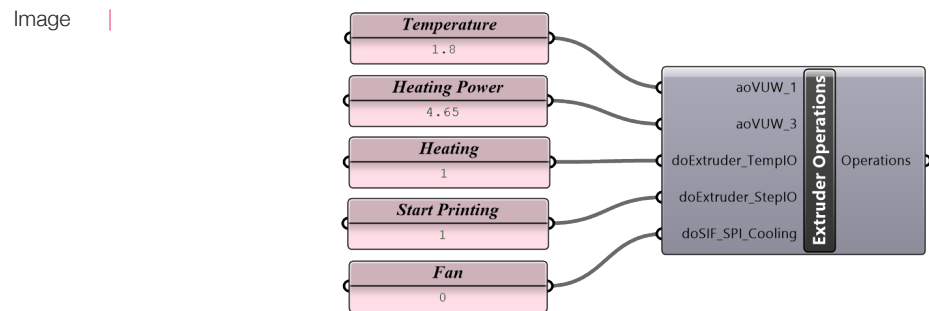
Image |



GRASSHOPPER CLUSTERS

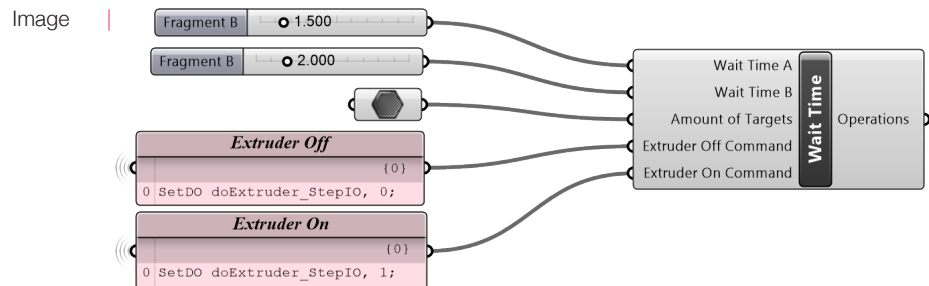
6. EXTRUDER OPERATIONS

- Location | Within HAL programming
- Purpose | Builds initial operations at beginning of robot routines. Inputs temperature and the initiation of the printer working, as well as the use of the fan
- Output | Series of text code operations to be input into toolpath designer



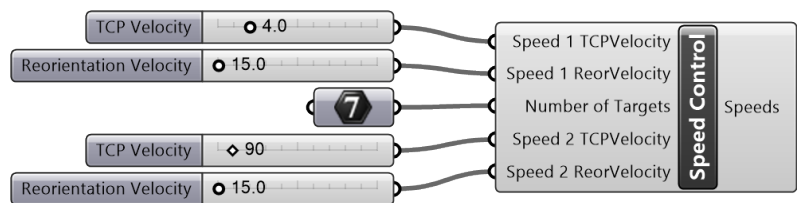
7. WAIT TIME COMMANDS

- Location | Within HAL programming
- Purpose | Builds Wait time code at certain targets to control the movement of the robot
- Output | Series of text code operations to be input into toolpath designer



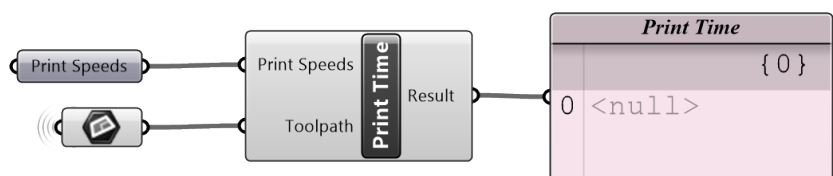
8. SPEED CONTROL

- Location | Within HAL programming
- Purpose | Builds a series of speed codes to control the robot. A number of different speeds are able to be input at different targets to control movement dependant on current task
- Output | Series of text code operations to be input into toolpath designer
- Image |



9. PRINT TIME CALCULATIONS

- Location | Adjacent to HAL programming
- Purpose | Calculates the amount of time a print will take. Calculated using the print speeds and the corresponding planes each will move to count the distance between each
- Output | Single integer of the time for the print to complete (in hours)
- Image |




```
public void flockLouise(ArrayList <Ple_Agent> boids, float desCoh , float desAli, float desSep ,
float cohScale, float aliScale, float sepScale){
```

```
//FLOCK : COH-ALI-SEP (true,true,true);
//COH --> Flock(true,false,false);
//ALI --> Flock(false,true,false);
//SEP --> Flock(false,false,true);
```

```
float cohNeighborDist = desCoh;
float desiredSeparation = desSep;
float aliNeighborDist = desAli;
```

```
Vec3D coh = new Vec3D();
Vec3D cohReturn = new Vec3D();
```

```
Vec3D sep = new Vec3D();
Vec3D sepReturn = new Vec3D();
```

```
Vec3D ali = new Vec3D();
Vec3D aliReturn = new Vec3D();
```

```
int count1 = 0;
int count2 = 0;
int count3 = 0; } amount of times each attribute is applied
```

```
for (int i = boids.size()-1; i >= 0; i--) {
Ple_Agent other = (Ple_Agent) boids.get(i);
//COHESION:
if (this != other) { incompatible operand types sketch_0509
if (loc.distanceToSquared(other.loc) < cohNeighborDist*cohNeighborDist) {
coh.addSelf(other.loc);
count1++;
}
}
```

maxSep - distance where agents get too far apart & need to come back together
minSep - distance where the separation rules should apply.

```
//SEPARATION:
if (this != other) {
float d = loc.distanceTo(other.loc); defining / finding distance to closest
// If the distance is greater than 0 and less than an arbitrary amount (0 when you are yourself)
if (d < desiredSeparation) { if d is smaller than desired separation.
// Calculate vector pointing away from neighbor
Vec3D diff = loc.sub(other.loc);
diff.normalizeTo(1.0f/d);
sep.addSelf(diff);
count2++;
}
}
```

```
//JOIN
if (this != other) {
```

```
if d > (desiredSeparation + limit) {
```

need the inverse to go back in

```
//ALIGNMENT:
if (this != other) {
if (loc.distanceToSquared(other.loc) < aliNeighborDist*aliNeighborDist) {
ali.addSelf(other.vel);
count3++;
}
}
```

```
//JOIN
```

⚡

join neighbour distance.

```
//COHESION:
if (count1 > 0) {
coh.scaleSelf(1.0f/count1);
cohReturn = steer(coh,false); }
//SEPARATION:
if (count2 > 0) {
sep.scaleSelf(1.0f/count2);
}
```

amount of changes found

steer - method that calculates a steering vector towards a target
using this boolean will have it slow down as it reaches the target

```
if (sep.magSquared() > 0) {
sep.normalizeTo(maxspeed);
sep.subSelf(vel);
sep.limit(maxforce);
}
```

```
sepReturn = sep;
```

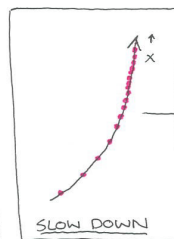
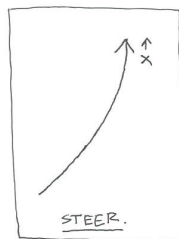
```
//ALIGNMENT:
if (count3 > 0) {
ali.scaleSelf(1.0f/count3);
}
```

```
if (ali.magSquared() > 0) {
ali.normalizeTo(maxspeed);
ali.subSelf(vel);
ali.limit(maxforce);
}
```

```
aliReturn = ali;
```

```
//-----
sepReturn.scaleSelf(sepScale);
cohReturn.scaleSelf(cohScale);
aliReturn.scaleSelf(aliScale);
```

```
acc.addSelf(sepReturn);
acc.addSelf(aliReturn);
acc.addSelf(cohReturn);
}
```



having more points / slowing down means robot printing will be thicker around connections

-dot- provides access to an object's methods and data. An object is one instance of a class and may contain both methods (functions) and data (variables & constants).
→ as specified in the class definition. The dot operator directs the program to the info encapsulated within an object.
authorship in the digital realm

Louise Wotton
Master of Architecture (Professional)
2019