CULTURAL URBANISATION:

The use of behavioural simulation in the design of indigenous urban settlements

Ariana Jemima Ballantyne

Cultural Urbanisation:
The use of behavioural simulation in the design of indigenous urban settlements

by
Ariana Jemima Ballantyne

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

Victoria University of Wellington
2017

Acknowledgements

There are many people who have contributed to the completion of this thesis, whether they know it or not, and I would like to offer special thanks to the following people -
I would like to thank my supervisor Derek Kawiti, for trusting me with the freedom to explore this topic in my own way, and knowing when to offer guidance.
I would also like to thank Marc Aurel Schnabel for your generosity; I greatly appreciate your time and expertise, as well as the reassurance it offered.
To the Architectural Science Association, I wish to offer thanks for publishing my paper, the people I talked to and the confidence I gained from that experience was invaluable.
To Shannon who went through this process with me, and to Prudence who dragged me away from it, thanks for always being there.
And lastly to my parents, thank you so much. Without the upbringing you gave me, I doubt I would have ever considered undertaking a project like this. Dad, thank you for always having faith in me, and Mum, thank you for being absolutely unique in the best way, I couldn't love either of you more.

CONTENTS:

Abstract....10
Introduction....12
Part I: Background
Chapter One: Māori Living....17
Chapter Two: Behavioural Relationships....31
Chapter Three: Multi-Agent Systems....41
Part II: Testing
Chapter Four: Processing....47
Chapter Five: 2D Simulations....57
Chapter Six: 3D Simulations....69
Part III: Results
Chapter Seven: Refinement....85
Chapter Eight: Interpreting Output....107
Chapter Nine: Architectural Implications....115
Conclusion....125
Glossary....131
References....132
List of Figures....134
Appendix....139

This thesis conducts an investigation into the use of multi-agent systems as a computational design research tool that implements a range of behavioural parameters relating to the design of specific cultural environments for Māori. It aims to offer an alternative methodology to the traditionally 'top-down' approach to Māori housing solutions within urban contexts, choosing instead to incorporate parameters that can be specific to a representative agent and their subsequent negotiation and interactions with other agents within a simulated environment.

This methodology works under the premise that by piecing together behavioural parameters that are specific to traditional Māori cultural environments, multi-agents can simulate these behaviours with respect to spatial occupation, establishing a system by which to construct the spatial organisation of a community of agents, and subsequently the communities they represent.

The use of cultural criteria enables us to contrast the research with standard multi agent simulations that operate on more generic rules of interaction. As a body of research it places an emphasis on the social and the collective identity or cohesion of bodies of single agents within a modern tribal structure as the main organisational vector. It is the hope that this methodology could lend itself to more diverse projects, aiding the design of spatial organisation for other socially orientated communities with needs beyond that of what can be provided by the western 'top-down' approach to architecture.

Keywords: Behavioural simulation, Cultural practices, Indigenous urban architecture, Multi-agent systems, Spatial organisation.

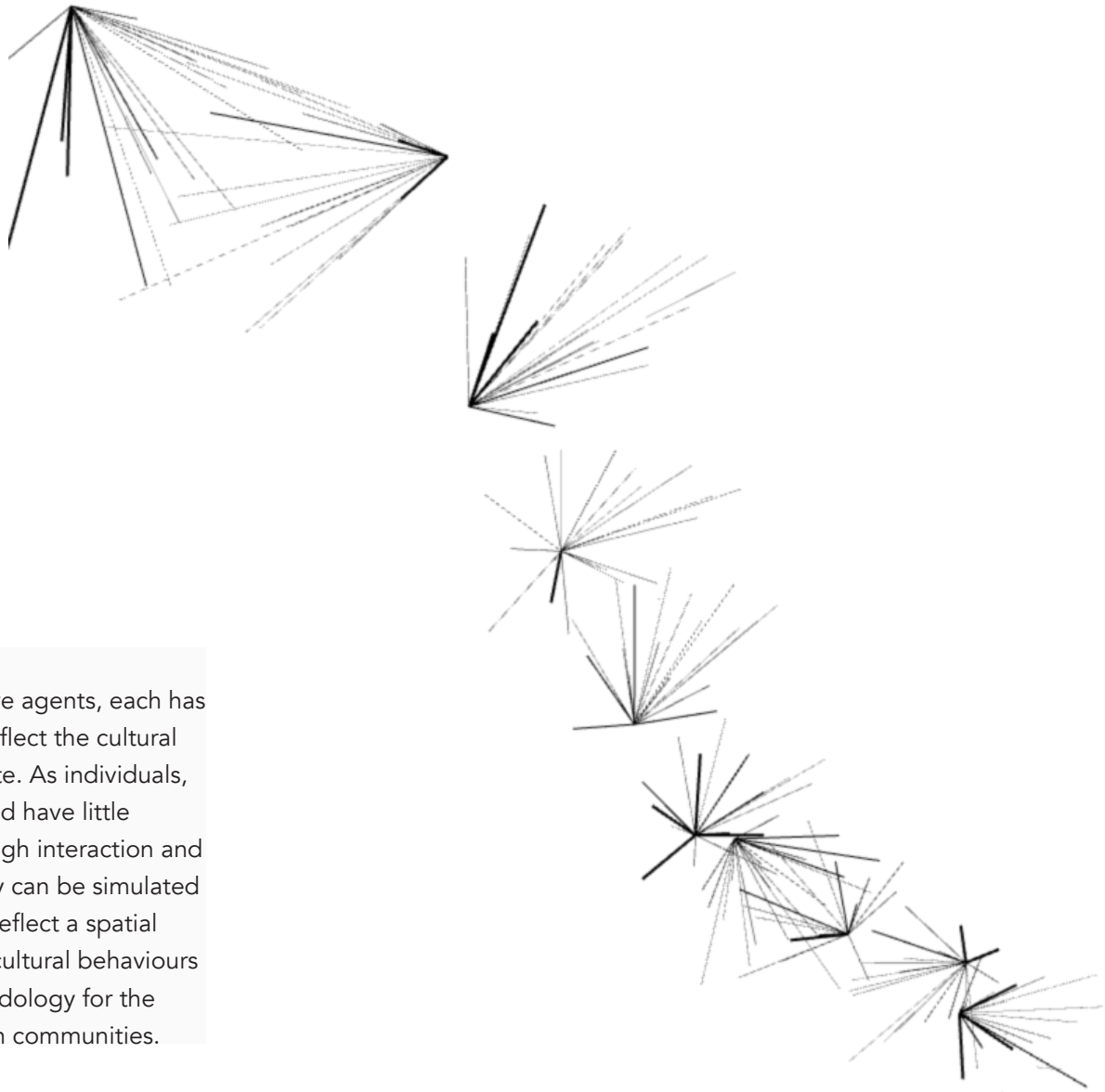


Figure 1: Individual Complexities
Within a community of representative agents, each has their own specific behaviours that reflect the cultural parameters under which they operate. As individuals, the behaviours of these agents would have little architectural bearing, however through interaction and negotiation within a community they can be simulated into a network of relationships that reflect a spatial organisation structure within which cultural behaviours operate, identifying a design methodology for the configuration of behaviourally driven communities.

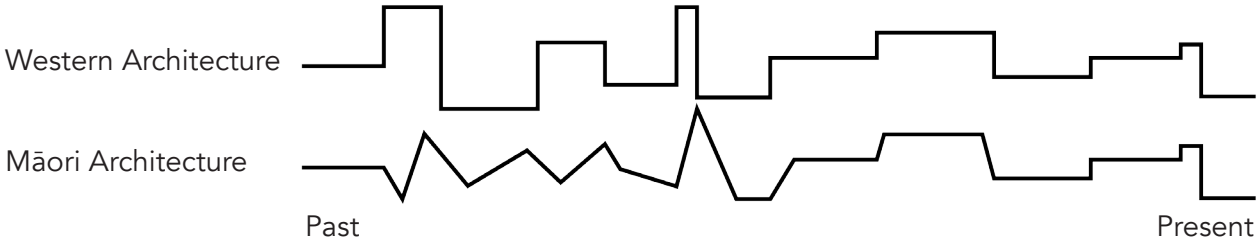
INTRODUCTION

There is an interesting phenomenon concerning the architecture of New Zealand, in which there are two different strands of architectural design. These two strands are separated by culture, one being influenced by western design, brought in to the country with European settlers, the other being indigenous, of Māori origin. While the western influenced strand flourishes, becoming more dominant as it adapts to the modern world, the indigenous strand has become more oppressed as it fails to make the transition into today's living environments.

While western influenced styles within New Zealand have mirrored that of international architecture, embracing the numerous styles of the times, from Georgian architecture to Gothic Revival, from Californian inspired bungalows to art deco, modernism, post-modernism and contemporary minimalist design, Māori architecture has faltered. With a stunted growth, Māori settlements have barely changed over the last few decades, with a very small range in typical model structures. Select western styles have leaked into Māori housing typologies, mirroring that of western styles, but the settlement structure has remained fairly constant, barely evolving beyond the rural papakainga models that have been around for decades. Understanding the reason behind this lack of adaptation, and identifying why Māori architecture has not found a place in modern living environments is the key to finding the common ground between the traditional and the modern, bringing cohesion to Māori architecture and the modern world, and reinstating a degree of cultural design independence.

The aim is not to create a specific building typology for Māori in urban environments, which in effect would isolate Māori from their western counterparts, rather the aim is to create a new methodology for design, one that can interpret cultural nuances in the design of Māori communities in urban settings, yet allows the same flexibility in style and construction as western design methodologies. When designing for urban environments, Māori and western architecture should align, they are both looking to address the same environmental issues but with different cultural interpretations, therefore this research looks to establish a methodology that puts aspects of the Māori culture at the forefront.

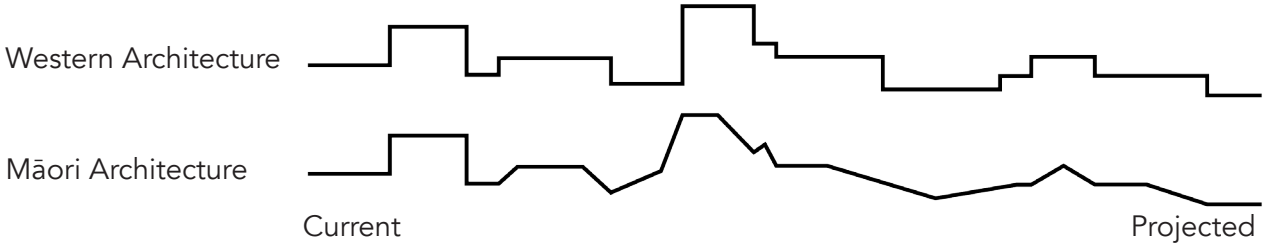
In the following chapters I discuss the composition of a typical traditional Māori papakainga, how the pertinent features create a culturally enriching environment, and why this has not as of yet been successfully translated into modern architectural design. Using multi-agent system technology as a medium for research, I investigate methods in which cultural spatial behaviours can be simulated, and look into how the architecture can be approached from a different perspective, in order to find a balance between the modern and the traditional.



This type of investigation could lead to methodology that is appropriate for designing for indigenous cultures around the world. Furthermore, this emerging field of culturally informed multi-agent research could be highly beneficial in facilitating the exposure of the Māori culture, reinstating it within the New Zealand architectural world, ensuring architectural relevancy is maintained through innovation.

Ray Bradbury once said, 'You don't have to burn books to destroy a culture, just get people to stop reading them'. The same applies to Māori architecture. We don't need to stop building to destroy the style; we just need to stop learning from it. Irrelevancy in the modern world is what will destroy the Māori way of life, and with an architecture that hasn't been evolving, hasn't been maintaining a level of cohesion with the modern world, the divide will be soon be insurmountable, and Māori architecture could be lost, taking a huge piece of cultural understanding with it.

Figure 2 (Opp.), 3 (Below): Past and Projected Interpretive Timelines of Western and Māori Architecture
2. Symbolises the shift from complete independence of architectural design, to the current mirroring of western architectural styles.
3. Describes the aim to shift from the complete mirroring of western architecture in urban settings, to regaining cultural independence through interpreting the same urban issues with Māori cultural considerations.



Part I - Background

CHAPTER ONE:

MĀORI LIVING

Architecture is as much about the human condition as it is about the built form, and understanding the users and movers within a particular spatial realm is the key to understanding the conditions for which we design.

This chapter identifies the composition and qualities within a traditional papakainga that provide the essence of the Māori way of life, while exploring how these factors may relate to the urban environment, as well as how this has been approached by current models which provide an interpretation of an urban papakainga. By understanding the issues within current urban papakainga models we can identify areas upon which to focus to expand knowledge to inform architectural design, and from there we can look into addressing the loss of genuinely Māori architecture within the modern world. The Māori culture is neither dead nor lost, so why do we allow an aspect as vital to cultural preservation as architecture, to become stagnant.

As a precursor to this chapter, it must be understood that there is no written rule, architectural code, or Māori tikanga that dictates the composition of a papakainga; it is simply logical arrangement according to cultural conditions that drives the formation of such a model (Schrader, 2013, pp. 1). Secondly, it must also be acknowledged that over time adjustments to the typical papakainga model have occurred as a result of European occupation, and the old traditional model has been transformed into an interpretation of the traditional, as Māori have adopted certain western elements to their settlement living (Martin, 1997). As these new elements have been successfully and permanently integrated into current interpretations of traditional papakainga settlements, they will be treated as authentic components of Māori living, integral to active papakainga communities.

Figure 4: Potatau Te Wherowhero's Pa
A 1847 George Angas painting depicts a traditional Māori papakainga in the early days of European occupation.
This fortified settlement can be seen to be made up of many housing units surrounding shared communal spaces, located in front of Taupiri maunga.

Image redacted for copyright reasons

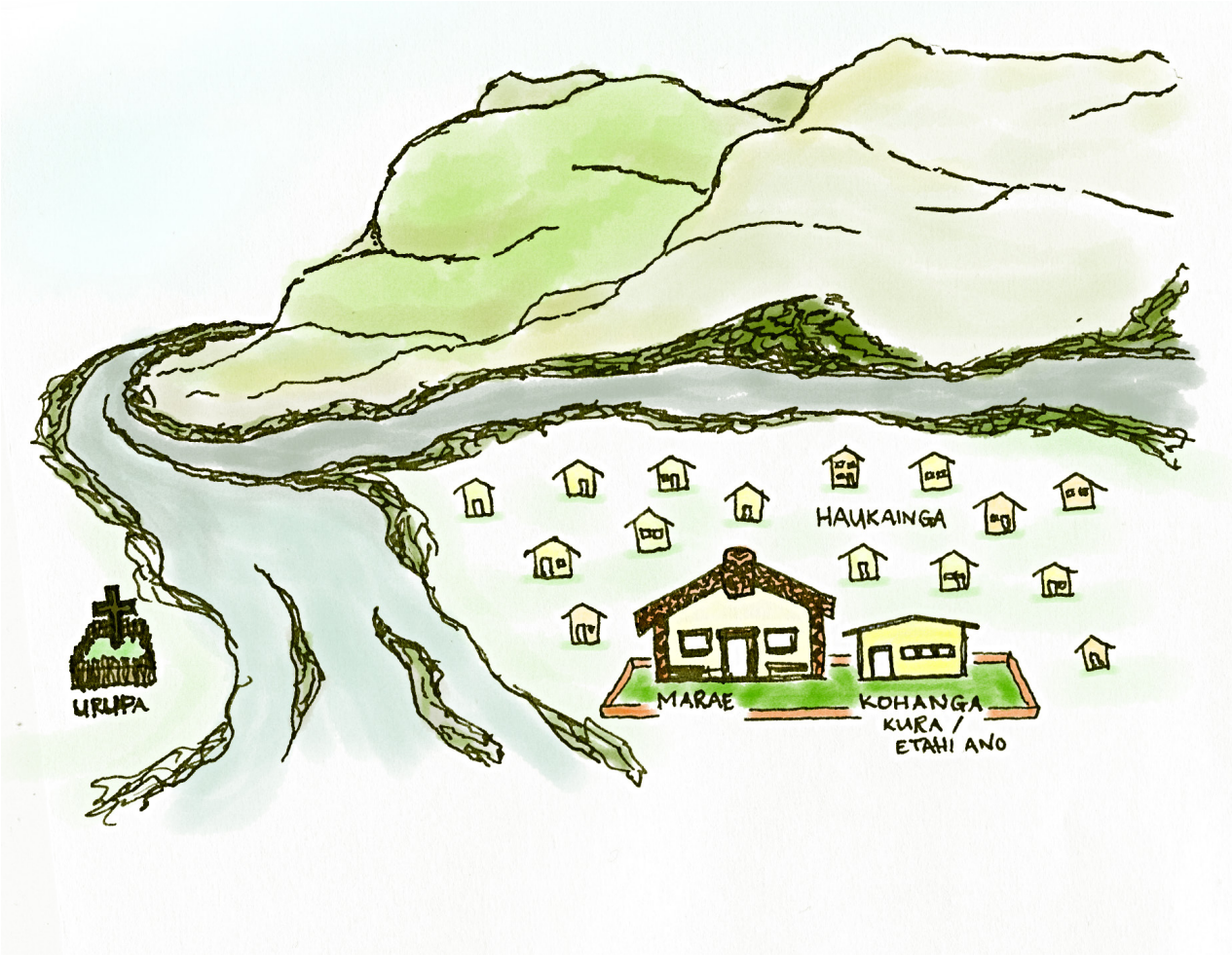


Figure 5:
Current Papakainga Composition Diagram
A descriptive diagrammatic model based on the author's interpretation of a current papakainga model informed by research included in Appendix Item 1. (Martin, 1997. Mead, 2016, pp. 78 - 93. Schrader, 2013, pp. 1 - 3).

COMPOSITION

In order to find commonalities between successful papakainga models, an exploration into typical papakainga was carried out, identifying the following components as being vital to the make up of a traditional Māori papakainga (Schrader, 2013, pp. 1 - 3, Martin, 1997, Joynt *et al*, 2016):

Shared communal spaces:

Typically centrally located, papakainga have at least one building that provides a degree of communal unity. A marae, often accompanied by a learning facility and occasionally an amenity (depending on level of isolation), provide for this function.

Private residences:

Dwellings are typically scattered, with simple housing typologies, many built using the simple state housing model. There is often very little in terms of private property, as is evidenced in physical and legal representation. Māori land is often co-owned, reducing individual claim, and boundary lines are often physically blurred, creating an open environment.

Landmarks:

Rural papakainga always have surrounding landmarks that relate to the history or beliefs of their iwi. Papakainga were settled in locations most often based on war and resources, so settlements are predominantly within close proximity to rivers, mountains, and areas of historical note; providing water, hunting grounds, and visual history to be passed down the generations.

VALUES AND PRACTICES

The values and practices that emerge from the unique compositional aspects of a rural papakainga are as follows (Burhardt and Swallow, 2014, pp.11-15, Mead, 2016, pp. 31, 163 - 178, 208):

Whanau:

Māori value whanau relationships to a great extent, operating under a nurturing system where no member is left isolated. Because of this, they tend to have large families that are multi-generational in structure, that stay within close proximity of each other.

Iwi:

Tribal connections offer a broader relationship base for Māori, and typically are the uniting factor within papakainga. Tribal connections are based on factors of heritage, with shared ancestors, histories, and belief systems offering points of commonality.

Whenua:

Māori have a well documented connection with the land, beyond that of property. Papatuanuku is the mother of gods, and with this reverence comes a sacred belief in land, and consequently iwi establish ties to land formations that have historical significance.

Turangawaewae:

Turangawaewae is a place to call home, and is of great significance to Māori. Turangawaewae is a rather alien concept in the urban world, as to Māori 'one (does) not own land, one belongs to land', and through this, people can always identify with home (Durie,1987, pp. 78).

Images redacted for copyright reasons

Figure 6, 7, 8, 9 (L to R): These figures depict the Māori values identified, a tipuna for whanau; an iwi collective; maunga, awa, and waka for the whenua; and a home for turangawaewae.

COMPARISONS TO URBAN LIVING

The contrasts between the rural Māori values mentioned and their urban counterparts begin to indicate why the transition is difficult, and how the lack of cultural alignment may be contributing to the issue (Fabian and Goodyear, 2014, pp. 66 -67):

Family:

It is unusual to find multiple generations co-residing in Western living, a belief in care institutions outweighing multi-generational living. This creates scenarios with more independence, looser proximity, and smaller family compositions.

Community:

Western communities are often formed based on location. Most central urban communities are inactive; it is more often fringe communities that have active social networks. These groups can be based on shared behaviours, but more often are individuals.

Land:

Within urban centres, land means little more than property lots for dwelling or income. Sites of historical significance are maintained and treasured, but rarely personalised, providing temporal significance to occupants.

Urban:

Urban centres emerged to satisfy a need for accessibility. It is the western equivalent of a large scale papakainga, providing for the needs of a western community, with mixed requirements pertaining to community ties and family life.

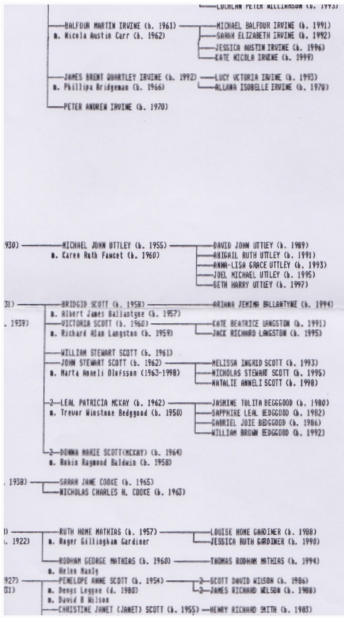


Figure 10, 11, 12, 13 (L to R):
These figures depict the western equivalent of the previous page. A family tree, cul-de-sac community, property lot, and a fast paced urban centre.

Images redacted for copyright reasons

URBAN TRANSLATIONS

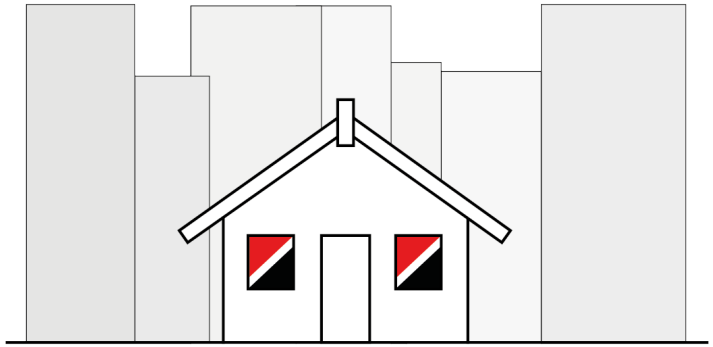
There have been recent attempts at establishing urban papakainga models, and as with any new development, these interpretations have faced intense scrutiny. The issue is that new developments up to this date can be viewed as falling into one of two models, which I identify as the isolation model, and the assimilation model. Both models are inadequate when it comes to providing the essence of a Māori papakainga in an urban setting, and in effect they either lose the mana of the papakainga or fail to embrace urban living, see Appendix Item 2 which describes a precedent study of previous interpretations of urban papakainga and informs these models. (Cropp, 2015).

Isolation Model:

The isolation model describes attempts at replicating rural papakainga in an urban context. There are instances where it is as if an entire village has been transported to the middle of an urban environment, and while the idea is logical in essence, there is no conversation to be had between the two, rather they are such separate identities that the papakainga becomes completely isolated from its surroundings.

Assimilation Model:

The assimilation model encompasses the larger group of urban papakainga interpretations, where housing is designed specifically for Māori to live in the urban environment. While considerations like room demands and planning composition is adjusted to meet larger Māori families, and more functional flexibility and communal spaces are offered and so forth, the approach is still very western in style. Analytical design is employed based on known spatial requirements, resulting in what is essentially urban housing in which a Māori community happens to reside.



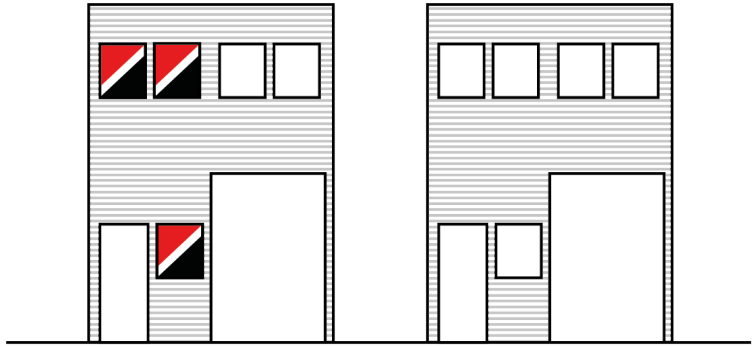


Figure 14, 15 (L to R):
Isolation vs Assimilation Model
The isolation model operates independent of its environment, holding true to Māori papakainga models, yet not to urban living. The assimilation model attempts to sculpt its users to fit into the environment, providing living environments that could be for anyone. Neither model provides a successful living environment for Māori in urban centres. Research included as Appendix Item 2.

UNDERLYING ISSUES AND FOCUS

What can be taken from this analysis is that the issues faced in finding a way to translate the papakainga model into an urban setting is not with the design of the built form itself, rather it is an issue of ignorance regarding the fundamental differences between Māori and western cultures. What is needed is a new model of integrating papakainga and urban environments, rather than isolation or assimilation, we need a model of adaptation, where commonalities are found between the Māori culture and western design practice, in order to establish a perspective of design that encompasses both aspects required for a successful settlement.

By this respect I conclude that the most significant area to focus upon is the difference in where value is placed in respect to settlement formation. In western living the focus is based on how needs are best serviced in relation to the immediate family within a housing unit, and which communities and housing typologies best satisfy those needs.
In the Māori world, settlements are based on iwi structure, and by that respect, based on relationships within a tribe, family and history. So where the western world bases housing on individual logical reasoning, the Māori world bases it upon a communal logic, with behavioural relationships forming the core of Māori living. This identifies the crucial factor that is missing from the western design process, causing the discordance between Māori communities and the urban environment.

$$\frac{2}{3} + \frac{4}{5} = ?$$

Method 1

$$\frac{2}{3} + \frac{4}{5} = \frac{6}{8}$$

Method 2

$$\frac{2 \times 5}{3 \times 5} + \frac{4 \times 3}{5 \times 3} = \frac{10}{15} + \frac{12}{15} = \frac{22}{15}$$

A simple way of looking at the issue would be as shown above, as a basic maths problem. Adding two different fractions can be approached in one of two ways. The first approach is far less complex, however will not provide the correct result, as two different entities cannot be treated as identical. The second method however, while more complex, ensures that the two entities are firstly converted to align with each other, before attempting to unify them, this results in a successful harmonious melding. It is this initial stage, of aligning the two entities that needs to be implemented in the transition of Māori communities to settings other than that of the traditional rural environment.

Part I - Background

CHAPTER TWO:

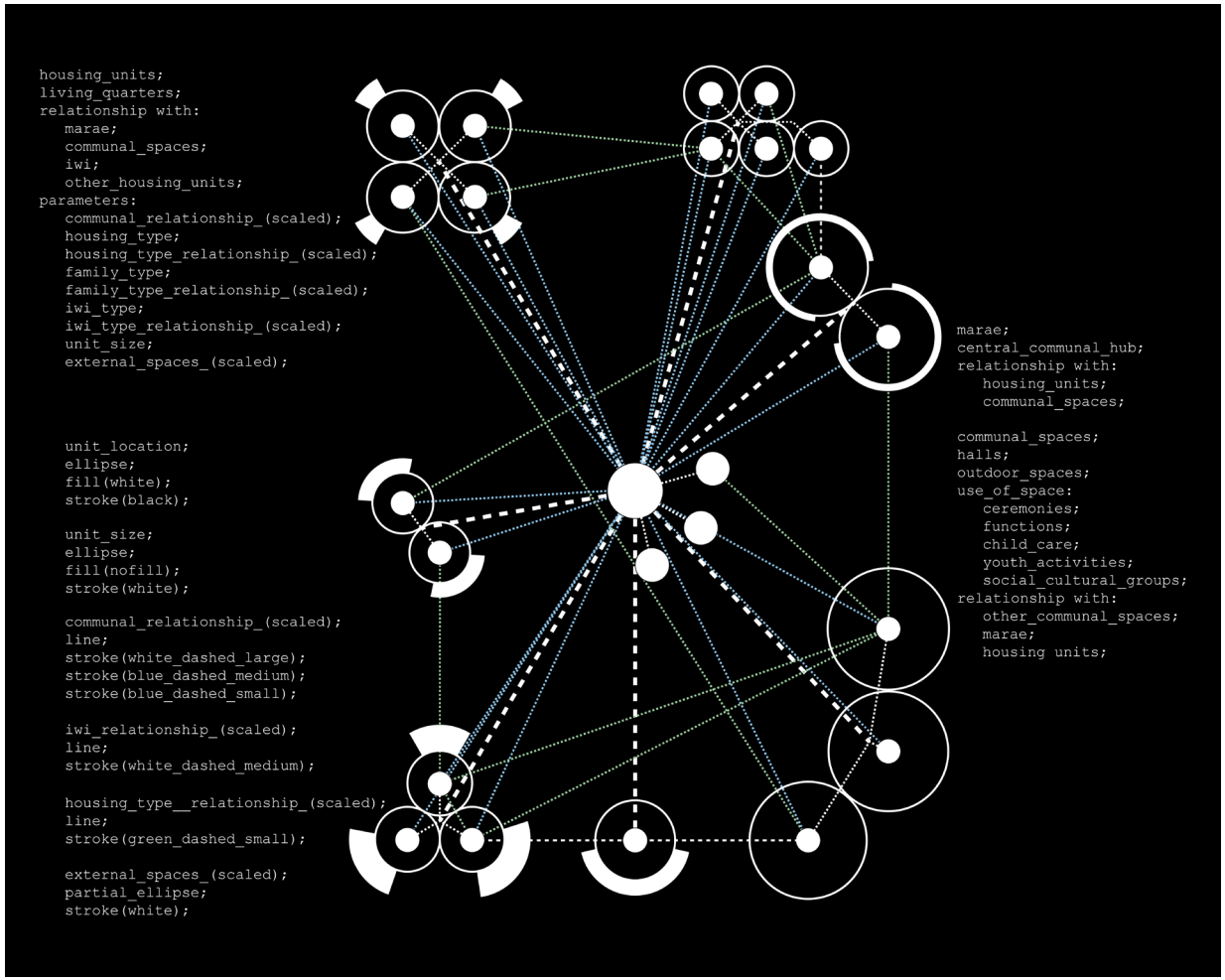
BEHAVIOURAL RELATIONSHIPS

As relationship structure has been identified as the basis of traditional Māori communities, it is important to understand what types of relationships form on a cultural level, and which are vital to ensuring the values of a Māori community can be replicated in an urban setting. The following relationship categories influence spatial organisation at a range of scales (Mead, 2016, pp. 163 - 178, 219 - 229);

- within the wider context, on a macro scale, concerning relationships between different iwi in the urban or rural setting
- within the community, unit organisation within a settlement
- within the unit itself, on a micro scale, addressing the composition of the unit itself

The scale that needs addressing the most in terms of finding a way to integrate Māori papakainga into urban contexts is the scale which addresses relationships within it's own settlement. This scale is the most complex in terms of relationship structure, and it requires the type of organisational methods that are the furthest from that of western understandings.
Within the wider context, western interpretations of community locations are valuable in terms of amenity prioritisation and financial motivations (which are realities in any urban development). Within the unit itself western design has already been adopted in rural papakainga settings (for example the use of state housing models in rural papakainga), however unit organisation and relationship structuring within a settlement has consistently been motivated by a purely Māori influence (Martin, 1997. Schrader, 2013, pp. 3). This is where we need to find a way to make this Māori influence understandable to designers, and find a way to adapt western design methods to be more inclusive of these as yet unknown qualities.

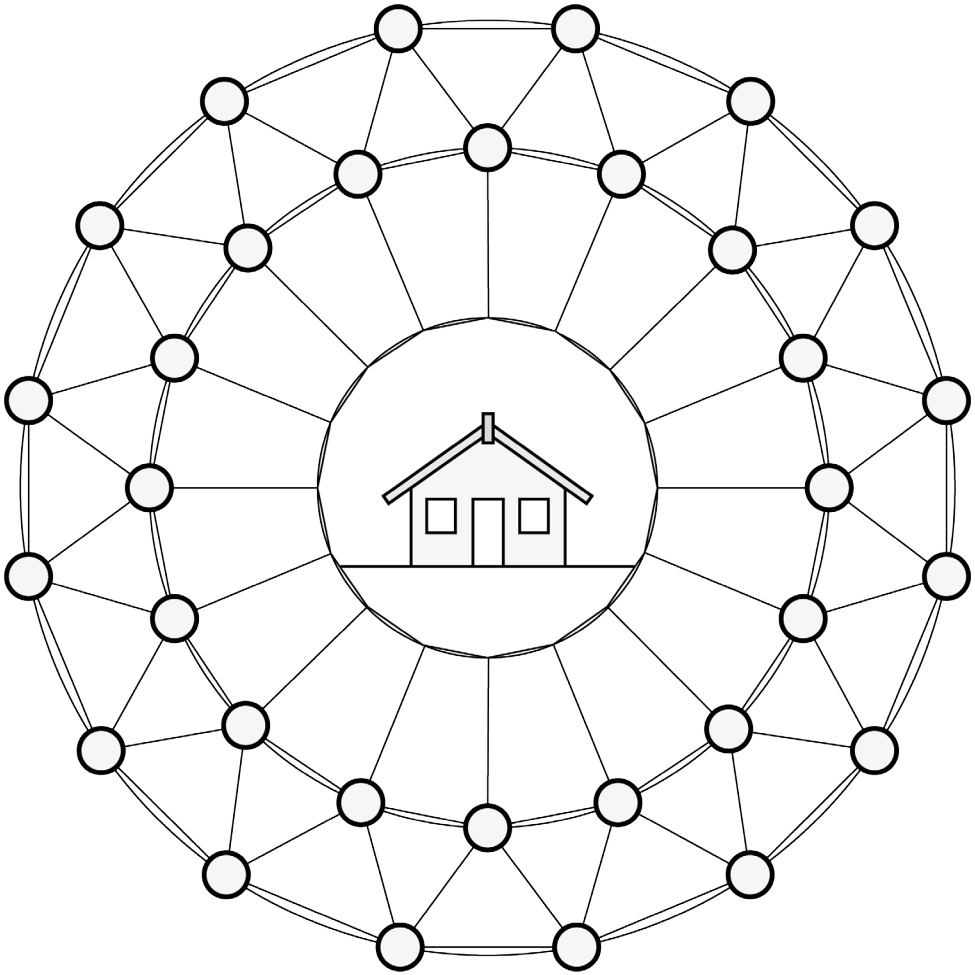
Figure 16:
Communal Relationship Structuring
This figure identifies some of the more significant relationships that actively shape Māori community spatial organisation. These relationships include family based connections, iwi based connections, generalised family and housing type connections etc.



MĀORI RELATIONSHIP STRUCTURES

As previously stated, the relationship structures with which we are working are exceedingly complex as they consist of several intertwined factors, selectively layering and combining to create a web of interaction that requires breaking down in order to understand. The subjects with whom these relationships form are not constant; rather they operate on the individual, familial, and tribal level, with many different motivations creating many different relationships with varying levels of importance at each level (Mead, 2016, pp. 163 - 178). These have been expressed in an expanded diagrammatic form for clarity; considering the iwi relationships, familial relationships, and individual relationships as separate motivations, when in reality they work in conjunction to form a complex decision making matrix.

Figure 17: Iwi Relationships
Diagrammatic interpretation of the relationships formed as a result of iwi dynamics. All members form relationships with each other through a mutual tribal connection. This involves shared histories, shared tipuna, and on a sub-tribal level, shared marae and papakainga (Mead, 2016, pp. 175).



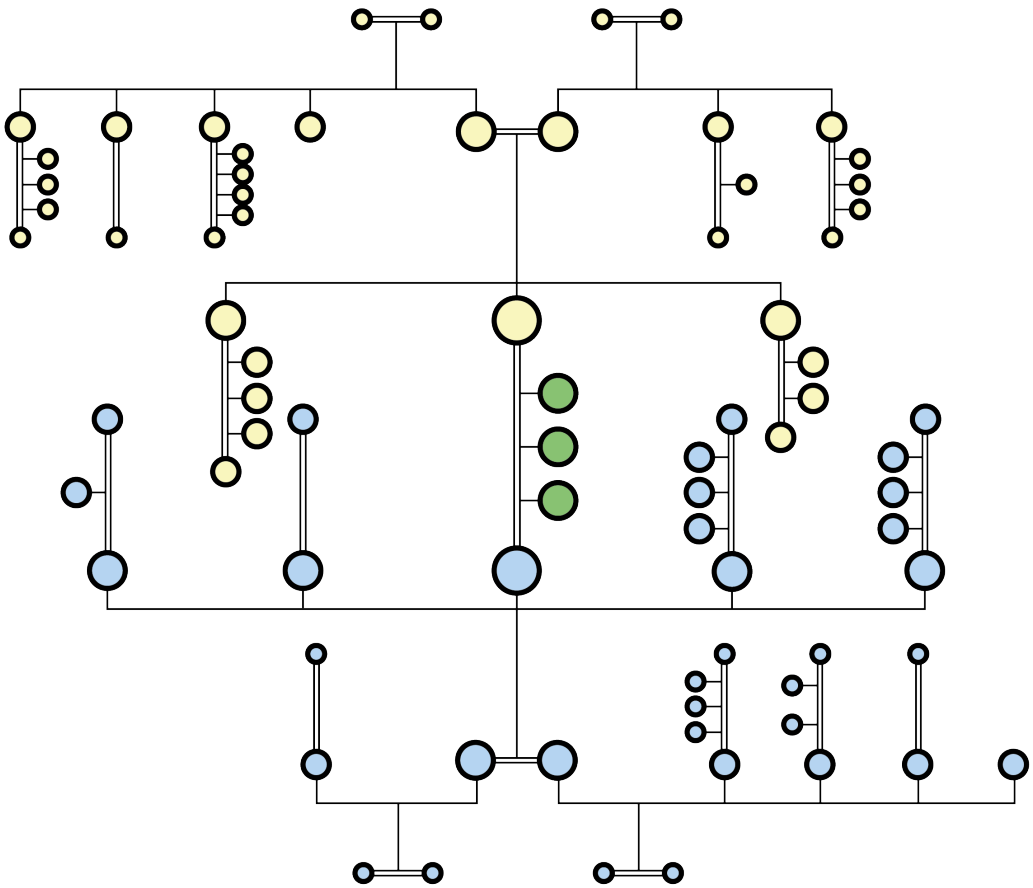


Figure 18: Familial Relationships
Diagrammatic interpretation of the relationships formed as a result of whanau dynamics. Whanau extends beyond that of immediate family, through to include the different levels of removed, extended and in law familial relationships. This diagram demonstrates how the union of two people form connections between many. (Mead, 2016, pp. 165)

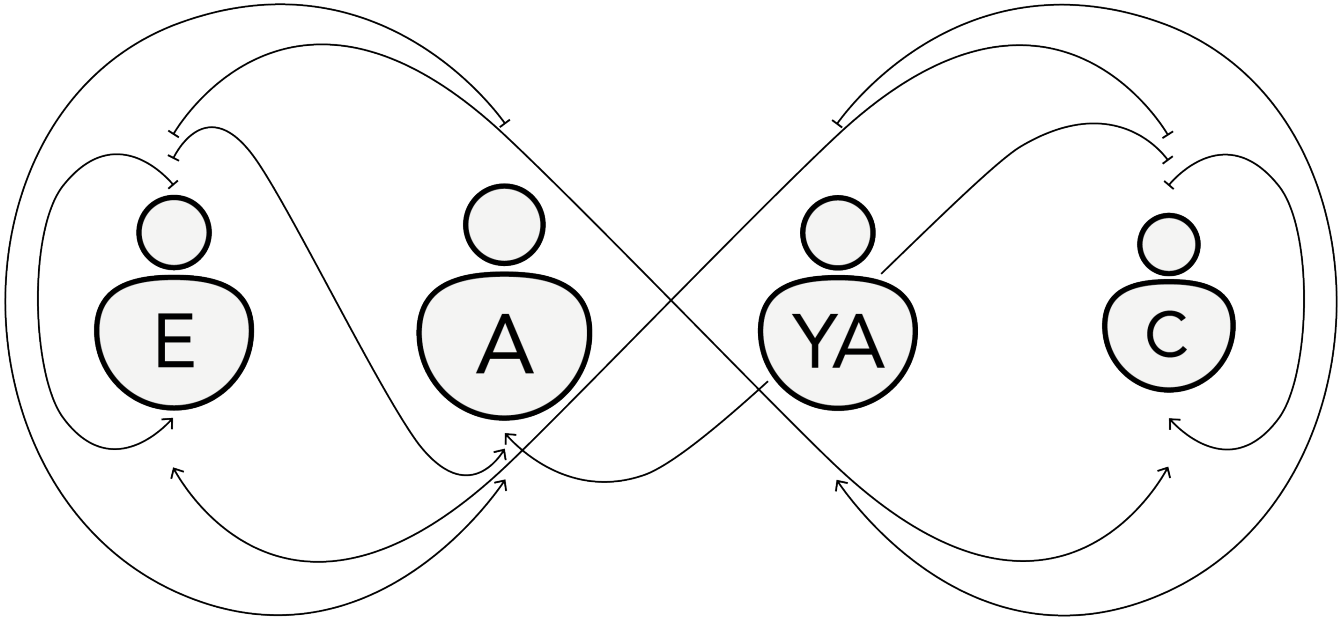


Figure 19: Individual Relationships (E = Elderly, A = Adults, YA = Young Adults, C = Children)
Diagrammatic interpretation of the relationships formed at an individual level. In scenarios where multi-generational living is so abundant, relationships formed based upon generational groupings become highly influential to the Māori way of life. An example from the diagram above can isolate the relationships children form with other generational groups; they gravitate towards their own generation for companionship, as well as the adult and elderly generations for different forms of care and support as well as respect of status within the iwi, however they are not so reliant on relationships formed with young adults (Mead, 2016, pp. 31, 32).

CURRENT ARCHITECTURAL MANIFESTATIONS

These relationship structures have found a method of coexisting through architecture in current rural papakainga, and the question is how designers can find a way for them to operate in the urban setting.

In rural papakainga the personal relationships that respect multi-generational living and familial obligation are reflected in the larger houses, with multi-functioning, adaptable spaces, to suit scenarios where people may house siblings, parents or grandchildren on a temporary or permanent basis. The familial relationships that result in large gatherings have a marae for support, the co-owned land results in a base for family to return to, and the tribally formed relationships are what unite the entire papakainga settlement.

These types of architectural manifestations are one method of creating an environment in which culturally motivated relationships can form; the rural method. The next step is to find what architectural interventions can help establish these relationships in an urban setting.

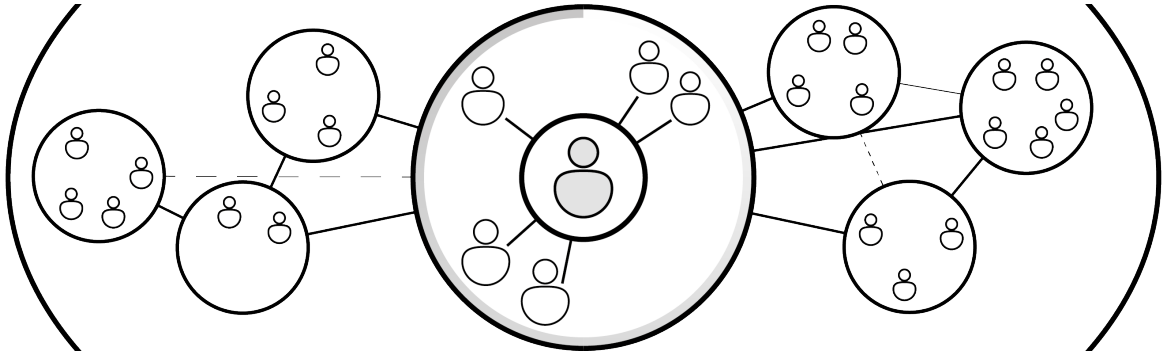


Figure 20: Unified Relationship Structuring
Relationships between iwi, whanau and individuals work together to create a matrix system that enables the success of papakainga living environments.

The underlying issue that has emerged can be simply put;

How can architecture influence certain desired relationships within a community?

We know which relationships we need to emulate, based on the relationships that form within rural papakainga that stem from the core values of the Māori culture, and we know these values are what are missing from current attempts at urbanising papakainga systems. Therefore, if we can find a way to introduce these desired conditions within the urban setting, we will be introducing the essence of the papakainga that has been otherwise lacking in the urban environment.

The strategy with which this is approached needs to address the problem directly. It has been proven through previous urban papakainga attempts that traditional western design methodology does not understand or allow for the issues at hand (Appendix Item 2), therefore we need to approach it in a more direct manner. In order to design architecture to inform relationships, why not design the relationships themselves, and build the architecture around them?

In a system where the priority is understanding how a person fits into a family, within a community, within the wider context, and the secondary point of focus is how architecture then fits within that model, we are no longer working with the traditional tools of modelling form and space, we are modelling relationships, and the tools of design become those that can predict and simulate behavioural properties on a communal scale, the most applicable being that of multi-agent system modelling.

Part I - Background

CHAPTER THREE:

MULTI-AGENT SYSTEMS

Multi-agent systems spawned from the study of Distributed Artificial Intelligence in the mid 1970's, to create 'systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks' (Weiss, 1991, pp. 1). In other words, an intelligent entity with scripted behaviours – the agent – simulates posed scenarios, as an individual within a system of other programmed agents.

The intelligent aspect of the system is based on the interaction of agents within it, either with each other, or with external factors. The agents react to proposed scenarios dependent on scripted behavioural responses, and by prioritising those responses agents use logical reasoning to perform the most appropriate reaction. Agents operate on a local network, communicating with other agents in the system, and with the surrounding environment, therefore the behaviours programmed into the agents are essentially based on communal relationship structures, providing an excellent tool for simulating behavioural relationships within a community, such as that of a cultural persuasion (Daniel *et al*, 2016, pp. 81).

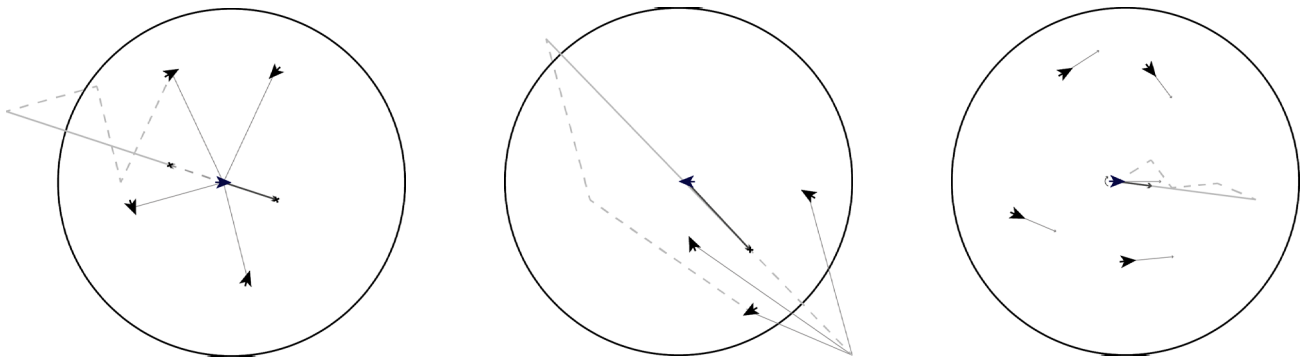


Figure 21, 22 and 23 (L to R): Flocking Examples
Diagrams depicting the logic behind Reynolds' principles Separation, Alignment, and Cohesion (Opp.).

REYNOLDS' PRINCIPLES OF FLOCKING

In 1987, Reynolds wrote a paper proposing the revolutionary concept of flocking; that is a notion that gives coders the ability to treat agents as a collective, with common motivations and behaviours, yet individual responses. Previously agents had been programmed as individuals, or as a group that responded in an identical manner. By this logic however, agents can be assigned the same motivations as each other, but will respond based on individual scenarios, offering a revolutionary degree of independence. This type of system in its purest form is most often compared to that of a flock of birds; the group behaves as a collective, with overarching synchronised behavioural systems, however within the system are individuals, each responding to the slight nuances of other individuals within the group. (Reynolds, 1987)
The key aspects of Reynolds model form the behaviours on which agents operate, and these are as follows;

Separation, or collision avoidance, operates under the logic that an agent with mass cannot occupy the same space as another.
Alignment, or velocity matching, addresses uniformity with the group, dictating the synchronisation of those in the system.
Cohesion, or flock centring, is the degree to which the group is drawn together, whether the group is relatively closed, or open and independent.

These rules can form the basis of any behavioural system, simple or complex, and operates on a hierarchical system of layered behaviours, providing the base for mapping Māori relationships in the urban environment.

Images redacted for copyright reasons

Figures 24, 25, 26 (Top to Bottom): Flocking Uses
24. Batman Returns bat breakdown.
25. Example of flock Traffic Navigation.
26. Work of Kokkugia.

CURRENT USES AND FUNCTIONALITY

In order to understand how flocking principles, with three relatively simple rules, can emulate highly complex behavioural systems, it is beneficial to see how it is doing so in current practice. Flocking systems have been used across many disciplines; with uses in the film industry to simulate wildebeest stampedes in Lion King, and bats flocking in Batman Returns, to being vital components in defence systems, data visualisation, multichannel radio programming, and providing vital improvements to road safety through flock traffic navigation systems, multi-agency has been implemented very successfully in a vast array of fields (Resnick, 1994).

Multi-agent systems have also been implemented within the architectural world, largely for exploratory organic form making. Snooks and Stuart-Smith use multi-agent systems in the 'development of a behavioural design process' with "Kokkugia", and Francois Roche explores 'research as speculation' with "New-Territories", both practices being forerunners in the incorporation of these behavioural simulation systems into the design process (Kokkugia.com, 2016. Roche, 2016).

BEHAVIOURAL PROPERTIES

The following examples indicate the power this type of tool has in creating a wide range of simulation variables, and how proper employment could create highly complex systems for simulating scenarios that are not yet fully understood. The first example is formed on a baseless logic, indicating they can emulate any type of desired system, and the second example demonstrates the ability to simulate very real scenarios, for example that of the behaviours of a curtain.

Images redacted for copyright reasons

Figure 27, 28a, b (Top to Bottom):
27. Swarm Flocking example, adapted from Attraction/Repulsion OpenProcessing sketch by Henderson. Video included as Appendix Item 3. (Henderson, 2011)
28a. Code for Curtain. (Fabric Simulator)
28b. Curtain (Fabric Simulator) example, OpenProcessing sketch by BlueThen. Video included as Appendix Item 4. (BlueThen, 2011)

Part II - Testing

CHAPTER FOUR:

PROCESSING

Processing is an open-source coding program that provides the vehicle for the following multi-agent explorations. Some view the program as a primitive coding environment, with no specific use or pre-scripted functions, however the flexibility and transparency it offers provides a great vehicle for exploring new ideas such as the programming of cultural behaviours. The accessibility and open-source nature of the software is of great benefit for furthering knowledge and sharing information, ensuring research findings can be implemented without prior expertise in coding.

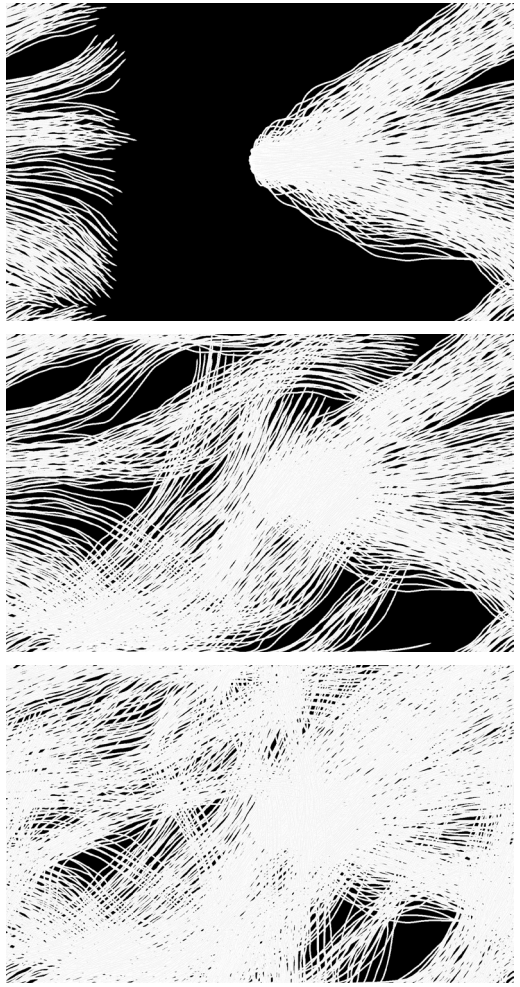
Behavioural functions are converted to if/then/else statements and input into the program using JavaScript coding, resulting in simulations imitating the relationships identified as being vital within Māori living environments. These simulations are re-run with additional functions until a relationship matrix has been created that simulates real life, and this provides an insight into how a specific Māori community could be organised to ensure success in providing a culturally appropriate environment in terms of relationship structuring.

Figure 29, 30a - c (L to R, Top to Bottom): Rain without Gravity

29. Code creating the Rain sketch.

30a, b, c. Different stages of the Rain simulation, showing Reynolds' principles of flocking in a very organic replication of rain forming, with no addition of gravitational force.

```
Flock flock;

void setup() {
  size(1280, 760);
  background(0);
  flock = new Flock();
  for (int i = 0; i < 500; i++) {
    flock.addBoid(new Boid(random(0, width),random(0, height)));
  }
}

void draw() {
  flock.run();
  fill(0, 6);
  rect(0, 0, width, height);
}


class Flock {
  ArrayList<Boid> boids;

  Flock() {
    boids = new ArrayList<Boid>();
  }

  void run() {
    for (Boid b : boids) {
      b.run(boids);
    }
  }

  void addBoid(Boid b) {
    boids.add(b);
  }

}



class Boid {

  PVector location;
  PVector velocity;
  PVector acceleration;
  float r;
  float maxforce;
  float maxspeed;

    Boid(float x, float y) {
    acceleration = new PVector(0, 0);

    float angle = random(TWO_PI);
    velocity = new PVector(cos(angle), sin(angle));

    location = new PVector(x, y);
    r = 2.0;
    maxspeed = 2;
    maxforce = 0.03;
  }

  void run(ArrayList<Boid> boids) {
```



```
Plane plane;

void setup () {
  background (255);
  size (1000, 800);
  plane = new Plane();
  for (int i = 0; i<250; i++) {
    plane.addAgent(new Agent(500, 40
  }
}

void draw () {
  plane.run();
}

class Plane {
  ArrayList<Agent> agents;

  Plane() {
    agents = new ArrayList<Agent>();
  }

  void run() {
    for (Agent b : agents) {
      b.run(agents);
    }
  }

  void addAgent(Agent b) {
    agents.add(b);
  }
}

class Agent {

  PVector loc;
  PVector vel;
  PVector acc;
  float r;
  float maxforce;
  float maxspeed;

  Agent(float x, float y) {
    acc = new PVector(0, 0);
```

```
    vel = PVector.random2D();
    loc = new PVector(x, y);
    r = 2;
    maxspeed = 2;
    maxforce = 1;
  }

  void run(ArrayList<Agent> agents) {
    plane(agents);
    update();
    borders();
    render();
  }

  void applyForce(PVector force) {
    acc.add(force);
  }

  void plane(ArrayList<Age
    PVector sep = separate
    PVector ali = align(ag
    PVector coh = cohesion
    sep.mult(0.15);
    ali.mult(1.0);
    coh.mult(0.1);
    applyForce(sep);
    applyForce(ali);
    applyForce(coh);
    smooth();
  }

  void update() {
    vel.add(acc);
    vel.limit(maxspeed);
    loc.add(vel);
    acc.mult(0);
  }

  PVector seek(PVector target) {
    PVector desired = PVector.sub(tar
    desired.normalize();
    desired.mult(maxspeed);

    PVector steer = PVector.sub(desir
    steer.limit(maxforce);
```

```
    return steer;
  }

  void render() {
    strokeWeight(0.5);
    stroke(120);
    point(loc.x, loc.y);
  }

  void borders() {
    if (loc.x < -r) loc.x = width+r;
    if (loc.y < -r) loc.y = height+r;
    if (loc.x > width+r) loc.x = -r;
    if (loc.y > height+r) loc.y = -r;
  }

  PVector separate (ArrayList<Agent> ag
    eparation = 25.0f;
    = new PVector(0, 0, (

    er : agents) {
    ector.dist(loc, other
    && (d < desiredsepara
    ff = PVector.sub(loc,
    lize();
    );
      steer.add(diff);
      count++;
    }
  }

  if (count > 0) {
    steer.div((float)count);
  }

  if (steer.mag() > 0) {
    steer.normalize();
    steer.mult(maxspeed);
    steer.sub(vel);
    steer.limit(maxforce);
  }
  return steer;
}
```

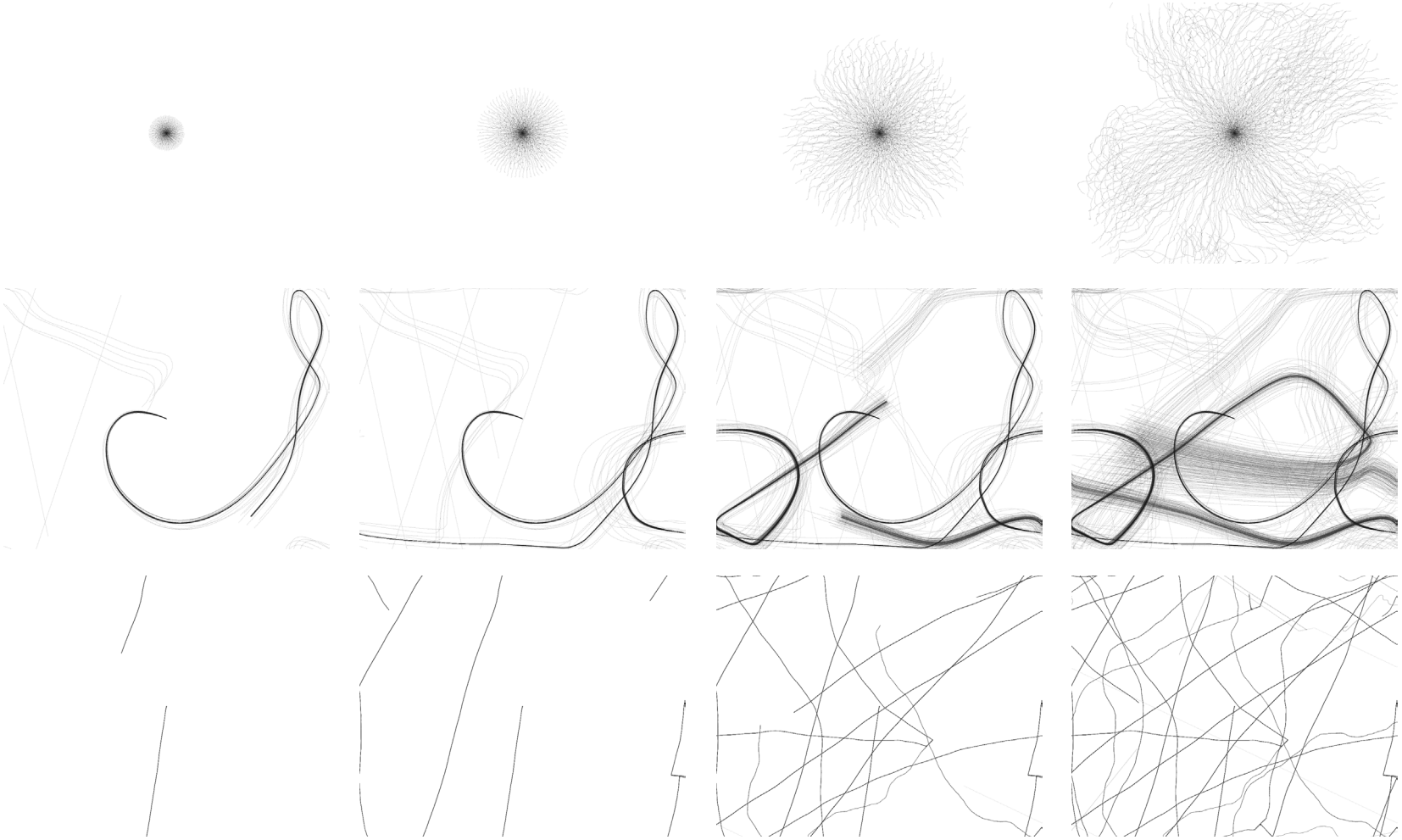
```
void keyPressed() {

  if (key=='s') {
    saveFrame();
  }
}
```

```
  PVector
    float
    PVecto
    int co
    for (A
      floa
      if (
        su
        co
      }
    }
    if (co
      sum.
      sum.
      sum.
      PVec
      stee
      retu
    } else
      retu
    }
  }

  PVector
    float
    PVecto
    int co
    for (A
      floa
      if (
        su
        co
      }
    }
    if (co
      sum.
      retu
    } else
      retu
    }
  }
}
```

```
 align (ArrayList<Agent> agents) {
 neighbordist = 50;
or sum = new PVector(0, 0);
ount = 0;
Agent other : agents) {
at d = PVector.dist(loc, other.loc);
((d > 0) && (d < neighbordist)) {
um.add(other.vel);
ount++;


ount > 0) {
.div((float)count);
.normalize();
.mult(maxspeed);
ctor steer = PVector.sub(sum, vel);
er.limit(maxforce);
urn steer;
e {
urn new PVector(0, 0);


 cohesion (ArrayList<Agent> agents) {
 neighbordist = 50;
or sum = new PVector(0, 0);
ount = 0;
Agent other : agents) {
at d = PVector.dist(loc, other.loc);
((d > 0) && (d < neighbordist)) {
um.add(other.loc);
ount++;


ount > 0) {
.div(count);
urn seek(sum);
e {
urn new PVector(0, 0);
```

Figures 31: Exploring Reynolds' Principles Through Code
Code exploring how to manipulate Reynolds' principles to work alone, and in conjunction with each other. Understanding the full scope of these principles allows for a system that can be manipulated to its complete extent, providing the most accuracy and consistency possible.

This image also provides the basis of the cover and chapter heading images. Coding is used to create marvellous simulations that completely obscure the method of creation, and this acts as homage to the source that enables these products.

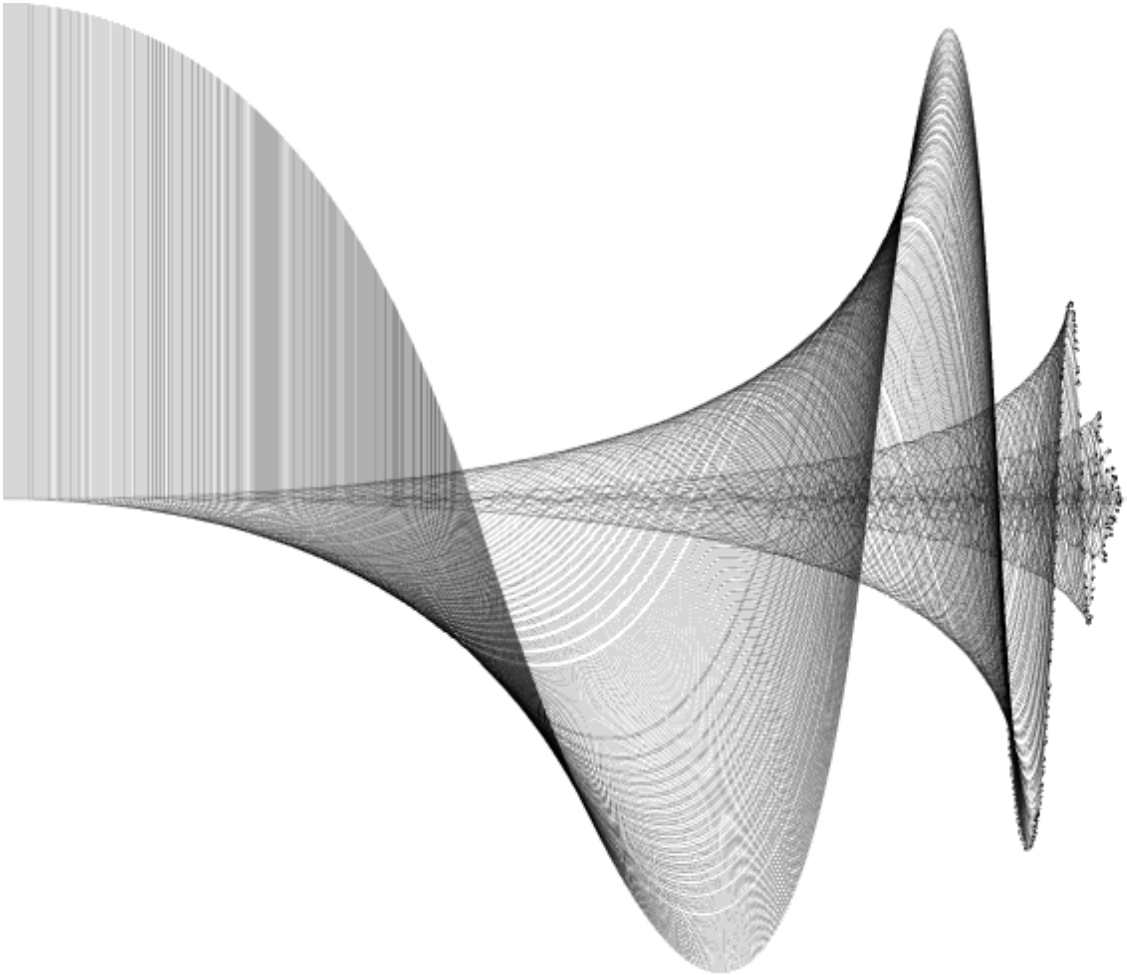


ISOLATED BEHAVIOURAL MODELLING

In order to simulate behavioural functions using Reynolds' principles of flocking, relationships need to be converted into measures of separation, alignment and cohesion. As individual functions these are relatively simple, with examples opposite.

The example of separation works with the idea of the territorial affiliation an iwi has with land, and the respect someone from another iwi would have for that pre-established bond. The agents have been instructed to wander randomly through the environment, but have also been told that an area occupied by another agent is deemed unapproachable, as that space is already associated with another tribe, and accordingly they disperse uniformly.
The example of alignment demonstrates a rather abstract concept in order to exemplify how encompassing the rules of flocking can be. It demonstrates shared motivations. The agents within this model are driven by the same behaviour, regardless of the goal. This shows us how agents within this system can be ordered to conform to a desired degree.
The example of cohesion gives a visual representation to the idea of cultural gathering. In this case the agents have been told they are Māori, and that they wish to associate with other Māori, therefore they are constantly analysing the agents around them looking for others sharing that cultural specification, then forming groups and sub-groups.

Figure 32a - d, 33a - d, 34a - d (Opp., L to R, Top to Bottom): Working with Separation, Alignment, and Cohesion - Appendix Item 5.
The extremes of Reynolds' principles in isolation. Separation models disperse from an initial point of concentration, alignment models maintain common behaviours above all else, and cohesion shows common goals based on individual scenarios, occasionally aligning.



COMBINED BEHAVIOURAL MODELLING

Combining the rules of flocking requires relating each rule to the others, and establishing a hierarchical system so as not to create conflicting behavioural scenarios. This can prove to be far more complex than establishing the behaviours in isolation, as many of the relationships within the system will have some effect on another, and while some may support each other, others will inevitably clash, meaning it is vital to establish a means in which the agents will know their behavioural priorities, so as to ensure they behave in the most appropriate manner, rather than attempting to act out all functions at once.

This dictates that each rule shall have to be individually added to the system, evaluated, modified, and perfected before the next layer is integrated; ensuring the system is fully functioning every step of the way and helping isolate scenarios that conflict within the system, establishing which relationships are better suited to the urban environment, and which may struggle to be fully integrated without environmental adjustments.

Figure 35 (Opp.): Working with Separation, Alignment, and Cohesion in Unison
Combining the three principles is shown to be more than layering, rather a discussion between the three must be established, or situations occur where the agents respond entirely unexpectedly, rather than all navigating to the right, this scenario caused a wave effect rather that all agents moving right in a straight line as desired.

Part II - Testing

CHAPTER FIVE:

2D SIMULATION

In order to test the layering of behavioural relationships in an initial simplified manner the first phase of testing has been carried out in two dimensional form, working with linear and spatial examples. This type of testing is the architectural equivalent to the planning phase of a design, or the mapping phase of a greater scale project, before the elevations and perspective drawings bring it into a three dimensional form.

This phase is working under the assumption of a type of planar living environment, which is a true representation of current relationship structuring if you consider the unelevated nature of rural Māori papakainga. A third dimension shall be incorporated when the first stage has adequately captured current living scenarios, at which point the three dimensional nature of the urban environment begins to impose an influence on the formation of relationship simulations.

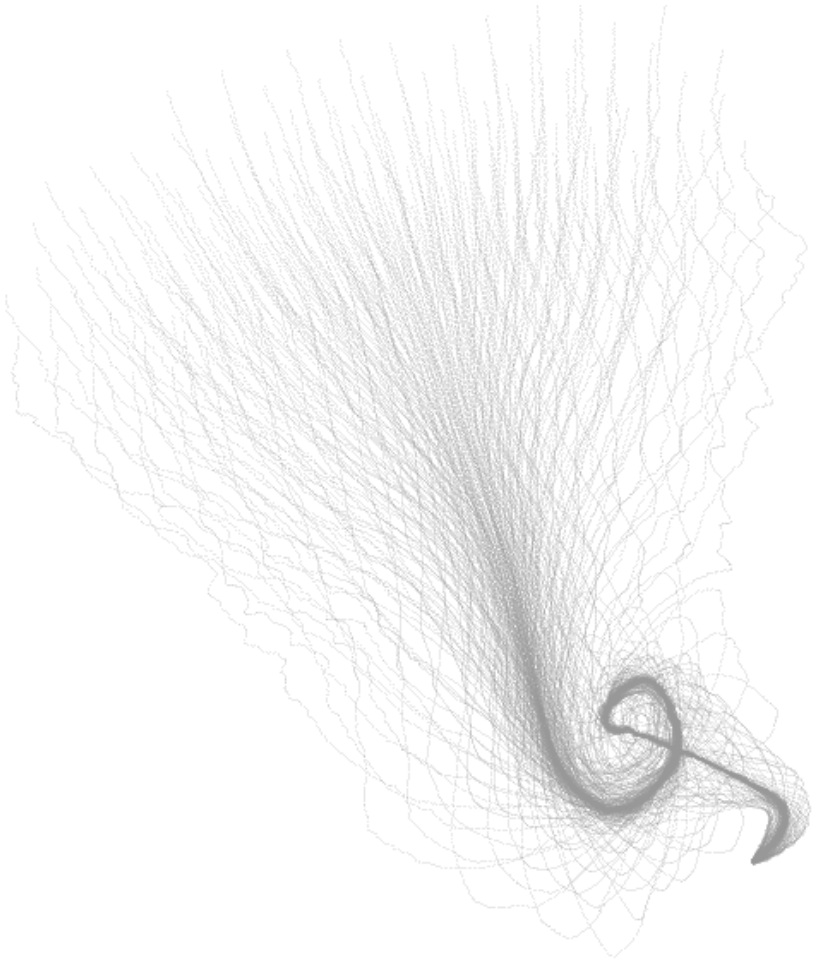
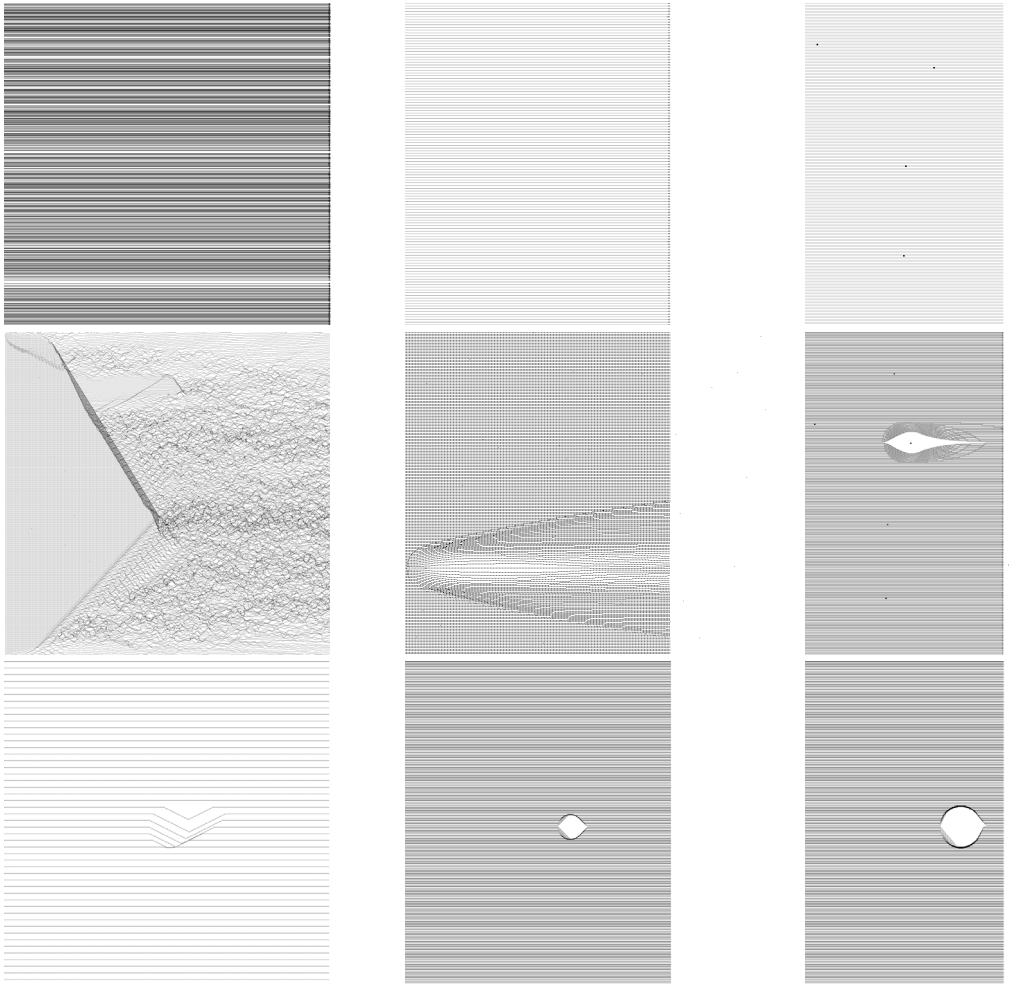


Figure 36: Reynolds' Principles Unified Harmoniously
Example of the three principles in Figure 32 - 34 united correctly to create a system where units emerge from the same location with equal degrees of separation, alignment, and cohesion. This type of simulation is created by simple JavaScript coding, with very few complexities, however produces very organic creations. Video included as Appendix Item 6.

SINGULAR FACTORS

Working with initial singular factors of relationship structuring identifies visualisation methods best suited to the investigation at hand.
The initial use of linear agent modelling is excellent for identifying the absolutes of positive and negative space, however it has been identified as requiring multiple layering of data when simulating the different factors related to a single relationship category, and would require exponentially more when researching more than one category. Despite this, linear data has been used to create a mapping system for potential areas of papakainga development that respect the Māori territorial affiliation with whenua as described by Mead, where Māori hold deep value in the rightful occupiers of land (2016, pp. 219 - 229). Through the testing phase it incidentally emerged that this type of system could prove very useful for identifying potential areas of settlement for groups formed from one or more iwi in any area of the country, therefore the system, method of use and alterations are attached as Appendix Item 7.
The later spatial simulations proved far more useful when considering the desired outcome in terms of locating housing units to establish desired relationship systems. This type of simulation utilises agents as units describing one dwelling, therefore one agent is responsible for a collective of relationships, and is motivated to find the most appropriate relationship structure that satisfies the family group that would occupy the dwelling.

Figure 37a - i (L to R): Mapping Iwi Relationships
Initial explorations investigated iwi relationships in their most simplified form, that of the territorial affiliation iwi have with their land. The system operates under the scenario that an iwi is looking to establish a new settlement, and in respect to other iwi, can only settle in unoccupied, and unaffiliated land areas. These images detail the exploration into manipulating agents to avoid certain areas of the system.



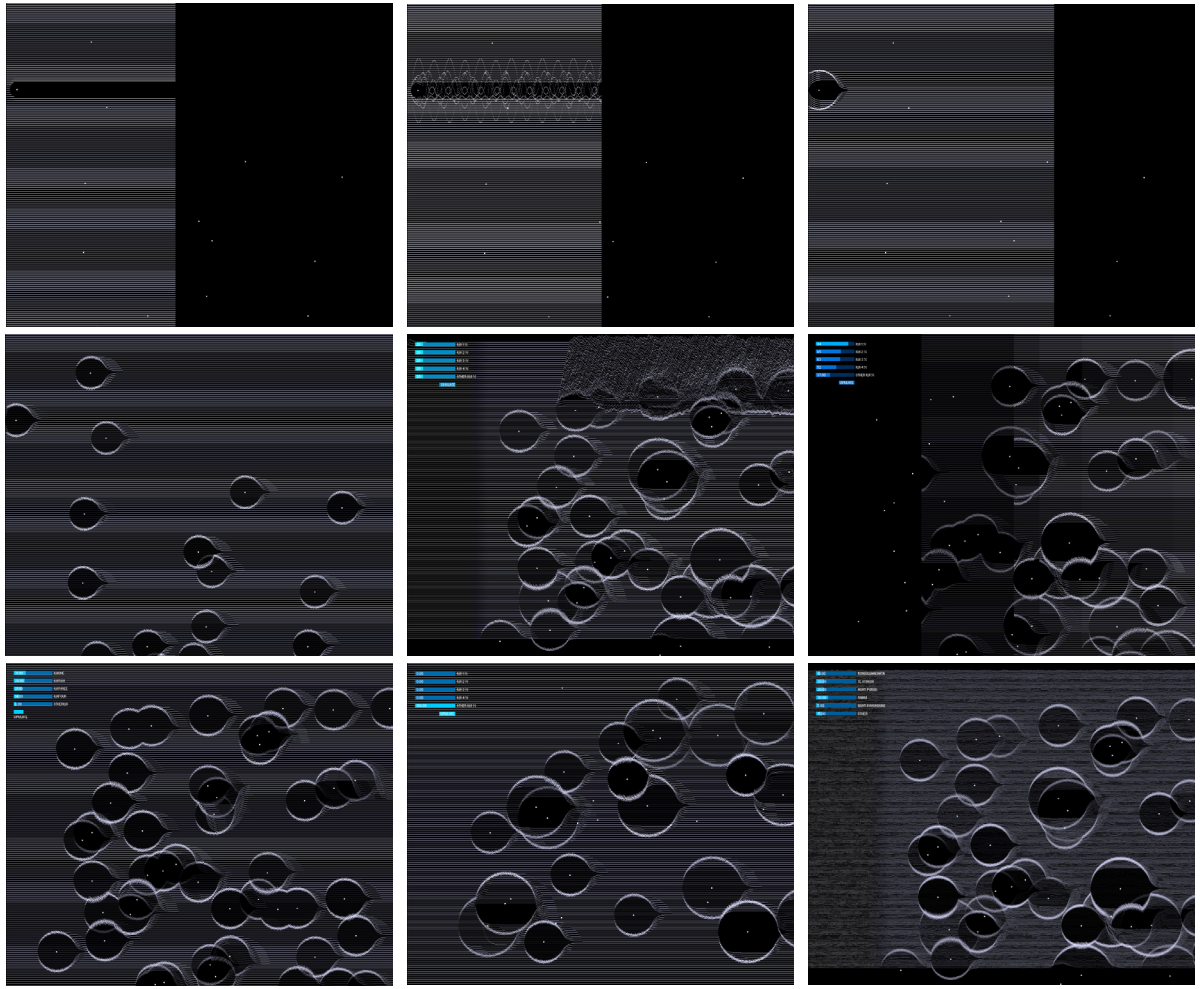
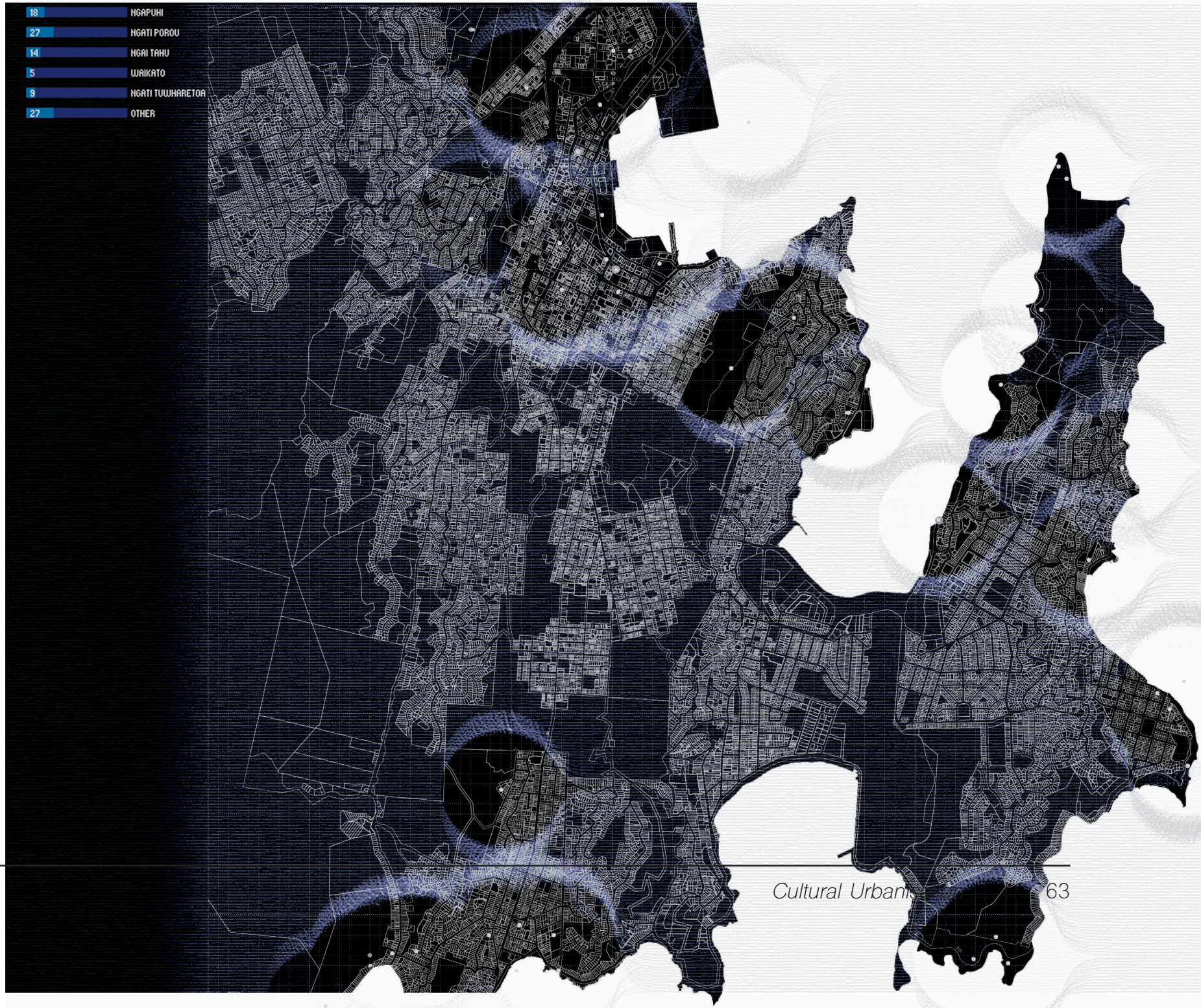


Figure 38a - i (L to R, Top to Bottom) and 39 (Opp.): Mapping Multiple Iwi Relationships
Previous simulations in Fig 37 have been refined into a system, using location data of where iwi have, or have had affiliations to establish a methodology of locating where new settlements can be developed. Working in the Wellington region, with a settlement group composed of 5 iwi, this system indicates through agent movement where settlements should not be established as indicated by where agents deviate from their path, leaving the remainder of the area open for development. This system is explained and the script included as Appendix Item 7.

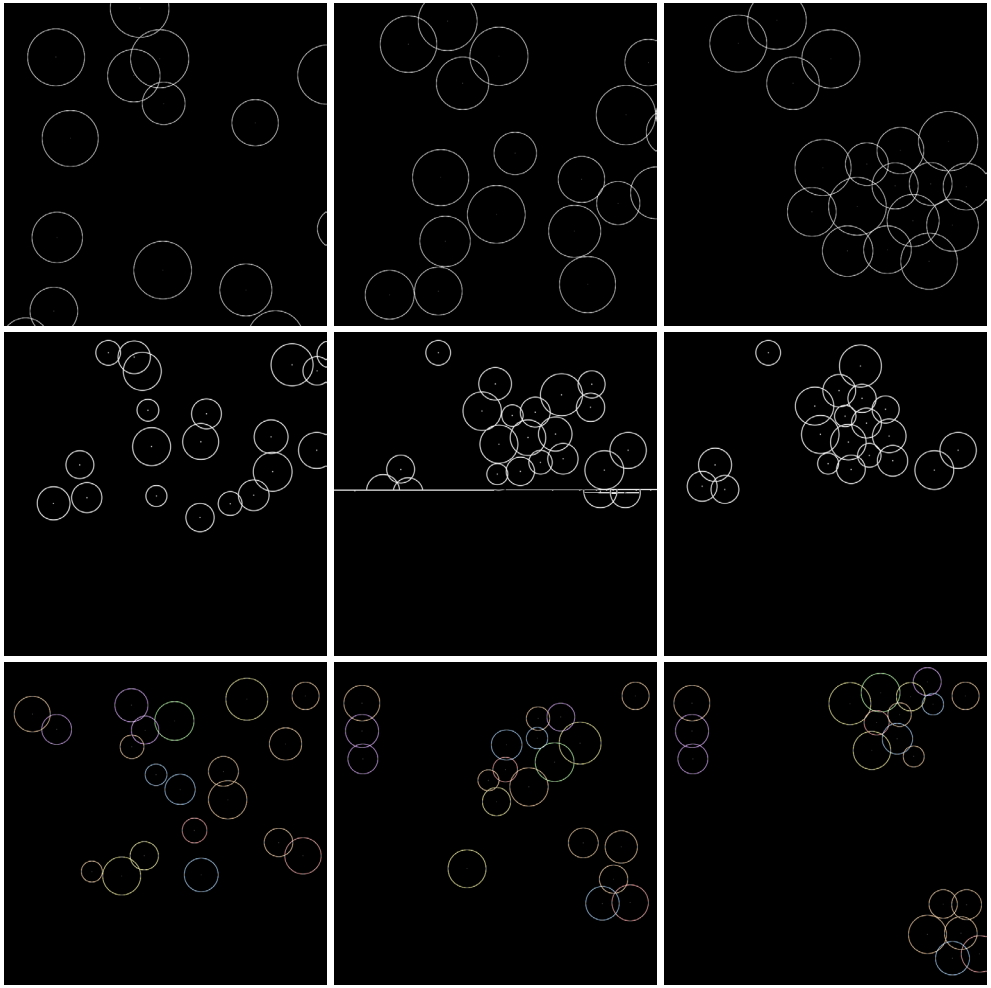

18 NGAPUHI
27 NGATI POROU
14 NGAI TAHU
5 WAIKATO
9 NGATI TUWHARETOA
27 OTHER

MULTIPLE FACTORS

Introducing multiple factors begins to simulate real scenarios in relation to Māori cultural behaviours. By layering relationships based on iwi, family and individual motivations we can see how the system adapts to cater to these additional forces that may influence spatial structure. In some a singular addition can completely rearrange the entire system, and in other cases motivations may have very little effect as they simply work in conjunction with previous functions.

Figure 40a - i (L to R, Top to Bottom): First Phase of Spatial Mapping
Initially using a system based on separation of units for personal housing, as well as cohesion of units based on shared iwi (40a - f), and later cohesion for shared family and housing types (40g - i), these units perform a basic series of goal seeking objectives in order to establish locations.

Investigations using family and housing types are a factor that were incorporated in this Part II Testing phase, however eliminated in Part III in favour of individual family breakdowns, as added data proved less complex to input than expected, and far more accurate. This research has therefore been located as Appendix Item 8 rather than included in the Part I research.
(Fitzgerald et al, 2006, pp. 23, 51-56, 120. Fabian and Goodyear, 2014, pp. 66-67, 80, Joynt et al., 2016, pp. 16. Dupuis and Lysnar, 2014, pp. 47-57, 60-61).



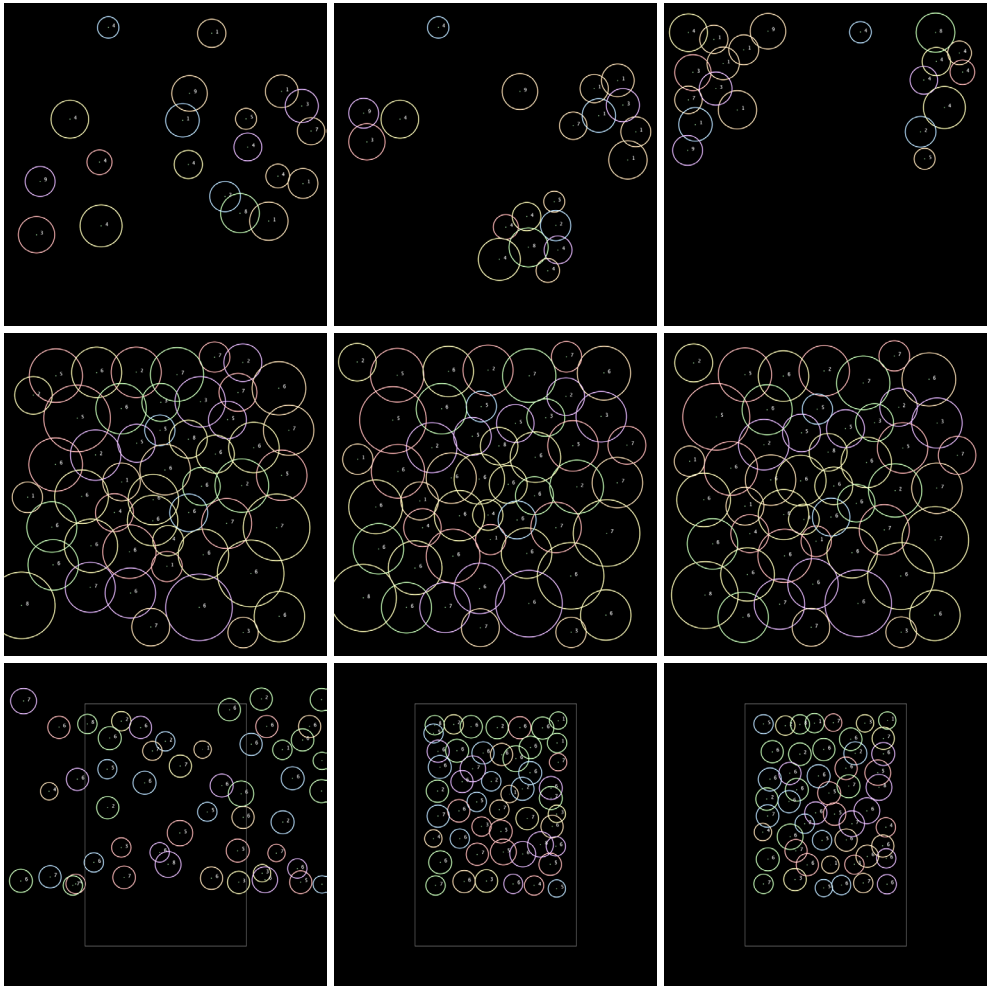
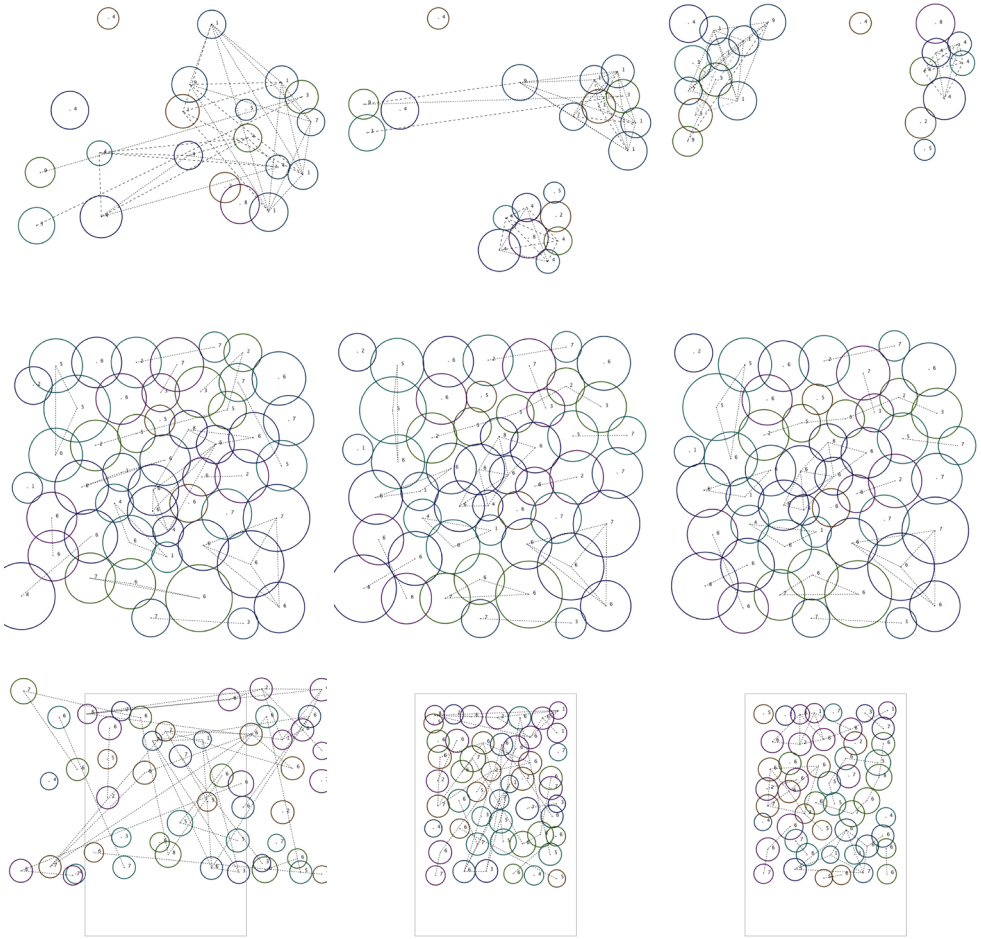


Figure 41a - i (L to R, Top to Bottom): Second Phase of Spatial Mapping
This phase of mapping investigates a stronger grouping attraction between iwi, family, and housing types (41a - c), as well as additional units to investigate if goals will still be followed when working within tighter confines (41d - i). This phase demonstrates that the system can produce outputs for sparse or more contained environments, depending on the amount of space assigned to units. a video simulation of this process is included as Appendix Item 9.

Figure 42a - i (L to R, Top to Bottom): Identifying Relationships in Second Phase of Mapping

Identifying how some of the relationships in phase two have manifested spatially. In each set of 3 images, a - c, d - f, g - i, the initial iwi, family, and housing type relationships have been simulated, and snapshots indicated how these relationships motivate the agents to group together.



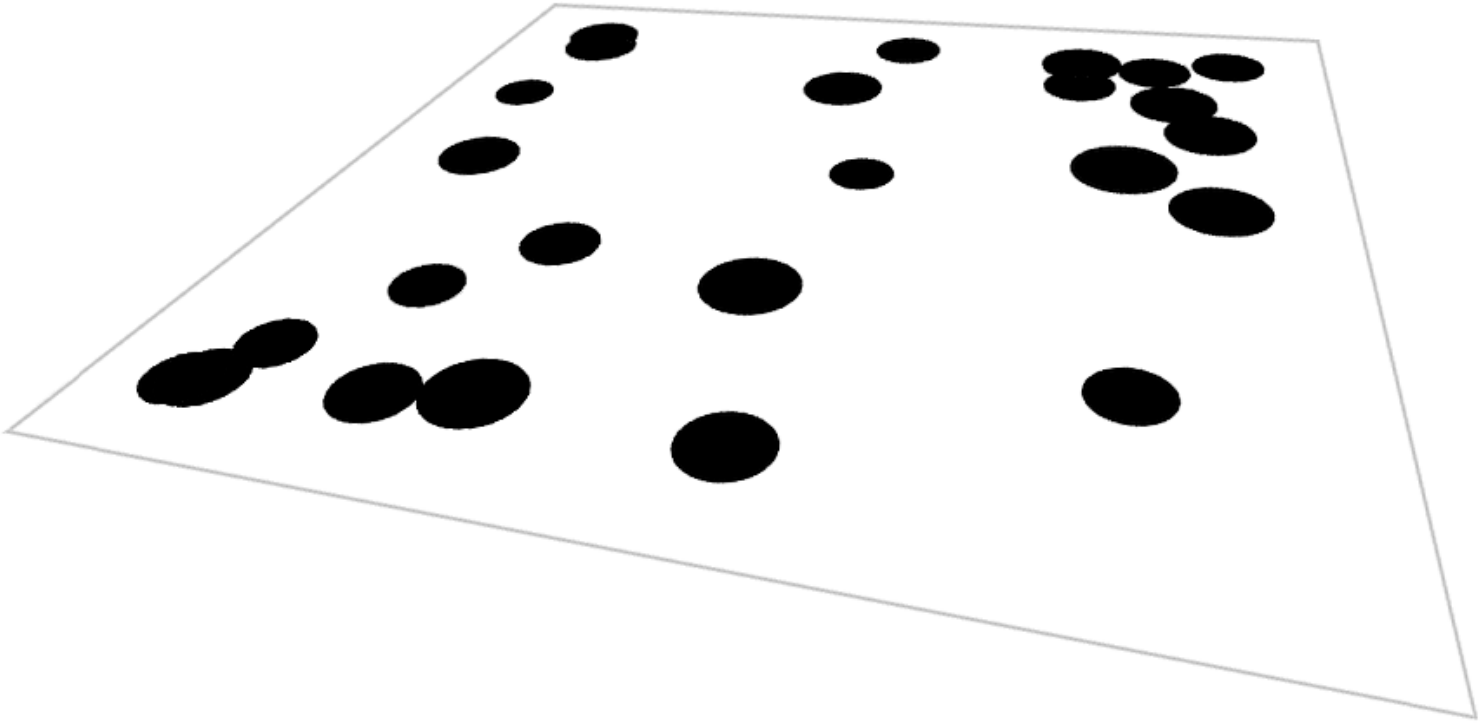
Part II - Testing

CHAPTER SIX:

3D SIMULATION

This chapter describes translating the system into a three dimensional form, where spatial occupation based on desired relationship structures becomes a visual reality. This step is essentially the purpose of the research, finding a way to simulate how dwellings should be organised within an urban setting in order to produce relationship structuring based on those present within a rural papakainga setting.
This step illustrates how these relationships have been added to the system step by step, indicating a method in which this could be replicated endlessly, building any type of behavioural matrix based on any system of relationship structuring desired. This step is what makes the system an amalgamation of the Māori culture and the urban setting, yet with different constraints it could reflect any culture, or any setting, with as many or as little behaviours as desired to create a well structured, or very flexible visual interpretation of spatial relationships.

Figure 43 (Opp.): Current Output Simulation
The stage at which two dimensional investigations are abandoned, having informed the design process enough that a third dimension of investigation can be added.



x	y	z	family type	iwi type	house type
91.54065	905.822	907.39355	1	1	1
301.07104	113.5101	605.0662	1	1	2
414.319	586.4602	397.83786	1	2	1
325.77652	357.37732	812.05054	2	4	3
104.641	378.50943	102.47642	2	3	2
242.44077	803.8966	680.7546	2	5	3
770.05664	117.61539	515.2142	2	3	4
694.522	279.53903	407.28482	2	3	2
898.98364	391.80154	352.56256	3	1	2
419.0358	836.0534	92.58435	3	6	1
325.66803	881.0512	496.10623	4	5	3
747.49554	753.81934	392.00195	4	2	2
382.8429	272.4868	643.7531	5	1	1
284.17307	577.7838	748.2954	5	4	5
547.9611	238.16003	671.4652	5	5	4
349.1543	681.19916	164.14627	5	5	2
687.0601	276.7534	601.461	5	5	2
713.8387	876.0749	553.94745	6	2	3
603.7226	332.01877	256.7607	6	2	1
565.3575	710.2684	874.25684	6	1	4
439.02792	434.2629	273.50366	6	1	3
879.78186	115.55287	887.69977	6	6	3
463.81488	292.82822	452.43094	0	0	0
531.5377	893.20074	450.26184	0	0	0
519.9657	85.44295	450.3925	0	0	0

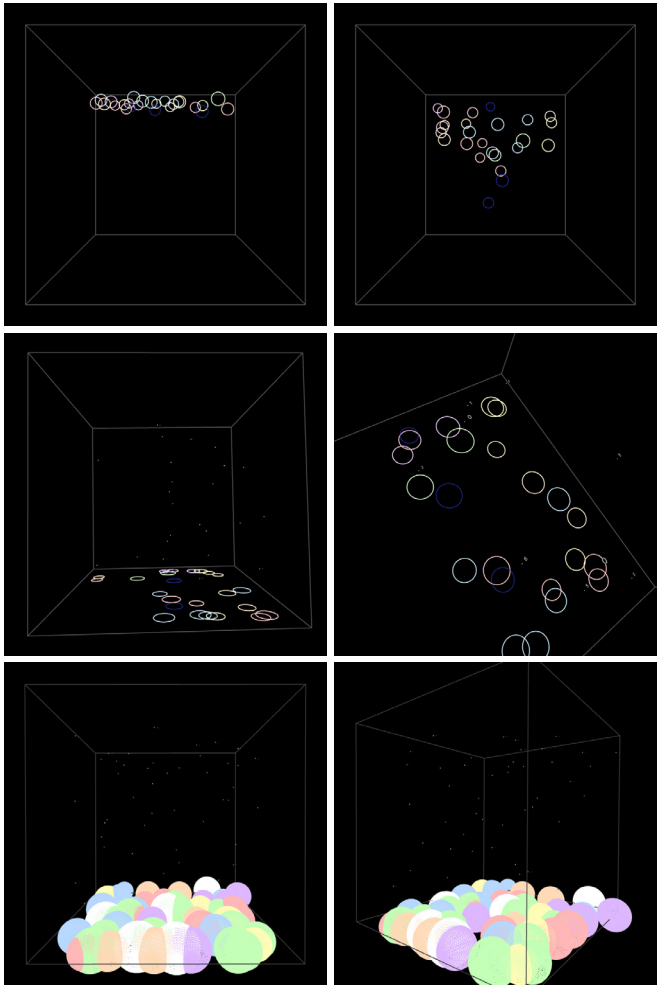
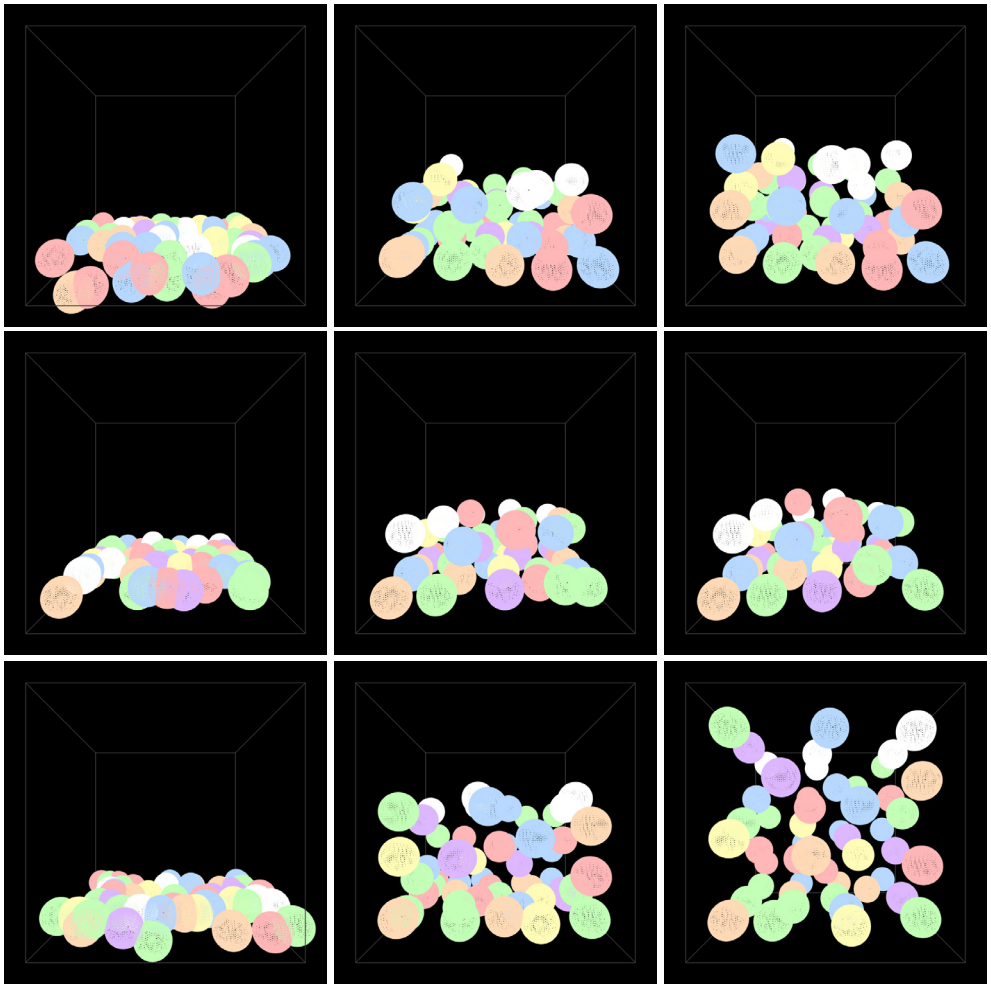


Figure 44a - i (L to R, Top to Bottom): Moving into a Three Dimensional System
This step investigates bringing the two dimensional system into a three dimensional world, then converting the flat circular agents into three dimensional spheres to simulate spatial occupation. The same relationship groupings are being simulated as in the two dimensional system, in order to monitor the adjustment of one variable at a time.

Figure 45a - i (L to R, Top to Bottom): Refining the Basic Principles for a Three Dimensional World

The previous stage put the two dimensional agents into a three dimensional environment, and then converted the agents to three dimensional figures, however behaviours were still programmed for two dimensional beings. This step introduced and refined those same behaviours along the z-axis.



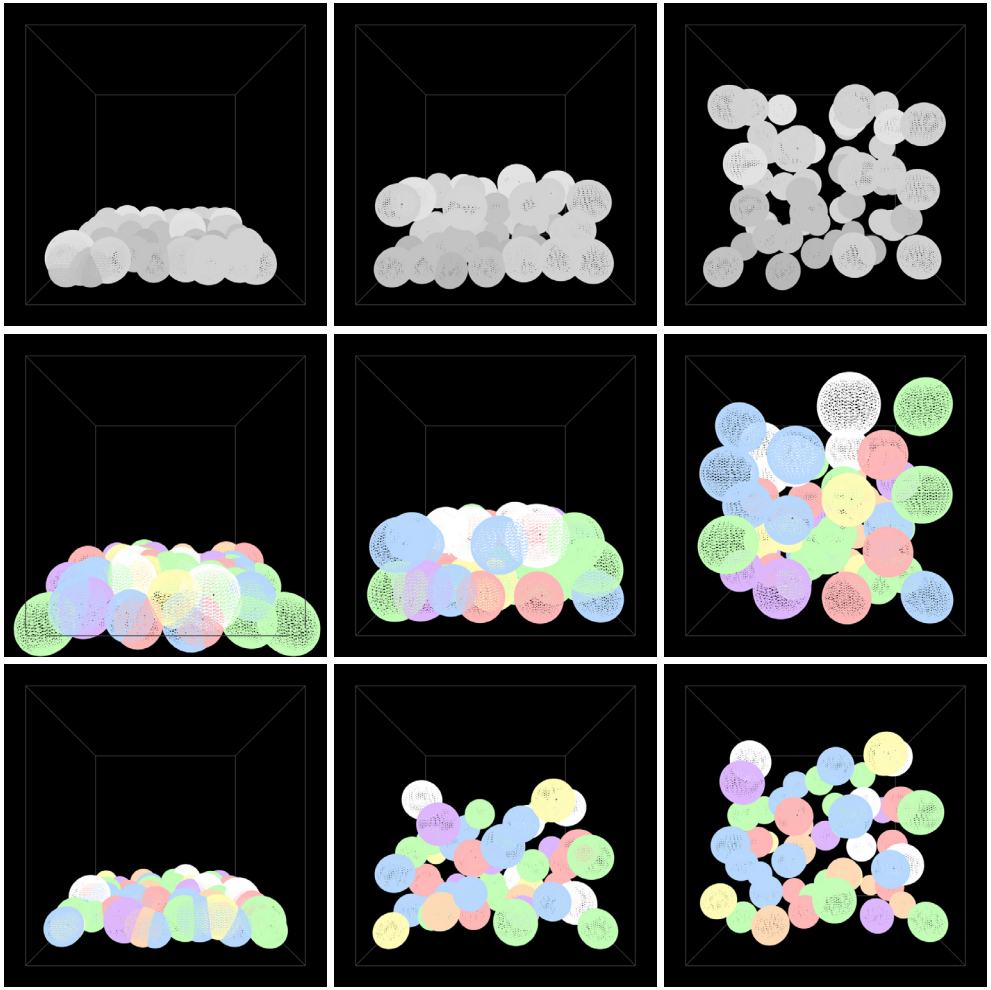
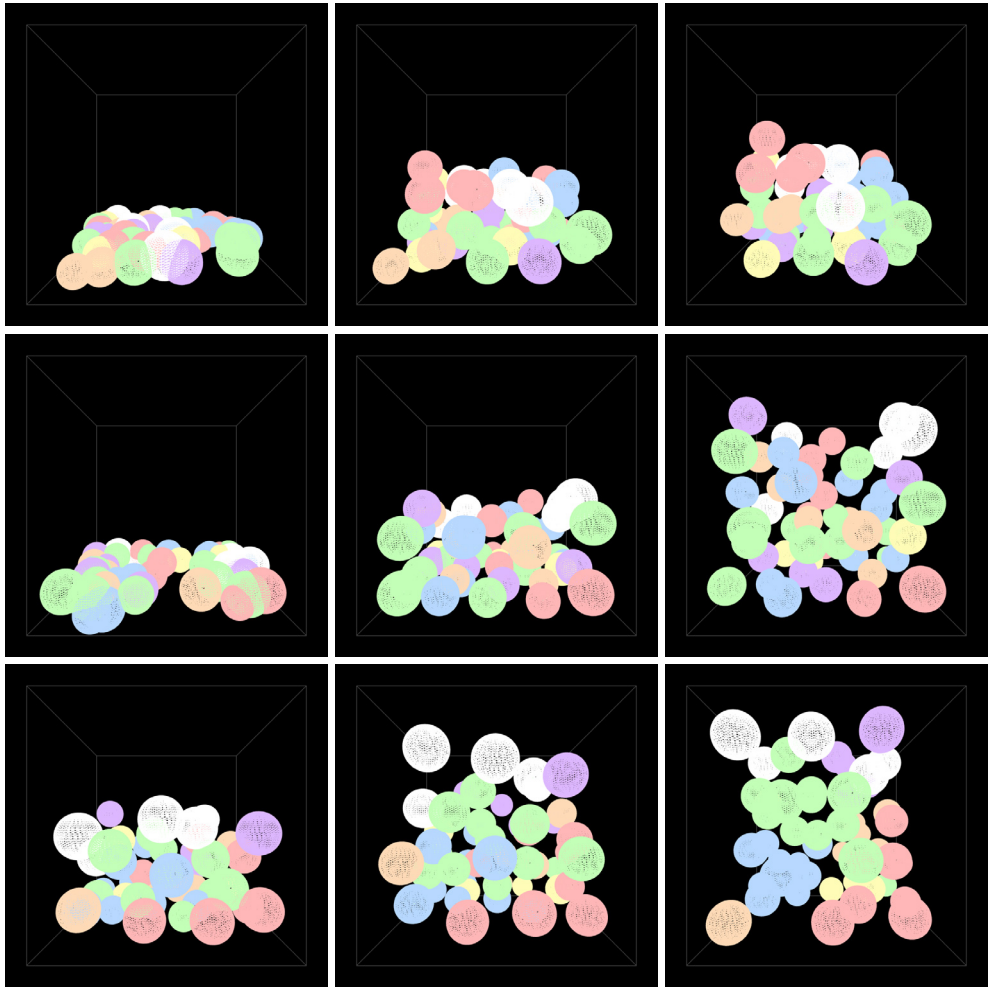


Figure 46a - i (L to R, Top to Bottom): Working with Area
This step investigates the idea of adding a specific mass to each agent based on the area of a each housing unit, first by exploring colour coding to identify different housing areas, then by manipulating the spherical mass of the unit to reflect the desired sizes, rather than simulating one universal agent size. The change in agent size is included in the degree of separation of the housing units, with larger areas requiring greater degrees of separation.

Figure 47a - i (L to R, Top to Bottom):
Refining Iwi Group Cohesion
Iwi group cohesion has been refined so that the level of desired cohesion can be adjusted, investigating the possibility of simulating iwi grouping being the most, or least defining factor of a group, as reflected in the degree to which the coloured spheres are grouped with their own kind, transitioning from very low levels of cohesion, to a very strong relationship formation.



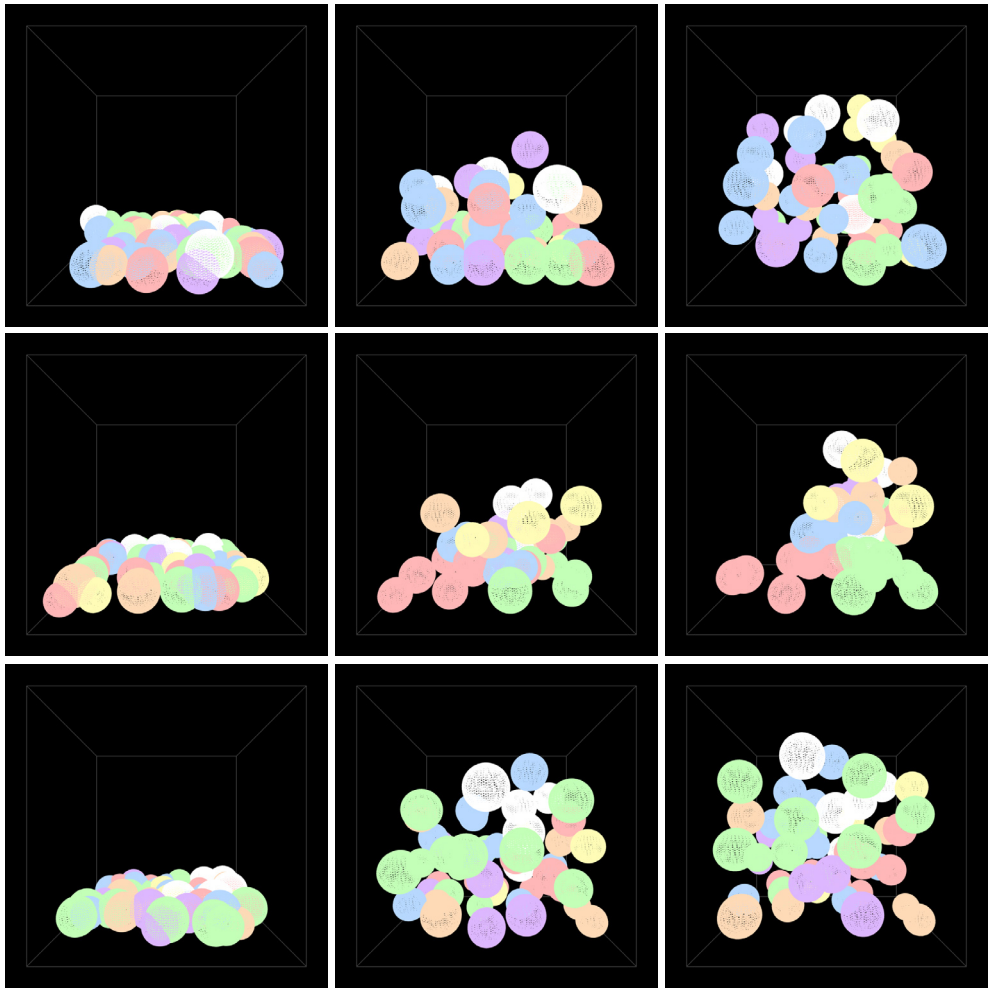
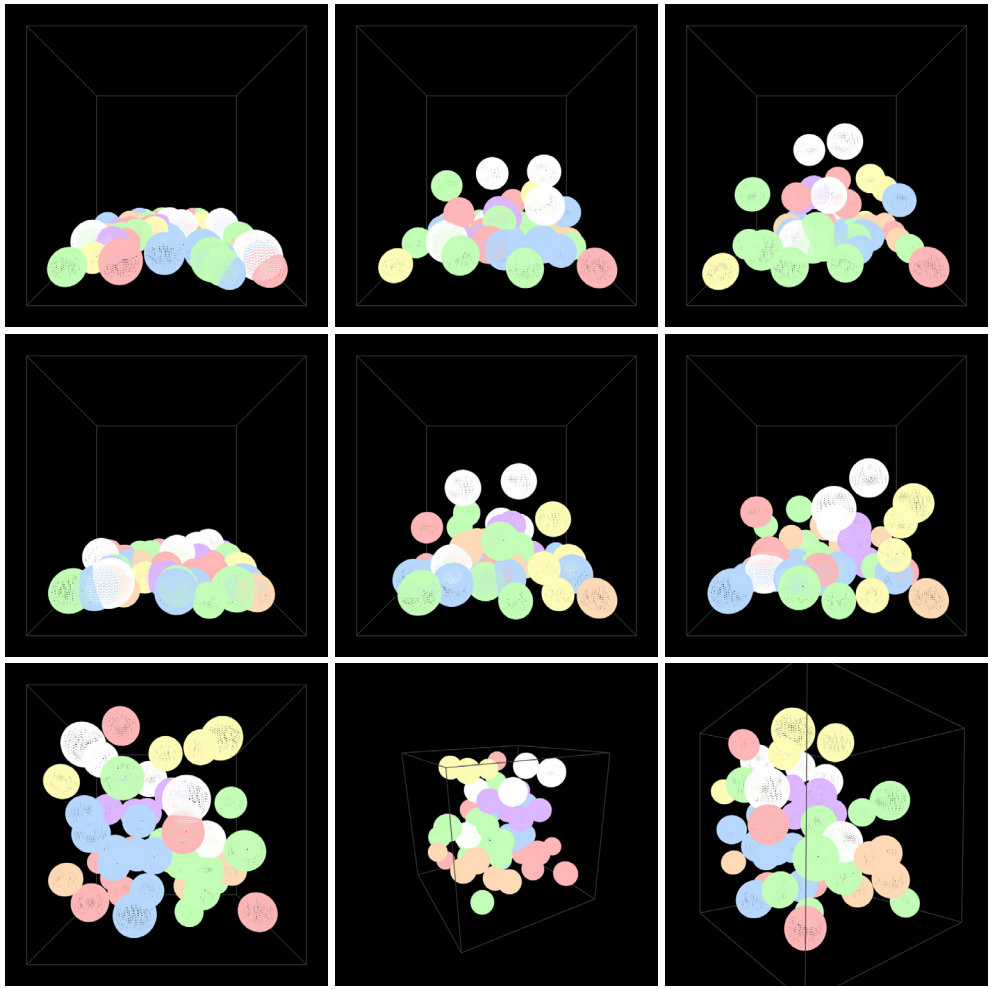


Figure 48a - i (L to R, Top to Bottom): Refining Family Type Cohesion and Separation

This phase investigates the levels of cohesion and separation between family types that are deemed to compliment or contrast with each other. This phase looks to establish the desired multi-generational relationships of Figure 19 within the settlement in a generalised manner, families will cohese with other families that have generation type that compliment their own. This variable works in conjunction with the iwi variable bringing unity between agents of contrasting iwi.

Figure 49a - i (L to R, Top to Bottom):
Adding External Factors
The final testing phase investigates the possibility of inputting relationships beyond that of unit to unit relationships, and inputting functions that relate units to their surrounding environment. This step investigates the addition of gravity, to ensure units are as low lying as possible without compromising relationship structure (for construction purposes), as well as investigating the possibility that some units may desire 'penthouse' style living, for example the input of a view function, where some units (coded yellow in 49g - i) gravitate to be at the upper levels of this community.



C6_Adding_Iwi_Group | Boids | Data | Environment

```
PVector cohesiongroup(ArrayList agents, float type) {
  float count = 0.0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i++) {
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (b.iwitype == type && iwitype == type) {
      float repulsion = 0;
      //float circle = 0;
      //circle = (area > b.area) ? area : b.area;
      //repulsion = circle * 2;
      repulsion = (area + b.area) * 1.2;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steer(sum);
  } else {
    return sum;
  }
}
PVector steerfamily(PVector target, float familymultfactor) {
  PVector desired = PVector.sub(target, pos);
  desired.normalize();
  desired.mult(maxspeed);
  desired.sub(vel);
  desired.limit(maxforce);
  desired.mult(familymultfactor);
  return desired;
}
```

C6_Adding_Iwi_Group | Boids | Data

```
//excel limits
float numIwi = 5;

//unit values
float publicspace = 1.3;

//neighbourhood values
float neighbourhood = 100;


//colours
color c_IT1 = #FFB7B7;
color c_IT2 = #FFDAB7;
color c_IT3 = #FFFCB7;
color c_IT4 = #C3FFB7;
color c_IT5 = #B7D8FF;
color c_IT6 = #DDB7FF;

//box
PVector point1 = new PVector (200, 100);
PVector point2 = new PVector (600, 600);
```

C6_Adding_Family_Type | Boids | Data | Environment

```
PVector familygrouping (ArrayList agents) {
  float count = 0.0;
  float mult = 0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i++) {
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (familytype == 1 && b.familytype == 3
      || familytype == 3 && b.familytype == 9
      || familytype == 5 && b.familytype == 2) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 3;
      }
    }
    if (familytype == 2 && b.familytype == 3
      || familytype == 4 && b.familytype == 8) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 2.5;
      }
    }
    if (familytype == 6 && b.familytype == 6) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
```

C6_Adding_Family_Type | Boids | Data | Environment

```
        sum.add(b.pos);
        mult = 2;
      }
    }
    if (familytype == 2 && b.familytype == 4
      || familytype == 7 && b.familytype == 9) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 1.5;
      }
    }
    if (familytype == 1 && b.familytype == 6) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 1;
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steerfamily(sum, mult);
  } else {
    return sum;
  }
}
```

Figure 50 (L), 51 (Opp.): Examples of Part II Coding

In order to produce what appear to be minuscule changes in simulation, coding is added and subtracted, adapted and refined, to the point where it behaves in the desired manner each and every time it is run.

50. Isolates selected codes that were utilised in Figure 47 and 48.

51. The final sketch code resulting from all phases of investigation.

```
import peasy.*;

PeasyCam cam;

environment Environment;

int worldsize;

void setup () {
  size (800, 800, P3D);
  cam = new PeasyCam(this, 1500);
  cam.setMinimumDistance(100);
  cam.setMaximumDistance(2000);
  worldsize = 1000;
  Environment = new environment();
}



void draw () {
  background(0);
  displayWorldSize();
  translate(-worldsize/2, -worldsize/2, -wo
  Environment.run();
  //strokeWeight(1);
  //stroke(120);
  //rect(point1.x, point1.y, point2.x, poin
}


void displayWorldSize () {
  pushMatrix();
  translate(0, 0, 0);
  noFill();
  stroke(60);
  box(worldsize);
  popMatrix();
}

void keyPressed () {
  if ( key == 's') {
    saveFrame();
  }
}

class Boid {

  PVector pos;
  PVector vel;
  PVector acc;
  int agentcode;
  int familytype;
  int iwitype;
  int housetype;
  float area;
  int multimportance;
  int children;
  int elderly;
  int dependantpersons;
  int shortterm;
  int longterm;
  float maxspeed;
  float maxforce;

  boolean checkiwiisolation;

  Boid (int code, int family, int iwi, int u

    pos = new PVector (random(1, 1000), 1000
    vel = new PVector ();
    acc = new PVector ();
    agentcode = code;
    familytype = family;
    iwitype = iwi;
    housetype = unit;
    area = (sqrt(mass/PI))*20;
    //area = mass;
    multimportance = mult;
    children = child;
    elderly = elders;
    dependantpersons = dependants;
    shortterm = shortt;
    longterm = longt;
    maxspeed = 1;
    maxforce = 1;
  }

  void run (ArrayList agents) {
    update(agents);
    vel.set(0, 0, 0);
    vel.add(acc);
    pos.add(vel);
    render();
  }

  void flock (ArrayList agents) {
    acc.set (0, 0, 0);
    PVector sep = new PVector();
    sep = separate(agents);
    sep.mult(1.2);
    PVector bor = new PVector();
    bor = borders();
    bor.mult(5);
    PVector gra = new PVector();
    gra = gravity();
    gra.mult(0.1);
    PVector vie = new PVector();
    vie = view(agents);
    vie.mult(1);
    //PVector bx = new PVector();
    //bx = box1();
    //bx.mult(0.2);
    acc.add(bor);
    acc.add(sep);
    acc.add(gra);
    acc.add(vie);
    //acc.add(bx);
    familytypeeffect(agents);
    iwieffect(agents);
  }

  PVector view (ArrayList agents) {
    PVector steer = new PVector();
    //frameCount;
    if (housetype == 5 && frameCount < 20
      steer = new PVector (0, -0.5, 0);
    }
    return steer;
  }

  PVector gravity () {
   PVector steer = new PVector(0, 0.1, 0)
   return steer;
   }

  PVector borders () {
    PVector steer = new PVector(0, 0, 0);
    if (pos.x < (0 + area)) {
      steer = new PVector(1, 0, 0);
    }
    if (pos.x > (worldsize - area)) {
      steer = new PVector(-1, 0, 0);
    }
    if (pos.y < (0 + area)) {
      steer = new PVector(0, 1, 0);
    }
    if (pos.y > (worldsize - area)) {
      steer = new PVector(0, -1, 0);
    }
    if (pos.z < (0 + area)) {
      steer = new PVector(0, 0, 1);
    }
    if (pos.z > (worldsize - area)) {
      steer = new PVector(0, 0, -1);
    }
    return steer;
  }

  //PVector box1 () {
  //  PVector steer = new PVector(0, 0);
  //  if (pos.x < (point1.x + area)) {
  //    steer = new PVector(1, 0);
  //  }
  //  if (pos.x > (point2.x - area)) {
  //    steer = new PVector(-1, 0);
  //  }
  //  if (pos.y < (point1.y + area)) {
  //    steer = new PVector(0, 1);
  //  }
  //  if (pos.y > (point2.y - area)) {
  //    steer = new PVector(0, -1);
  //  }
  //  return steer;
  //}

  void familytypeeffect (ArrayList agents
    PVector fam = new PVector();
    fam = familygrouping(agents);
    fam.mult(0.5);
```

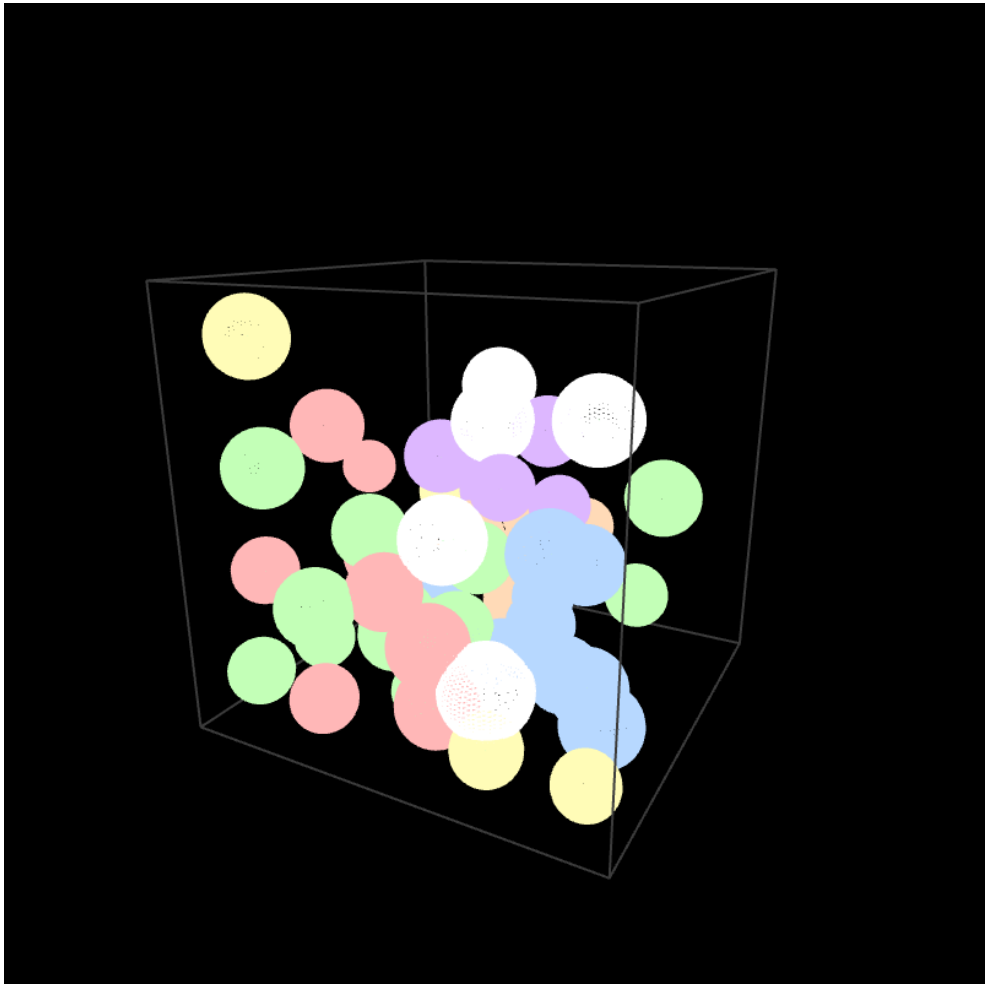


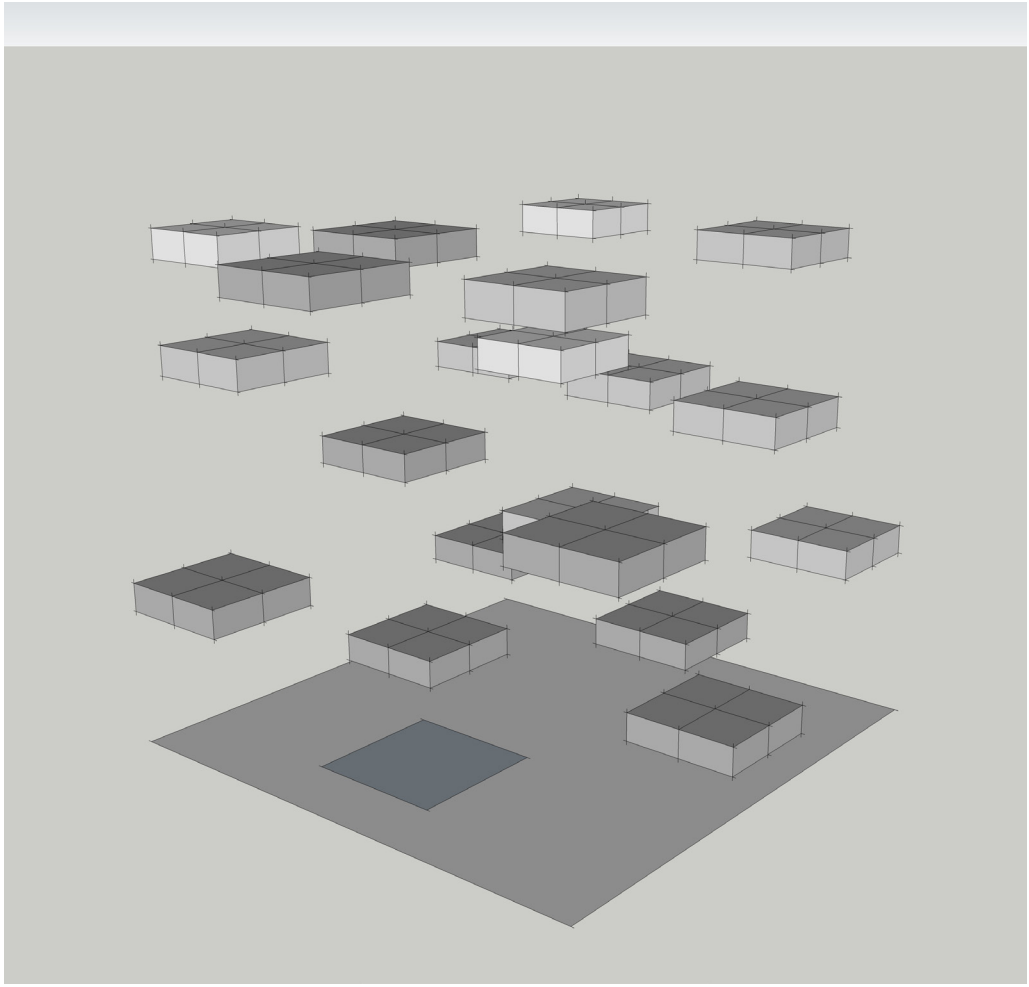
Figure 52: Final Simulation Model
Final three dimensional form as output by the resolved sketch from the testing phase. This simulation depicts how agents (and the housing units within the community they represent) should be spatially organised as a response to the cultural parameters tested up until this point. This output creates an accompanying excel file indicating the location of each unit, and can be used to adapt data to external applications for stages in the design process beyond that of cultural relationship organisation. Appendix Item 10 includes a video simulating the creation and final output of this system.

Figure 53: Traditional Spatial Interpretation of Final Sketch

This offers a more traditional spatial simulation of what the final model may look like, providing cuboid footprints to simulate individual unit area rather than spheres simulating mass.

This indicates how data from the relationship structuring system can be used after it has been generated, substituting the spheres for cubes, then for housing units that fit the requirements of the family in each location.

In the western world, houses are designed then families are placed in them, in this scenario families are designed, and houses placed around them; it is simply another way at looking at how families and their homes are united.



THE THREE DIMENSIONAL SYSTEM

This system, with all behavioural factors included, generates a spatial visualisation of the relationship structure between specific units in an urban environment. While this system would be too precise for the occupation period of western families, this level of specificity is completely appropriate for Māori housing, who view property and housing in a more permanent manner.

These dwellings are more than likely to stay within the same family group for a significant period of time, and as the family incrementally changes and mutates, they will adapt use of the dwelling to fit their specific needs. By the time the original members within the unit have left it will no longer be as important to generate the desired relationships through spatial formation, as the community will by this time have become integrated within the environment, having found other outlets for cultural enrichment, and will no longer solely rely on relationship structuring.

I strongly believe that the establishment of known relationship structures are what is required to grow a Māori community within an urban environment, but I believe just as strongly that once the papakainga has been established, and an outlet for cultural realisation is present within the community, the adaptable and flexible nature of Māori living will instigate a far greater, and far more appropriate means of integration, a form of natural evolution, at which point the relationship matrix will have fulfilled its purpose. The Māori culture has not lasted this long by accident, but by a form of survival through adaptation, and as long as the culture has a means of expression, a way will be found to adapt to the ever changing urban environment.

Part III - Results

CHAPTER SEVEN:

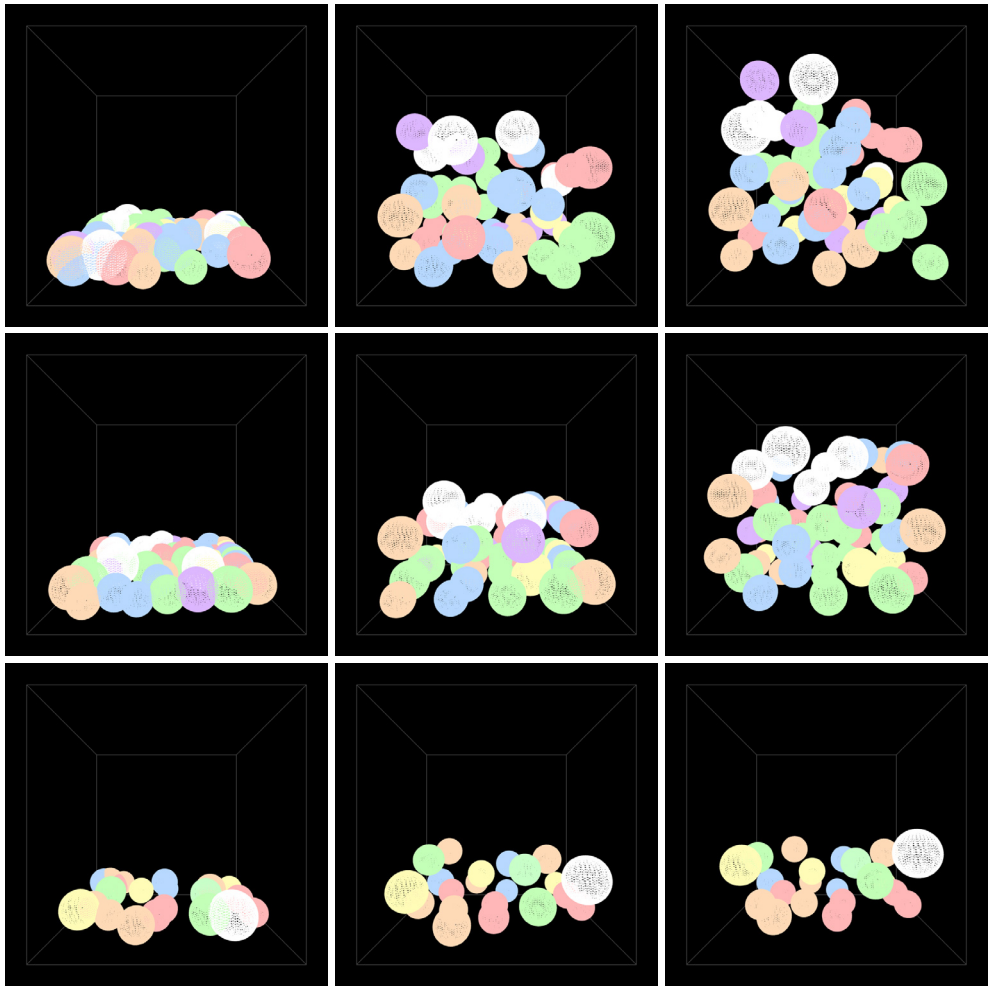
REFINEMENT

The logic introduced and explored in the previous chapter is a non-specific, representative example of the types of relationships within traditional Māori communities, and establishes the format of an organisational system that can realise actual relationships within an urban environment. Through further additions of behaviours, rigorous testing and refinement, a more accurate system can be established with far more regulated and reliable outcomes than the previous examples, creating a far greater consistency in product than the simulations which were informed by primarily randomised content.

This chapter documents the successful and effectual components of this testing phase, identifying the aspects of code that most accurately describe the desired relationships, as well as the identified outcomes. By initially refining the current level of coding, and removing aspects that are superfluous, a pared back example of the more significant relationships are simulated. Secondly, by re-articulating additional unit variables to a binary system (whether the variable is present or not), and then to a slightly more complex system in which there are three degrees of variable presence within the unit composition (present in numbers, present in isolation, or not present at all) we see the platform begin to emerge for which any relationship structure can be based. Within this system the code has been refined to the point that any relationship can be input using one of two coding systems, a system that relates units to one another, or one that relates units to the surrounding environment, requiring very little in terms of code manipulation to create any new relationship structures.

Figure 54a - i (L to R, Top to Bottom): Refining Code to Date

Previous explorations had a few inconsistencies that needed remedying. The gravitational coding was not strong enough to cause units to behave in a manner that worked within the confines of construction capabilities, therefore the gravitational factor was increased (54a - f). The iwi factor has been decreased to allow for other variables to take effect, and lastly, the units for which the system is designed have been revised significantly, working with a hypothetical settlement group of 20 units (54 g - i), the specifications of which are included as Appendix Item 11.

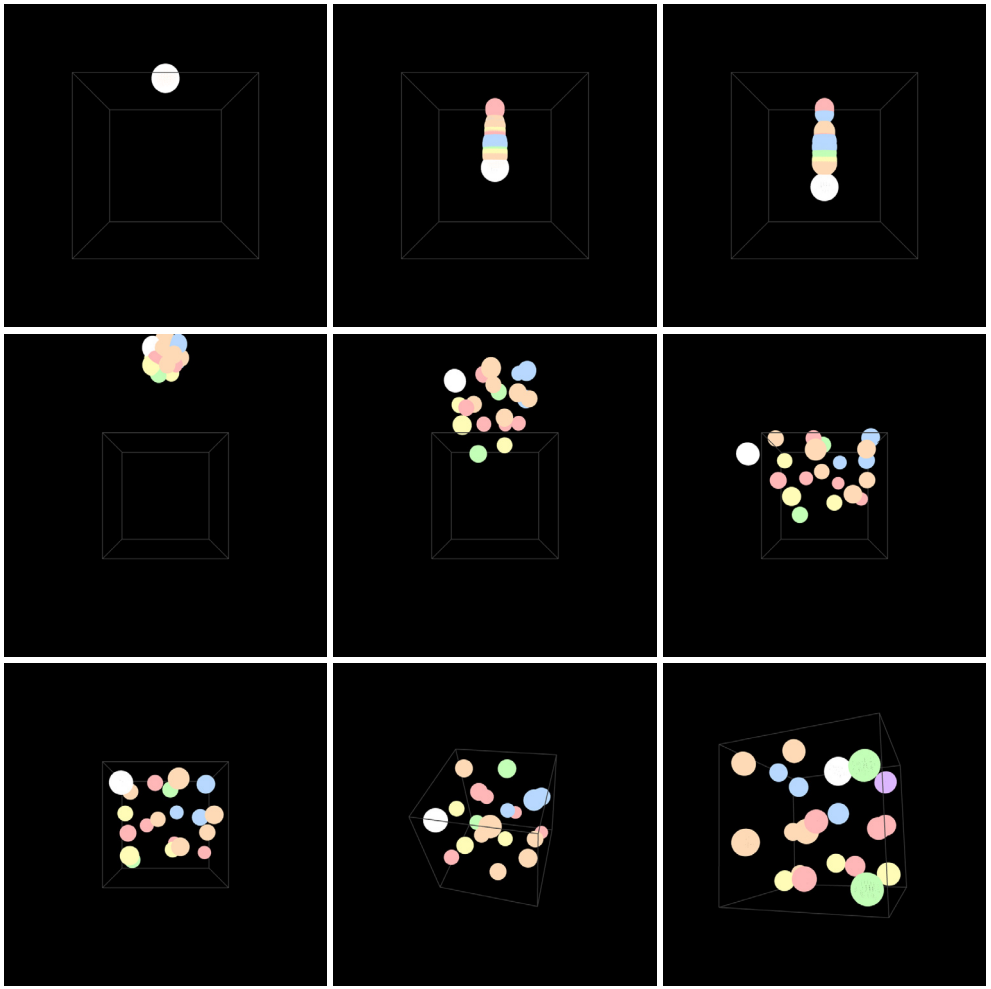


```
Boid (int code, int family, int iwi, int unit, int mass, int mult, int ch

  pos = new PVector (random(1, 1000), 1000, random(1, 1000));
  vel = new PVector ();
  acc = new PVector ();
  agentcode = code;
  familytype = family;
  iwitype = iwi;
  housetype = unit;
  area = (sqrt(mass/PI))*20;
  //area = mass;
  multimportance = mult;
  children = child;
  elderly = elders;
  dependantpersons = dependants;
  shortterm = shortt;
  longterm = longt;
  adults = adult;
  youngadults = yadult;
  family = fam;
  efamily = efam;
  housesize = hsize;
  maxspeed = 1;
  maxforce = 1;
}

void run (ArrayList agents) {
  update(agents);
  vel.set(0, 0, 0);
  vel.add(acc);
  pos.add(vel);
  render();
```

```
Boid (int code, int family, int iwi, int unit, int mass, int mult, int

  pos = new PVector (random(500), -1000, random(500));
  vel = new PVector ();
  acc = new PVector ();
  agentcode = code;
```

```
Boid (int code, int family, int iwi, int unit, int mass, int mult, in

  pos = new PVector (random(499, 501), -1000, random(499, 501));
  vel = new PVector ();
  acc = new PVector ();
```

Figure 55 (Opp.), 56a - i (L to R, Top to Bottom): Relocating the Source
With units initiating from a random source in all previous investigations, there were preconceived relationships from the initial stages of the simulation that heavily influence final outcomes. To counter this all units were moved to the same source, out of the environment with which they desire to occupy in order to establish a blank slate from which to begin, where no unit has a closer relationship with any other. This however proved problematic in that with no deviation in x, y, or z location, units had no distance between them, meaning there was no data to tell them which way to move in order to cohere or separate (56a - c). Because of this, the source point was adapted to be a 2 by 2 pixels square outside of the environment, which proves to be enough space to initiate movement, without significantly effecting the final outcome.
55. Code created for final test simulation.
56a - i. Outcomes of testing.

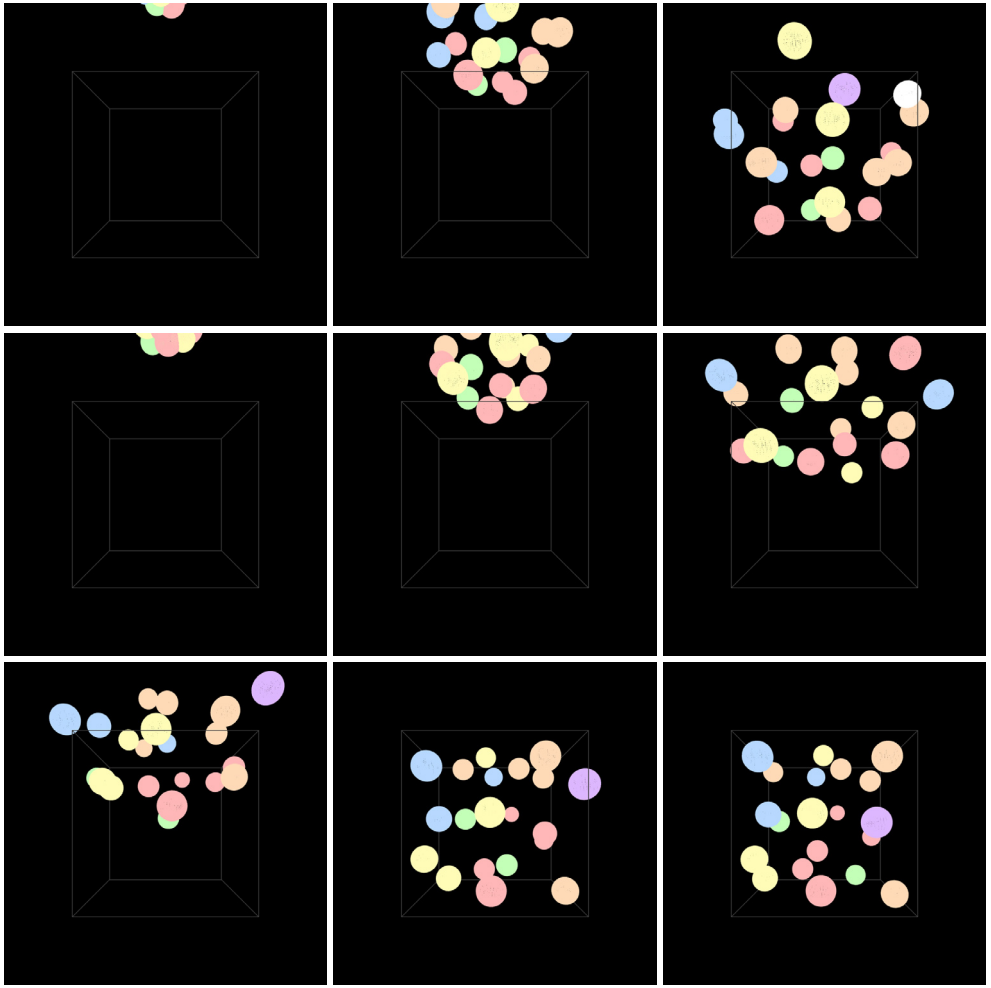


```
void iwieffect (ArrayList agents) {
  PVector coh1 = new PVector();
  for (int i = 1; i < 6; i++) {
    coh1 = cohesiongroup(agents, i);
    coh1.mult(1);
    acc.add(coh1);
  }
}

PVector cohesiongroup(ArrayList agents,
  float count = 0.0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i+
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (b.iwitype == type && iwitype ==
      float repulsion = 0;
      //float circle = 0;
      //circle = (area > b.area) ? area
      //repulsion = circle * 2;
      repulsion = (area + b.area) * 1.2
      if (d > repulsion && d < neighbou
        count+=1.0;
        sum.add(b.pos);
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steer(sum);
  } else {
    return sum;
  }
```

```
void flock (ArrayList agents) {
  acc.set (0, 0, 0);
  PVector sep = new PVector();
  sep = separate(agents);
  sep.mult(1.2);
  PVector bor = new P
  bor = borders();
  bor.mult(1);
  PVector gra = new P
  gra = gravity();
  gra.mult(0.1);
  //PVector vie = new
  //vie = view(agents
  //vie.mult(1);
  //PVector bx = new
  //bx = box1();
  //bx.mult(0.2);
  acc.add(bor);
  acc.add(sep);
  acc.add(gra);
  //acc.add(vie);
  //acc.add(bx);
  //familytypeeffect(agents);
  iwieffect(agents);
}
```

```
if (iwitype == 1) {
  stroke(c_IT1);
} else if (iwitype == 2) {
  stroke(c_IT2);
} else if (iwitype == 3) {
  stroke(c_IT3);
} else if (iwitype == 4) {
  stroke(c_IT4);
} else if (iwitype == 5) {
  stroke(c_IT5);
} else if (iwitype == 99) {
  stroke(c_IT6);
}
```

Figure 57 (Opp.), 58a - i (L to R, Top to Bottom): Finalising Iwi Type Simulations
Finding the degree to which iwi cohere with units of the same type proved to be the most difficult step within the simulation. Each refinement of another variable changes the effect of iwi type relationships, requiring a great deal of tweaking to create the right level of motivation to induce units to cohere with their iwi type, but not to a degree that causes iwi isolation within the relationship structure. This phase of iwi refinement was the last, as further variable refinement did not skew previously programmed behaviours, indicating the right balance has been achieved.
57. Code created for final test simulation.
58a - i. Outcomes of testing.

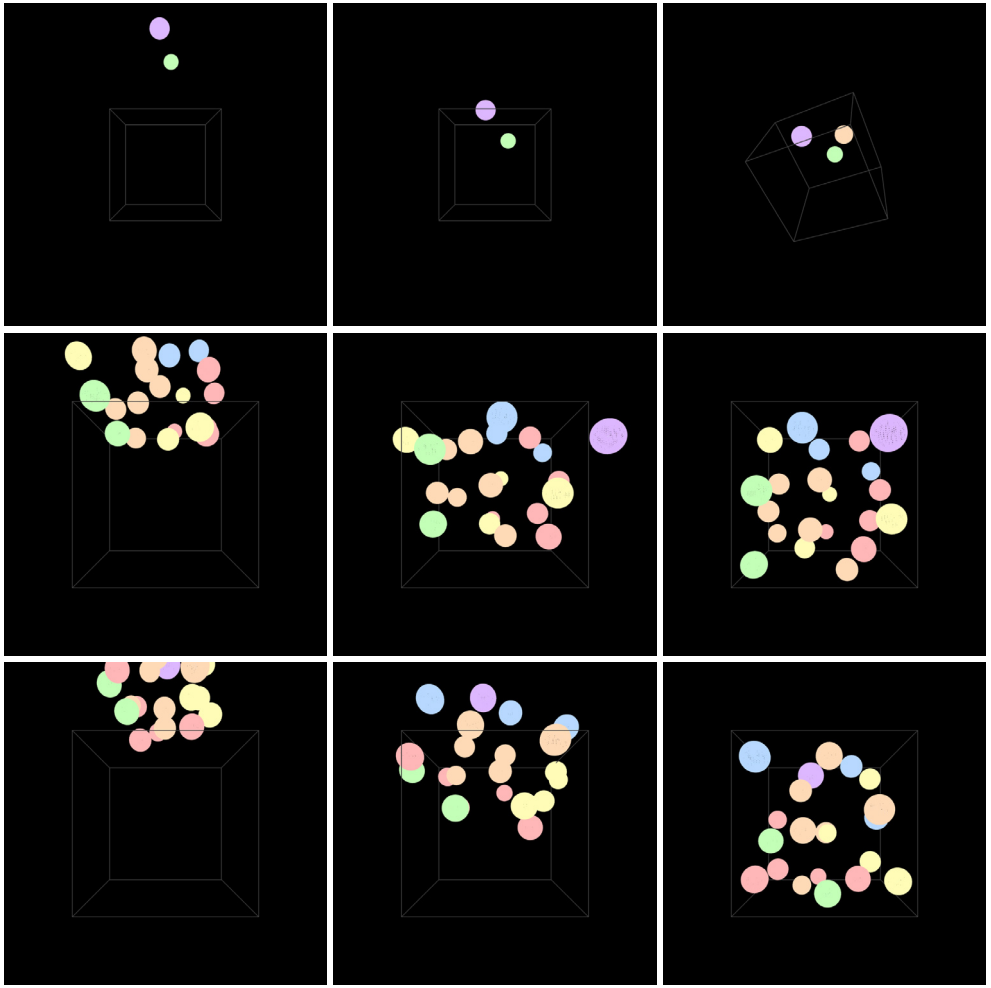


```
void ageeffect (ArrayList agents) {
  PVector age = new PVector();
  age = agegrouping(agents);
  age.mult(0.6);
  acc.add(age);
}

PVector agegrouping (A
  float count = 0.0;
  float mult = 0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i++)
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (elderly == 1 && children == 0 &&
      || elderly == 1 && children == 0 &&
      || elderly == 2 && children == 0 &&
      || elderly == 2 && children == 0 &&
      float repulsion = 0;
      repulsion = (area > b.area) ? area
      if (d > repulsion && d < neighbourh
        count+=1.0;
        sum.add(b.pos);
        mult = 1;
      }
    }
    if (elderly == 0 && children == 1 && b.elderly == 1
      || elderly == 0 && children == 1 && b.elderly == 2
      || elderly == 0 && children == 2 && b.elderly == 1
      || elderly == 0 && children == 2 && b.elderly == 2) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.a
      if (d > repulsion && d < neighbourhood) {
```

```
void ageeffect (ArrayList
  PVector age = new PVecto
  age = agegrouping(agen
  age.mult(0.6);
  acc.add(age);
}
```

```
if (elderly == 2 && children == 0 && b.children == 1
  || elderl if (elderly == 2 && adults == 0 && b.adults == 1
      if (   || elderly == 2 && adults == 0 && b.adults == 2) {
        || if (elderly == 1 && adults == 0 && b.adults == 1
        if (el   || elderly == 1 && adults == 0 && b.adults == 2) {
          || elderly == 1 && children == 0 && b.children == 2) {
if (elderly == 2 && b.elderly == 1
  || elderly == 2 && b.elderly == 2) {
      if (elderly == 0 && children == 1 && b.elderly == 1
if (children == 2 && adults == 0 && b.adults == 1          == 2) {
  || children == 2 && adults == 0 && b.adults == 2) {
  float repulsion = 0;
  repulsion = (area > b.area) ? area : b.area;
  if (d > repulsion && d < neighbourhood) {
    count+=1.0;
    sum.add(b.pos);
    mult = 3;
  }
}
if (adults == 2 && b.adults == 1
  || adults == 2 &if (children == 1 && adults == 0 && b.adults == 1
                  || children == 1 && adults == 0 && b.adults == 2) {
  if (children == 2 && b.children == 1
    || children == 2 && b.children == 2) { children == 1
if (youngadults == 2 && b.youngadults == 1
  || youngadults == 2 && b.youngadults == 2) {
if (elderly == 0 && children == 2 && b.elderly == 1
  || elderly == 0 && children == 2 && b.elderly == 2) {
```

Figure 59 (Opp.), 60a - i (L to R, Top to Bottom): Generational Attraction
This variable replaced that of family type, as each unit in the refined data specifies family composition in terms of Children, Young Adults, Adults, and Elderly, with degrees of 0 (no members), 1 (1 member), or 2 (more than 1 member). By this degree we can specify that each generation gravitate towards others of their generation, the degree of which is dependant on the number of members within the unit. It is also specified for example that units with elderly members yet no adults would gravitate towards units with adult members who can provide support. This variable constructs many rules based on the multi generational relationships Māori live with, and provides a great deal of the relationship structuring within papakainga. The specific relationships are scripted within the code of Appendix Item 12, along with explanations.
59. Code created for final test simulation.
60a - i. Outcomes of testing.

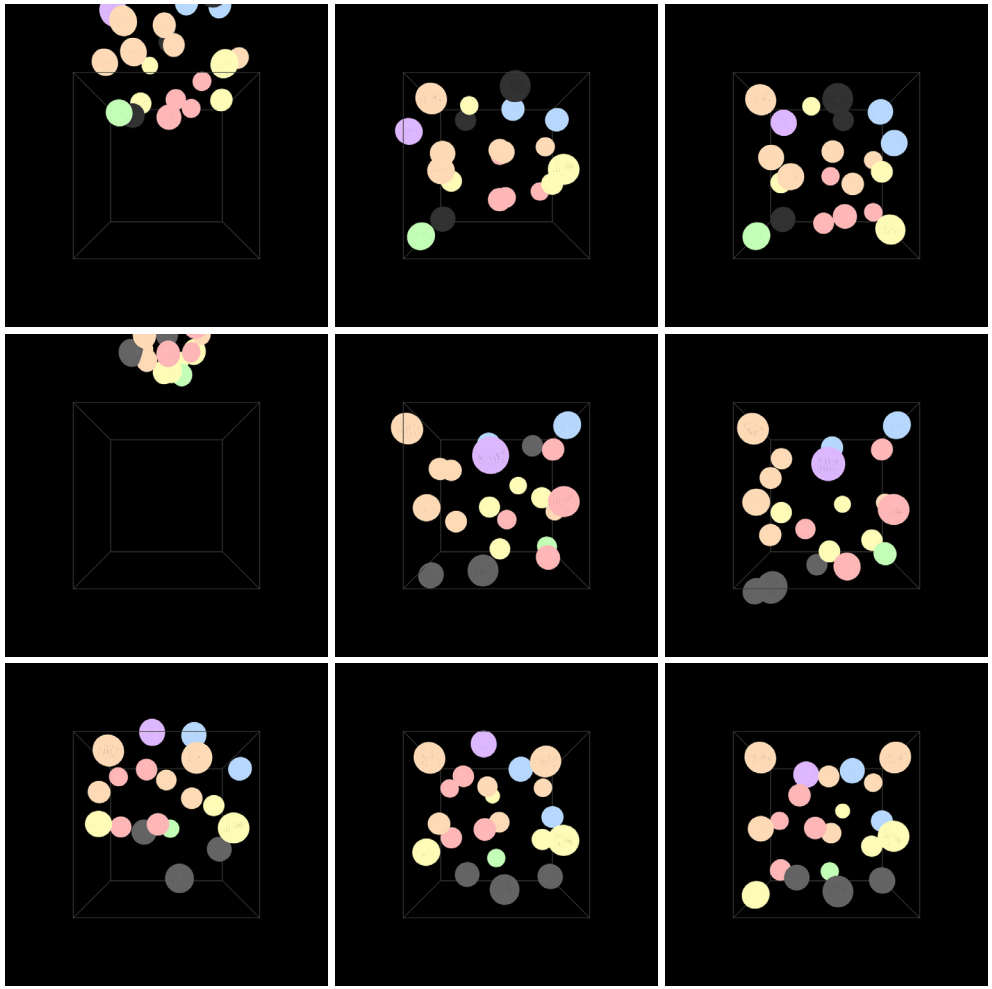


```
void flock (ArrayList agents) {
  acc.set (0, 0, 0);
  PVector sep = new PVector();
  sep = separate(agents);
  sep.mult(1.2);
  PVector bor = new PVector();
  bor = borders();
  bor.mult(1);
  PVector gra = new PVector();
  gra = gravity();
  gra.mult(0.1);
  //PVector vie = new PVector();
  //vie = view(agents);
  //vie.mult(1);
  //PVector bx = new PVector();
  //bx = box1();
  //bx.mult(0.2);
  acc.add(bor);
  acc.add(sep);
  acc.add(gra);
  //acc.add(vie);
  //acc.add(bx);
  //familytypeeffect(agents);
  iwieffect(agents);
  ageeffect(agents);
}
```

```
acc.add(bor);
acc.add(sep);
acc.add(gra);
acc.add(pdep);
//acc.add(vie);
//acc.add(bx);
//familytypeeffect(agents);
iwieffect(agents);
ageeffect(agents);
```

```
PVector physicaldependance(ArrayList agents) {
  PVector steer = new PVector();
  if (dependantpersons == 1 && pos.y < (area - 1000)) {
    steer = new PVector(0, 1, 0);
  }
  return steer;
}
```

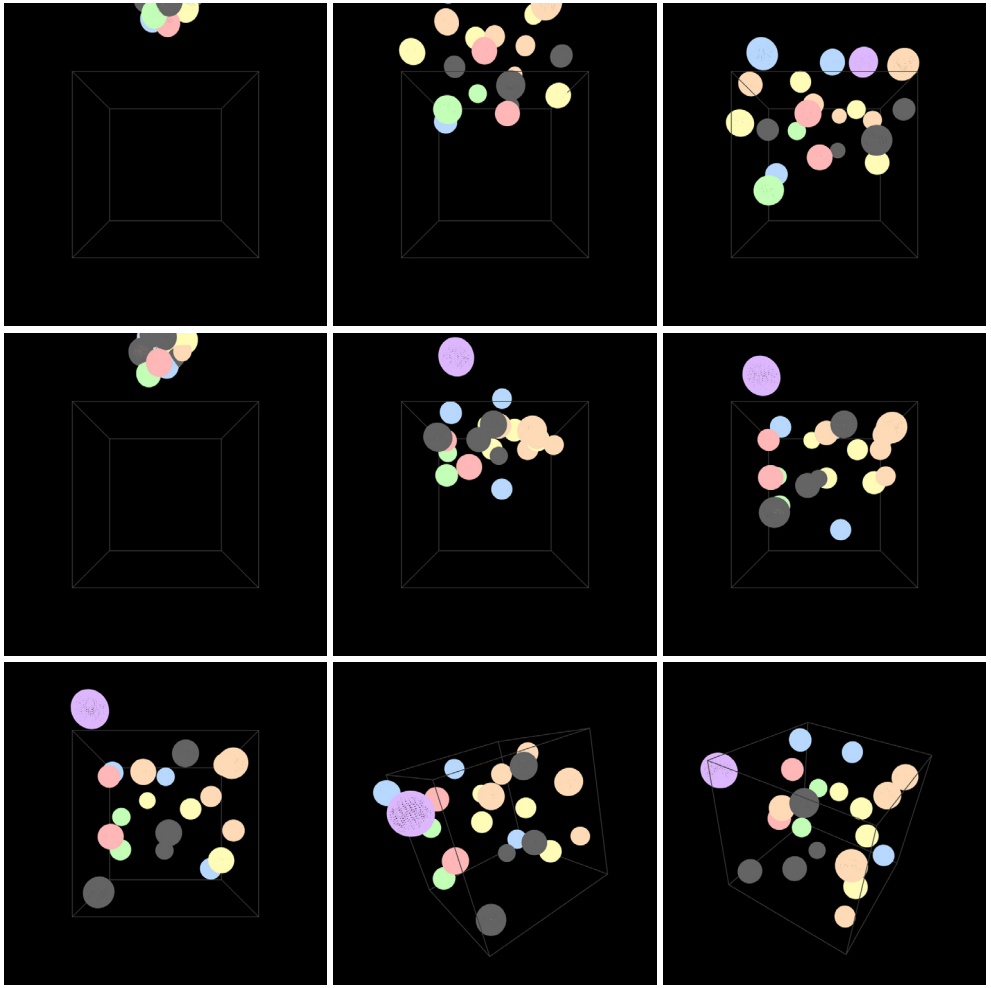
Figure 61 (Opp.), 62a - i (L to R, Top to Bottom): Dependant People
Investigating the gravitational variable provides the means for designing relationships that to cater to units with members who are dependant on certain aspects of accessibility in order to live comfortably. This addresses the issue of physical disability or insufficiency and works to position certain units at the lower levels of the spatial relationship network. In this scenario, with a variable acting on an isolated selection of units, the agents deemed dependant are coloured grey in order to identify the function in effect.
61. Code created for final test simulation.
62a - i. Outcomes of testing.



```
PVector relationgrouping (ArrayList agents, float type) {
  float count = 0.0;
  float mult = 0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i++) {
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (family == type && b.family == type
      || family == type && b.efamily == type
      || efamily == type && b.family == type
      || efamily == type && b.efamily == type) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < famneighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 4;
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steerfamily(sum, mult);
  } else {
    return sum;
  }
}
```

```
void relationseffect (ArrayList agents) {
  PVector rel = new PVector();
  for (int i = 1; i < 11; i++) {
    rel = relationgrouping(agents, i);
    rel.mult(0.2);
    acc.add(rel);
  }
}
```

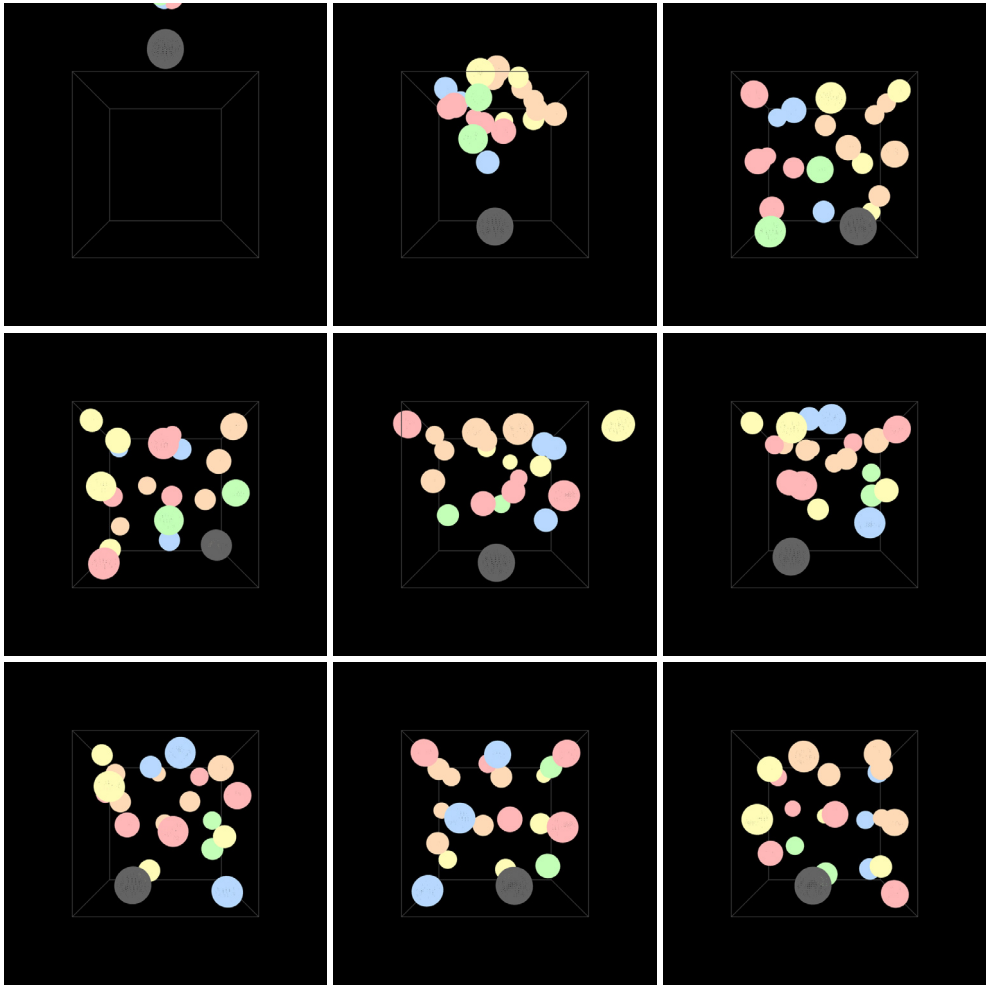
Figure 63 (Opp.), 64a - i (L to R, Top to Bottom): Familial Relationships
The refined data for simulation has the input of identifying a family or families to which a unit is associated. This addresses the unity within Māori families and works to simulate close relationships that may not otherwise be identified through iwi identification due to scenarios where a unit may adopt another iwi to that of their family due to marriage or any other reason. One family group is identified through the use of a grey identifier in these simulations, and the final result is that each unit is within a close proximity to a fellow family member, yet not confined to being in close proximity with them all.
63. Code created for final test simulation.
64a - i. Outcomes of testing.



```
void keyPressed (ArrayList agents) {
  if (key == 'e' ) {
    table = new Table();
    table2 = new Table();

table.addColumn("ID");
table.addColumn("x");
table.addColumn("y");
table.addColumn("z");
table.addColumn("area");
  PVector ent = new PVector();
  ent = entry();
  ent.mult(5);
  //PVector vie = new PVector();
  //vie = view(agents);
  //vie.mult(1);
  //PVector bx = for (int i = 0; i <
  //bx = box1();     Boid agent = (Boid) agents.get(i);
  //bx.mult(0.2) TableRow newRow = table.addRow();
  acc.add(bor);  newRow.setInt("ID", agent.agentcode);
  acc.add(sep);  newRow.setFloat("x", agent.pos.x);
  acc.add(gra);  newRow.setFloat("y", agent.pos.y);
  acc.add(pdep); newRow.setFloat("z", agent.pos.z);
  acc.add(ent);  newRow.setFloat("area", agent.area);

                 saveTable(table, "data/location.csv");
                   }
```

```
PVector entry() {
  PVector steer = new PVector();
  if (iwitype == 99 && pos.y < (worldsize - (area*1.1))) {
    steer = new PVector(0, 1, 0);
  }
  if (iwitype == 99 && pos.z < (worldsize - (area*1.1))) {
    steer = new PVector(0, 0, 1);
  }
  if (iwitype == 99 && pos.x < ((worldsize/2) - 200)) {
    steer = new PVector(1, 0, 0);
  }
  if (iwitype == 99 && pos.x > ((worldsize/2) + 200)) {
    steer = new PVector(-1, 0, 0);
  }
  return steer;
}
```

Figure 65 (Opp.), 66a - i (L to R, Top to Bottom): Addition of Communal Spaces
This stage included the addition of an entryway along a selected street front, in the most appropriate central location. The entry space has been treated as Unit #21, and has unique behaviours to other units; each variable has been assigned a number that does not correspond with the variables of any other unit, therefore will not be called to carry out any behaviour with other agents. This type of functionality could be used to construct as many or as few communal spaces as desired, to be located specifically, as it is in this example, or to react within the other units. In this case an entry was the only addition as Māori greatly value spaces that provide a threshold when needed, so an entry is deemed appropriate.
65. Code created for final test simulation.
66a - i. Outcomes of testing.



```
for (int i=1; i < lines.length
  agentNum = int(csv[i][0]);
  houseArea = int(csv[i][1]);
  iwiNum = int(csv[i][2]);
  family = int(csv[i][3]);
  additionalFamily = int(csv[i
  children = int(csv[i][5]);
  youngAdults = int(csv[i][6])
  adults = int(csv[i][7]);
  elderly = int(csv[i][8]);
  dependantPersons = int(csv[i
  familyType = int(csv[i][11])
  houseType = int(csv[i][12]);
  centralImportance= int(csv[i
  shortTerm = int(csv[i][15]);
  longTerm = int(csv[i][16]);
  Boid point = new Boid(agentN
    additionalFamily, children
    dependantPersons, familyTy
    shortTerm, longTerm);
  agents.add(point);
}
```

```
Boid (int code, int harea, int iwi
      int yadult, int adult, int e
      int htype, int cimp, int ste

  pos = new PVector (random(499, 5
  vel = new PVector ();
  acc = new PVector ();
  agentcode = code;
  area = (sqrt(harea/PI))*20;
  iwitype = iwi;
  family = fam;
  afamily = afam;
  children = child;
  youngadults = yadult;
  adults = adult;
  elderly = elders;
  dependantpersons = dependants;

  familytype = ftype;
  housetype = htype;

  centralimportance = cimp;
  shortterm = sterm;
  longterm = lterm;

  maxspeed = 1;
  maxforce = 1;
}
```

```
  PVector sep = new PVector();
  sep = separate(agents);
  sep.mult(1.2);
  PVector bor = new PVector();
  bor = borders();
  bor.mult(1);
  PVector gra = new PVector();
  gra = gravity();
  gra.mult(0.1);
  PVector pdep = new PVector();
  pdep = physicaldependance();
  pdep.mult(1);
  PVector ent = new PVector();
  ent = entry();
  ent.mult(1);
  //PVector vie = new PVector();
  //vie = view(agents);
  //vie.mult(1);

  acc.add(sep);
  acc.add(bor);
  acc.add(gra);
  acc.add(pdep);
  acc.add(ent);
  //acc.add(vie);

  iwieffect(agents);
  ageeffect(agents);
  relationseffect(agents);
  //familytypeeffect(agents);
}
```

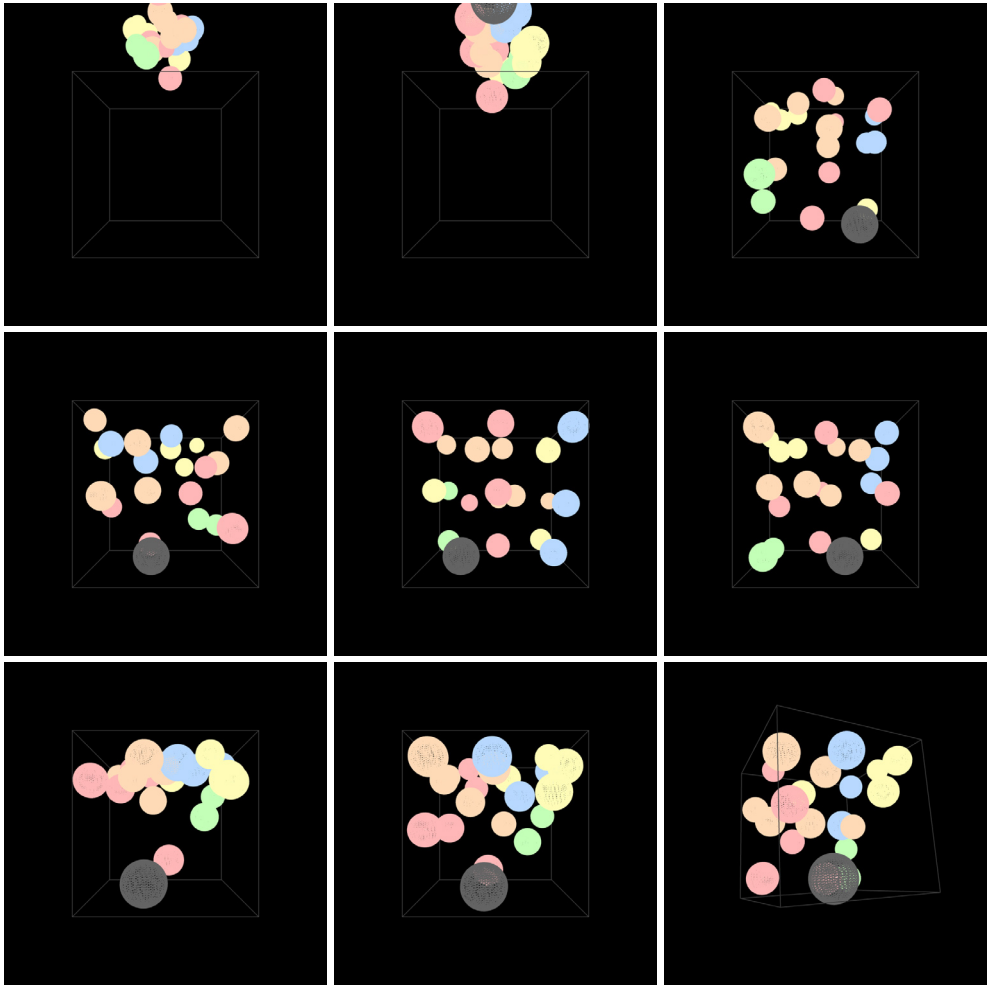
Figure 67 (Opp.), 68a - i (L to R, Top to Bottom): Final System

The final step in the refinement of the system involved making it fool proof and adaptable for future use. Potential code has been tested and left as an option for use should it be needed, and the system has been refined for the purpose of being used for other scenarios requiring relationship organisation systems.

See Appendix Item 12 for the final system.

67. Code created for final simulation.

68a - i. Outcomes of testing.



```
import peasy.*;

PeasyCam cam;

environment Environment;

int worldsize;

void setup () {
  size (800, 800, P3D);
  cam = new PeasyCam(this, 2000);
  cam.setMinimumDistance(100);
  cam.setMaximumDistance(3000);
  worldsize = 1000;
  Environment = new environment();
}


void draw () {
  background(0);
  displayWorldSize();
  translate(-worldsize/2, -worldsize/2, -worldsize/2);
  Environment.run();
}


void displayWorldSize () {
  pushMatrix();
  translate(0, 0, 0);
  noFill();
  stroke(60);
  box(worldsize);
  popMatrix();
}

void keyPressed () {
  if ( key == 's') {
    saveFrame();
  }
}


class Boid {

  PVector pos;
  PVector vel;
  PVector acc;
  int agentcode;
  float area;
  int iwitype;
  int family;
  int afamily;
  int children;
  int youngadults;
  int adults;
  int elderly;
  int dependantpersons;

  int familytype;
  int housetype;

  int centralimportance;
  int shortterm;
  int longterm;

  float maxspeed;
  float maxforce;

  Table table;
  Table table2;


  Boid (int code, int harea, int iwi, int fam, int afam,
        int adult, int elders, int dependants, int ftype
        int sterm, int lterm) {

    pos = new PVector (random(499, 501), -1000, random(4
    vel = new PVector ();
    acc = new PVector ();
    agentcode = code;
    area = (sqrt(harea/PI))*20;
    iwitype = iwi;
    family = fam;
    afamily = afam;
    children = child;
    youngadults = yadult;
    adults = adult;
    elderly = elders;
    dependantpersons = dependants;

    familytype = ftype;
    housetype = htype;

    centralimportance = cimp;
    shortterm = sterm;
    longterm = lterm;

    maxspeed = 1;
    maxforce = 1;
  }


  void run (ArrayList agents) {
    update(agents);
    vel.set(0, 0, 0);
    vel.add(acc);
    pos.add(vel);
    render();
    keyPressed(agents);
  }


  void flock (ArrayList agents) {
    acc.set (0, 0, 0);

    PVector sep = new PVector();
    sep = separate(agents);
    sep.mult(1.2);
    PVector bor = new PVector();
    bor = borders();
    bor.mult(1);
    PVector gra = new PVector();
    gra = gravity();
    gra.mult(0.1);
    PVector pdep = new PVector();
    pdep = physicaldependance();
    pdep.mult(1);
    PVector ent = new PVector();
    ent = entry();
    ent.mult(1);
    //PVector vie = new PVector();
    //vie = view(agents);
    //vie.mult(1);

    acc.add(sep);
    acc.add(bor);
    acc.add(gra);
    acc.add(pdep);
    acc.add(ent);

    iwieffect(agents);
    ageeffect(agents);
    relationseffect(agents);
    //familytypeeffect(agents);
  }


  PVector separate(ArrayList agents) {
    float count = 0.0;
    PVector sum = new PVector();
    for (int i = 0; i < agents.size(); i++) {
      Boid b = (Boid) agents.get(i);
      float d = pos.dist(b.pos);
      float repulsion = 0;
      repulsion = (area + b.area) * 2;
      if (d > 0 && d < repulsion) {
        count+=1.0;
        PVector diff = PVector.sub(pos, b.pos);
        diff.normalize();
        diff.div(d);
        sum.add(diff);
      }
    }
    if (count > 0 ) {
      sum.div(count);
    }
    if (sum.mag() > 0) {
      sum.normalize();
      sum.mult(maxspeed);
      sum.sub(vel);
      sum.limit(maxforce*1.5);
    }
    return sum;
  }


  PVector borders () {
    PVector steer = new PVector(0, 0, 0);
    if (pos.x < (0 + area)) {
      steer = new PVector(1, 0, 0);
    }
    if (pos.x > (worldsize - area)) {
      steer = new PVector(-1, 0, 0);
    }
    if (pos.y < (0 + area)) {
      steer = new PVector(0, 1, 0);
    }
    if (pos.y > (worldsize - area)) {
      steer = new PVector(0, -1, 0);
    }
    if (pos.z < (0 + area)) {
      steer = new PVector(0, 0, 1);
    }
    if (pos.z > (worldsize - area)) {
      steer = new PVector(0, 0, -1);
    }
    return steer;
  }


  PVector gravity () {
    PVector steer = new PVector();
    if ( frameCount < 1000) {
      steer = new PVector(0, 0.1, 0);
    }
    return steer;
  }


  PVector physicaldependance() {
    PVector steer = new PVector();
    if (dependantpersons == 1 && pos.y < (area - 1000))
      steer = new PVector(0, 1, 0);
    }
    return steer;
  }


  PVector entry() {
    PVector steer = new PVector();
    if (iwitype == 99 && pos.y < (worldsize - (area * 1
      steer = new PVector(0, 1, 0);
    }
    if (iwitype == 99 && pos.z < (worldsize - (area * 1
      steer = new PVector(0, 0, 1);
    }
    if (iwitype == 99 && pos.x < ((worldsize/2) - 200))
      steer = new PVector(1, 0, 0);
    }
    if (iwitype == 99 && pos.x > ((worldsize/2) + 200))
      steer = new PVector(-1, 0, 0);
    }
    return steer;
  }


  //PVector view (ArrayList agents) {
  //  PVector steer = new PVector();
  //  //frameCount;
  //  if (housetype == 5 && frameCount < 1000) {
  //    steer = new PVector (0, -0.5, 0);
  //  }
  //  return steer;
  //}


  void iwieffect (ArrayList agents) {
    PVector iwi = new PVector();
    for (int i = 1; i < 6; i++) {
      iwi = iwigroup(agents, i);
      iwi.mult(0.5);
      acc.add(iwi);
    }
  }


  PVector iwigroup(ArrayList agents, float type) {
    float count = 0.0;
    PVector sum = new PVector();
    for (int i = 0; i < agents.size(); i++) {
      Boid b = (Boid) agents.get(i);
      float d = pos.dist(b.pos);
      if (b.iwitype == type && iwitype == type) {
        float repulsion = 0;
        repulsion = (area + b.area) * 1.2;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
        }
      }
    }
    if (count > 0) {
      sum.div(count);
      return steer(sum);
    } else {
      return sum;
    }
  }


  PVector steer(PVector target) {
    PVector desired = PVector.sub(target, pos);
    desired.normalize();
    desired.mult(maxspeed);
    desired.sub(vel);
    desired.limit(maxforce);
    return desired;
  }


  void ageeffect (ArrayList agents) {
    PVector age = new PVector();
    age = agegrouping(agents);
    age.mult(0.2);
    acc.add(age);
  }


  PVector agegrouping (ArrayList agents) {
    float count = 0.0;
    float mult = 0;
    PVector sum = new PVector();
    for (int i = 0; i < agents.size(); i++) {
      Boid b = (Boid) agents.get(i);
      float d = pos.dist(b.pos);
      if (elderly == 2 && children == 0 && b.children ==
        || elderly == 2 && children == 0 && b.children =
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 2;
        }
      }
      if (elderly == 1 && children == 0 && b.children ==
        || elderly == 1 && children == 0 && b.children =
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 3;
        }
      }
      if (elderly == 0 && children == 2 && b.elderly ==
        || elderly == 0 && children == 2 && b.elderly ==
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 1;
        }
      }
      if (elderly == 0 && children == 1 && b.elderly ==
        || elderly == 0 && children == 1 && b.elderly ==
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 3;
        }
      }
      if (children == 1 && b.children == 1
        || children == 1 && b.children == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 3;
        }
      }
      if (children == 2 && b.children == 1
        || children == 2 && b.children == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 2;
        }
      }
      if (adults == 1 && b.adults == 1
        || adults == 1 && b.adults == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 3;
        }
      }
      if (adults == 2 && b.adults == 1
        || adults == 2 && b.adults == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 1;
        }
      }
      if (youngadults == 1 && b.youngadults == 1
        || youngadults == 1 && b.youngadults == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 2;
        }
      }
      if (youngadults == 2 && b.youngadults == 1
        || youngadults == 2 && b.youngadults == 2) {
        float repulsion = 0;
        repulsion = (area > b.area) ? area : b.area;
        if (d > repulsion && d < neighbourhood) {
          count+=1.0;
          sum.add(b.pos);
          mult = 1;
        }
      }
```

```
    if (elderly == 1 && adults == 0 && b.adults == 1
      || elderly == 1 && adults == 0 && b.adults == 2
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 3;
      }
    }
    if (elderly == 2 && adults == 0 && b.adults == 1
      || elderly == 2 && adults == 0 && b.adults == 2
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 2;
      }
    }
    if (children == 1 && adults == 0 && b.adults == 1
      || children == 1 && adults == 0 && b.adults ==
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 2;
      }
    }
    if (children == 2 && adults == 0 && b.adults == 1
      || children == 2 && adults == 0 && b.adults ==
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < neighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 3;
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steerunit(sum, mult);
  } else {
    return sum;
  }
}


void relationseffect (ArrayList agents) {
  PVector rel = new PVector();
  for (int i = 1; i < 11; i++) {
    rel = relationgrouping(agents, i);
    rel.mult(0.2);
    acc.add(rel);
  }
}


PVector relationgrouping (ArrayList agents, float typ
  float count = 0.0;
  float mult = 0;
  PVector sum = new PVector();
  for (int i = 0; i < agents.size(); i++) {
    Boid b = (Boid) agents.get(i);
    float d = pos.dist(b.pos);
    if (family == type && b.family == type
      || family == type && b.afamily == type
      || afamily == type && b.family == type
      || afamily == type && b.afamily == type) {
      float repulsion = 0;
      repulsion = (area > b.area) ? area : b.area;
      if (d > repulsion && d < famneighbourhood) {
        count+=1.0;
        sum.add(b.pos);
        mult = 4;
      }
    }
  }
  if (count > 0) {
    sum.div(count);
    return steerunit(sum, mult);
  } else {
    return sum;
  }
}


//  void familytypeeffect (ArrayList agents) {
//    PVector fam = new PVector();
//    fam = familygrouping(agents);
//    fam.mult(0.6);
//    acc.add(fam);
//  }


//  PVector familygrouping (ArrayList agents) {
//    float count = 0.0;
//    float mult = 0;
//    PVector sum = new PVector();
//    for (int i = 0; i < agents.size(); i++) {
//      Boid b = (Boid) agents.get(i);
//      float d = pos.dist(b.pos);
//      if (familytype == 1 && b.familytype == 3
//        || familytype == 3 && b.familytype == 9
//        || familytype == 5 && b.familytype == 2) {
//        float repulsion = 0;
//        repulsion = (area > b.area) ? area : b.area;
//        if (d > repulsion && d < neighbourhood) {
//          count+=1.0;
//          sum.add(b.pos);
//          mult = 3;
//        }
//      }
//      if (familytype == 2 && b.familytype == 3
//        || familytype == 4 && b.familytype == 8) {
//        float repulsion = 0;
//        repulsion = (area > b.area) ? area : b.area;
//        if (d > repulsion && d < neighbourhood) {
//          count+=1.0;
//          sum.add(b.pos);
//          mult = 2.5;
//        }
//      }
//      if (familytype == 6 && b.familytype == 6) {
//        float repulsion = 0;
//        repulsion = (area > b.area) ? area : b.area;
//        if (d > repulsion && d < neighbourhood) {
//          count+=1.0;
//          sum.add(b.pos);
//          mult = 2;
//        }
//      }
//      if (familytype == 2 && b.familytype == 4
//        || familytype == 7 && b.familytype == 9) {
//        float repulsion = 0;
//        repulsion = (area > b.area) ? area : b.area;
//        if (d > repulsion && d < neighbourhood) {
//          count+=1.0;
//          sum.add(b.pos);
//          mult = 1.5;
//        }
//      }
//      if (familytype == 1 && b.familytype == 6) {
//        float repulsion = 0;
//        repulsion = (area > b.area) ? area : b.area;
//        if (d > repulsion && d < neighbourhood) {
//          count+=1.0;
//          sum.add(b.pos);
//          mult = 1;
//        }
//      }
//    }
//    if (count > 0) {
//      sum.div(count);
//      return steerfamily(sum, mult);
//    } else {
//      return sum;
//    }
//  }


PVector steerunit(PVector target, float familymultfa
  PVector desired = PVector.sub(target, pos);
  desired.normalize();
  desired.mult(maxspeed);
  desired.sub(vel);
  desired.limit(maxforce);
  desired.mult(familymultfactor);
  return desired;
}


void update (ArrayList agents) {
  flock(agents);
  vel.add(acc);
  pos.add(vel);
}


void render () {
  smooth();
  stroke (200, 255, 200);
  point(pos.x, pos.y, pos.z);
  text(familytype, (pos.x + 10), pos.y, pos.z);
  strokeWeight(2);
  if (iwitype == 99) {
    stroke(100);
  } else if (iwitype == 1) {
    stroke(c_IT1);
  } else if (iwitype == 2) {
    stroke(c_IT2);
  } else if (iwitype == 3) {
    stroke(c_IT3);
  } else if (iwitype == 4) {
    stroke(c_IT4);
  } else if (iwitype == 5) {
    stroke(c_IT5);
  } else if (iwitype == 99) {
    stroke(c_IT6);
  }
  noFill();
  pushMatrix();
  translate(pos.x, pos.y, pos.z);
  sphere(area / publicspace);
  popMatrix();
}


void keyPressed (ArrayList agents) {
  if (key == 'e' ) {
    table = new Table();
    table2 = new Table();

    table.addColumn("ID");
    table.addColumn("x");
    table.addColumn("y");
    table.addColumn("z");
    table.addColumn("area");

    table2.addColumn("ID");
    table2.addColumn("x");
    table2.addColumn("y");
    table2.addColumn("z");
    table2.addColumn("family type");
    table2.addColumn("iwi type");
    table2.addColumn("house type");
    table2.addColumn("area");
    table2.addColumn("multimportance");
    table2.addColumn("children");
    table2.addColumn("elderly");
    table2.addColumn("dependantpersons");
    table2.addColumn("shortterm");
    table2.addColumn("longterm");
    table2.addColumn("adults");
    table2.addColumn("youngadults");
    table2.addColumn("family");
    table2.addColumn("efamily");
    table2.addColumn("housesize");

    for (int i = 0; i < agents.size(); i++ ) {
      Boid agent = (Boid) agents.get(i);
      TableRow newRow = table.addRow();
      newRow.setInt("ID", agent.agentcode);
      newRow.setFloat("x", agent.pos.x);
      newRow.setFloat("y", agent.pos.y);
      newRow.setFloat("z", agent.pos.z);
      newRow.setFloat("area", agent.area);
      newRow.setFloat("multimportance", agent.centrali
      newRow.setFloat("children", agent.children);
      newRow.setFloat("elderly", agent.elderly);
      newRow.setFloat("dependantpersons", agent.depend
      newRow.setFloat("shortterm", agent.shortterm);
      newRow.setFloat("longterm", agent.longterm);
      newRow.setFloat("adults", agent.adults);
      newRow.setFloat("youngadults", agent.youngadults
      newRow.setFloat("family", agent.family);
      newRow.setFloat("efamily", agent.afamily);

      saveTable(table, "data/location.csv");
    }

    for (int i = 0; i < agents.size(); i++ ) {
      Boid agent = (Boid) agents.get(i);
      TableRow newRow = table2.addRow();
      newRow.setInt("ID", agent.agentcode);
      newRow.setFloat("x", agent.pos.x);
      newRow.setFloat("y", agent.pos.y);
      newRow.setFloat("z", agent.pos.z);
      newRow.setFloat("family type", agent.familytype)
      newRow.setFloat("iwi type", agent.iwitype);
      newRow.setFloat("house type", agent.housetype);
      newRow.setFloat("area", agent.area);
      newRow.setFloat("multimportance", agent.centrali
      newRow.setFloat("children", agent.children);
      newRow.setFloat("elderly", agent.elderly);
      newRow.setFloat("dependantpersons", agent.depend
      newRow.setFloat("shortterm", agent.shortterm);
      newRow.setFloat("longterm", agent.longterm);
      newRow.setFloat("adults", agent.adults);
      newRow.setFloat("youngadults", agent.youngadults
      newRow.setFloat("family", agent.family);
      newRow.setFloat("efamily", agent.afamily);

      saveTable(table2, "data/fulldata.csv");
    }
  }
}


//excel limits
float numIwi = 5;

//unit values
float publicspace = 1.3;

//neighbourhood values
float neighbourhood = 300;
float famneighbourhood = 600;


//colours
color c_IT1 = #FFB7B7;
color c_IT2 = #FFDAB7;
color c_IT3 = #FFFCB7;
color c_IT4 = #C3FFB7;
color c_IT5 = #B7D8FF;
color c_IT6 = #DDB7FF;

//box
PVector point1 = new PVector (200, 100);
PVector point2 = new PVector (600, 600);


class environment {

  ArrayList agents;

  environment () {
    initboids();
  }

  void initboids () {
    int agentNum;
    int houseArea;
    int iwiNum;
    int family;
    int additionalFamily;
    int children;
    int youngAdults;
    int adults;
    int elderly;
    int dependantPersons;

    int familyType;
    int houseType;

    int centralImportance;
    int shortTerm;
    int longTerm;

    agents = new ArrayList ();

    String lines[] = loadStrings("2D - Agents.csv");
    String [][] csv;
    int csvWidth=0;

    for (int i=0; i < lines.length; i++) {
      String [] chars=split(lines[i], ',');
      if (chars.length>csvWidth) {
        csvWidth=chars.length;
      }
    }

    csv = new String [lines.length][csvWidth];

    for (int i=0; i < lines.length; i++) {
      String [] temp = new String [lines.length];
      temp= split(lines[i], ',');
      for (int j=0; j < temp.length; j++) {
        csv[i][j]=temp[j];
      }
    }

    for (int i=1; i < lines.length; i++) {
      agentNum = int(csv[i][0]);
      houseArea = int(csv[i][1]);
      iwiNum = int(csv[i][2]);
      family = int(csv[i][3]);
      additionalFamily = int(csv[i][4]);
      children = int(csv[i][5]);
      youngAdults = int(csv[i][6]);
      adults = int(csv[i][7]);
      elderly = int(csv[i][8]);
      dependantPersons = int(csv[i][9]);
      familyType = int(csv[i][11]);
      houseType = int(csv[i][12]);
      centralImportance= int(csv[i][14]);
      shortTerm = int(csv[i][15]);
      longTerm = int(csv[i][16]);
      Boid point = new Boid(agentNum, houseArea, iwiNum, f
        children, youngAdults, adults, elderly, dependantP
        centralImportance, shortTerm, longTerm);
      agents.add(point);
    }
  }

  void run () {
    update();
  }


  void update() {
    for (int i = 0; i < agents.size(); i++) {
      Boid agent = (Boid) agents.get(i);
      agent.run(agents);
    }
  }
}
```

Figure 69: Code of the Final Product

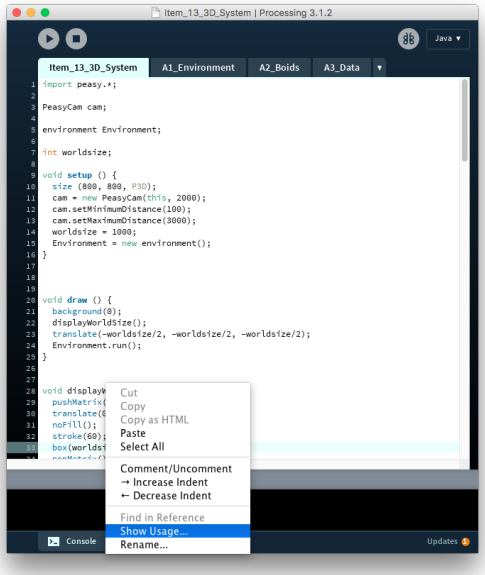
Final product is Appendix Item 12 including instructions for use and adaptation.

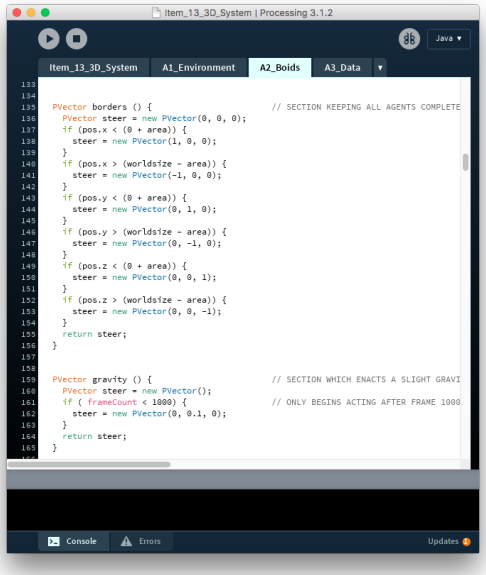
RUNNING THE FINAL SYSTEM

The final system provides the design methodology from which the spatial organisation of a Māori community can be created, and acts as the final output of this research, therefore understanding how this system operates is the key to functionality.

Firstly the freeware application Processing must be installed to run the script, as the Appendix Item 12 file is run through the Processing platform. The folder of Item 12 contains a file of the same name with the .pde extension, running this through Processing provides the exact simulation variables as which created Figure 69. Please note, sketch will not run without the peasycam library loaded, this can be accessed from Sketch > Import Library > Add Library, then searching and downloading peasycam before running.

The following section details how the inputs can be adjusted in order to simulate any given scenarios, rather than just the scenario with which I have been working, creating a methodology for spatial design rather than an application for a single project.





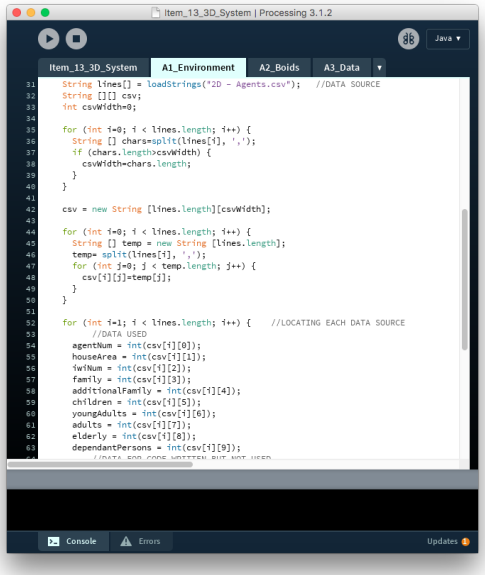
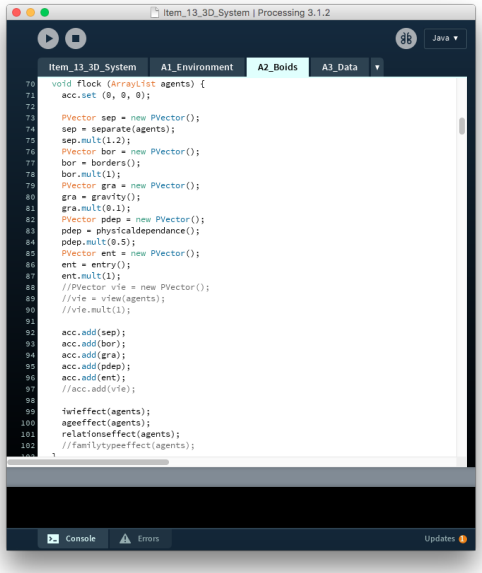
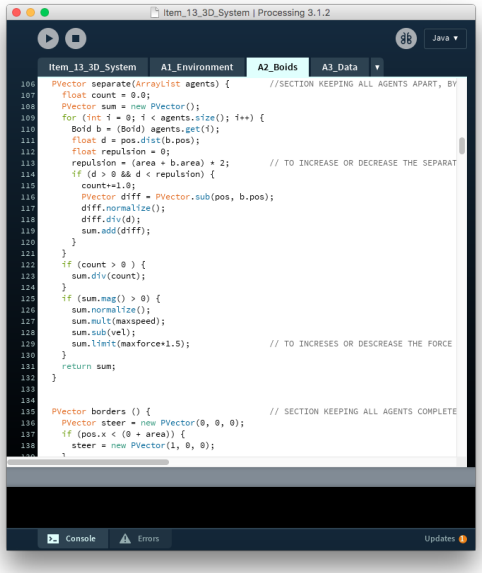


Figure 70, 71, and 72: Input Variables

The input specification of the system are contained within the original sketch tab (70), and the A1_Environment tab (72). Figure 70 indicates where the environment can be edited, replacing the 'worldsize' order with x, y, and z figures representing the constraint of the new environment, and Figure 71 indicates the code that dictates agents remain within these confines (replace 'worldsize' with the new x, y, and z figures in the 'PVector borders()' command in Line 135 of the A2_Boids tab). Figure 72 shows where the agent data is sourced from. Line 31 of the A1_Environment tab calls the csv file "2D Agents.csv" located in the data folder of the sketch (Appendix Item 11). This csv file contains a list of all the agents, as well as their corresponding traits, and can be edited as desired, as long as the sketch name remains the same.





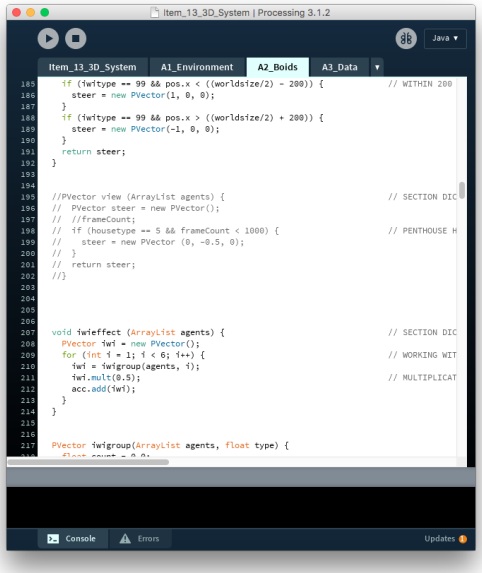
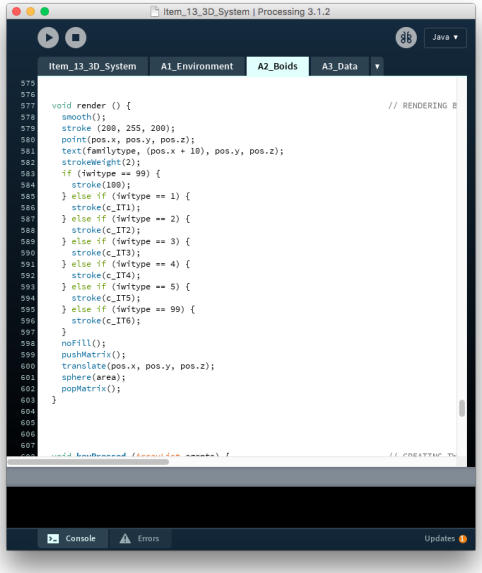
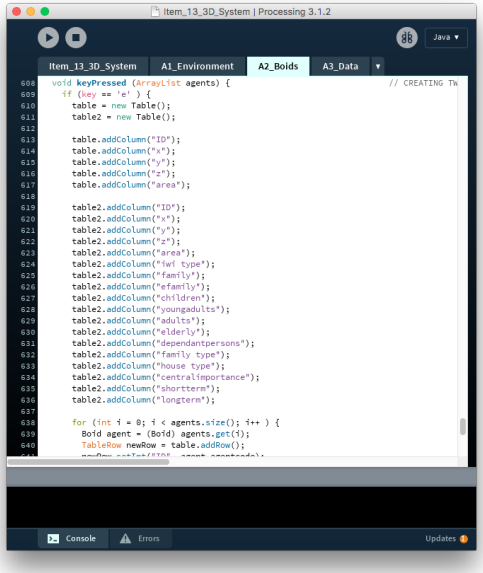


Figure 73, 74, and 75: Behavioural Coding

The A2_Boids tab has full control over the scripted behaviours and relationships of the agents in the system. Lines 70 - 113 call all behaviours that are active in the system (73), and Lines 116 - 575 describe those behaviours (74). Within the system a few of the behaviours are not currently active (75), these are proceeded by '//', if these are desired they may be re introduced by the removal of the '//' prefix for the code, and the initial call to function. The script itself contains information on all the behaviours, as well as how to edit them, and by analysing the different functions and their meanings the formula for construction becomes apparent, enabling the addition of new behaviours if desired.





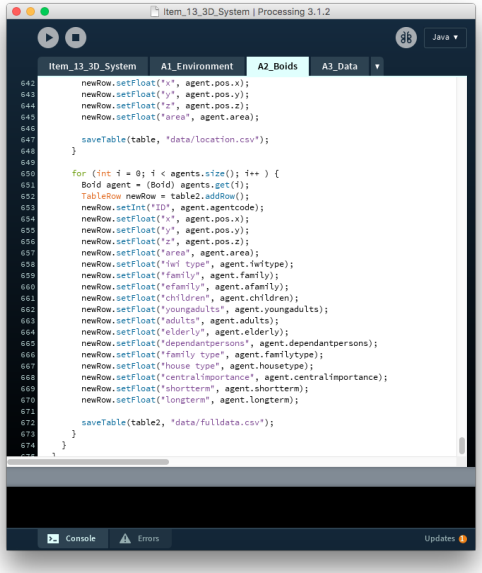


Figure 76, 77, and 78: Output Directives

These images identify the are in code where visual and numeric outputs are described. Lines 577 - 603 dictate the visual outputs of this system (76), the provided code dictating spherical units of the mass of the area input, colour coded in respect to iwi, however each of these variables could be substituted for another, or removed entirely. Figures 77 and 78 dictate the should the key "e" be pressed during simulation, two output files shall be created in the data folder of the sketch, one of which has basic information relating to the location and unit size of each agent, the other including all input variables also. It must be noted that each time the key is pressed, the created file will overwrite any previous iterations with the same name, therefore file names should be changed to be kept.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	ID	x	y	z	family type	iwi type	house type	area	multimporta	children	elderly	dependantpe	shortterm	longterm	adults	youngadults	family	efamily
2	1	171.04053	81.55986	338.866	1	1	1	79.78845	0	0	1	0	0	0	0	0	1	0
3	2	892.2755	113.05993	865.9268	1	3	2	97.7205	1	1	0	0	0	0	2	0	4	0
4	3	113.13774	860.0994	233.80318	1	1	1	112.83791	0	2	0	0	1	0	2	0	1	0
5	4	899.3625	635.90594	97.15427	2	4	3	97.7205	1	1	2	0	0	0	0	0	6	0
6	5	533.11847	885.6686	395.21054	2	1	2	112.83791	0	2	1	1	0	0	2	1	5	0
7	6	113.51849	492.63193	445.94217	2	2	3	97.7205	0	0	0	0	0	0	2	0	1	2
8	7	487.59152	163.92084	112.15345	2	3	4	112.83791	1	0	1	0	0	0	1	1	3	0
9	8	595.64355	558.2118	708.13153	2	2	2	79.78845	0	0	0	0	0	0	2	1	2	0
10	9	873.5071	289.07626	710.47046	3	3	2	112.83791	0	1	0	0	0	0	2	2	4	0
11	10	393.02106	576.34393	98.80359	3	1	1	97.7205	0	2	0	0	0	0	1	0	1	0
12	11	884.02185	887.05585	115.40133	3	4	3	112.83791	0	1	2	1	0	0	1	1	6	0
13	12	351.60883	517.9149	570.1346	4	2	2	97.7205	1	0	0	0	0	0	1	0	2	0
14	13	513.34753	114.414696	516.9884	4	2	1	112.83791	0	2	0	0	0	0	2	1	7	0
15	14	919.1837	80.40492	609.41125	5	3	5	79.78845	2	0	2	0	0	0	0	0	4	0
16	15	472.24924	118.859314	887.51306	5	5	4	112.83791	1	1	0	1	1	0	2	0	8	7
17	16	92.5514	493.55923	903.4605	5	1	2	97.7205	1	1	0	0	0	0	1	1	9	0
18	17	112.72147	115.42194	721.2579	5	2	2	112.83791	0	2	0	0	0	0	2	1	7	0
19	18	895.95575	103.2443	250.80055	5	5	3	97.7205	0	0	0	0	0	0	2	2	8	0
20	19	720.20636	475.59857	350.0316	5	5	1	112.83791	0	0	2	0	0	0	0	0	8	0
21	20	94.277565	326.91666	96.243225	6	2	4	97.7205	1	1	1	0	0	0	0	1	10	0
22	21	348.85	885.73883	862.1677	99	99	99	138.19766	99	99	99	99	99	99	99	99	99	99

Figure 79: Data Files

The output data files contain all information required to recreate the simulation in an editable environment, in order to address design issues outside of the scope of this methodology. The supplied information is appropriate for the design of a settlement using any second methodology desired, providing the locations and sizes of housing units for specific agents, as well as including the initial housing unit profile information which may be of use for more specific design stages.

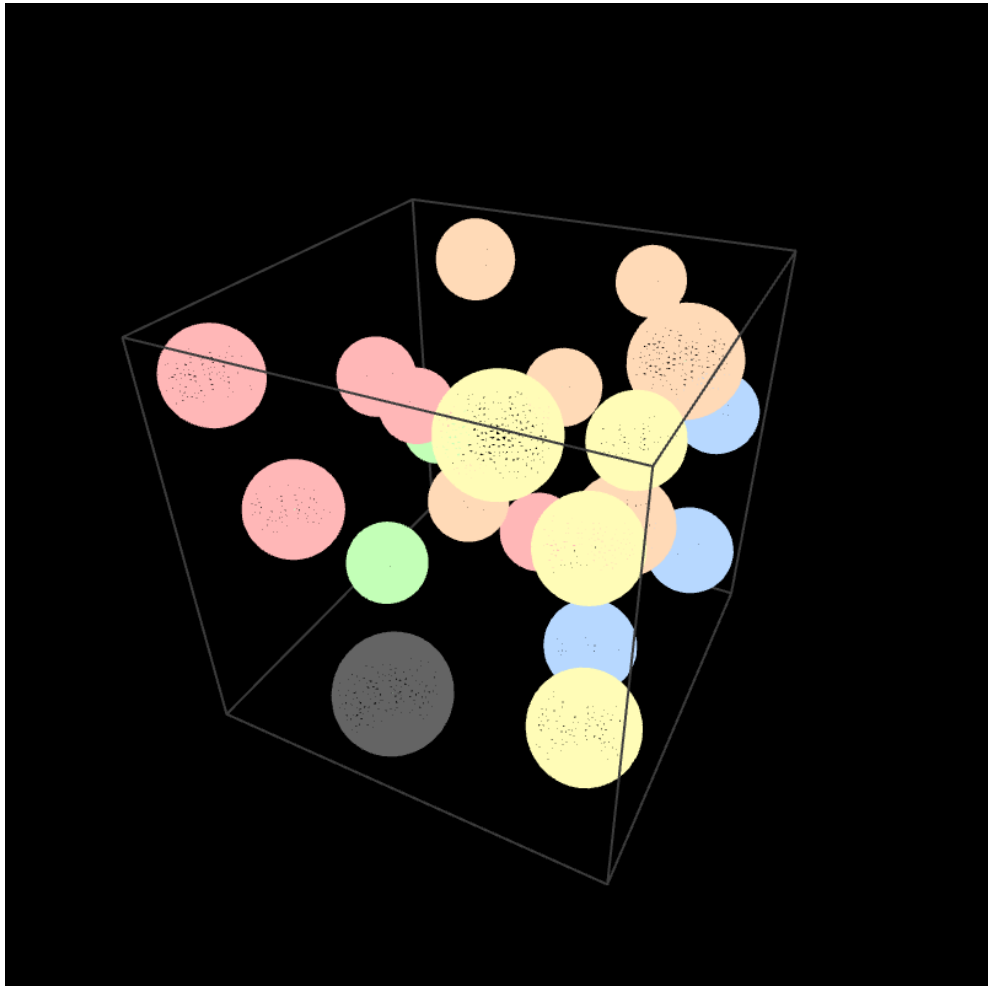


Figure 80: Final Product of System
The final visual simulation of the relationship structuring this methodology produces.
This simulation has accompanying data files for the repurposing of information for further stages in a design process.
See Appendix Item 12 for system.

The use of this designed system creates two different forms of output, both describing spatial relationships and occupation, yet neither appearing to be particularly complex or architectural in nature.
The first output is a basic visual simulation of these relationships, with spheres acting as units, the mass of which corresponds to the area each unit would require, and with the function to be colour coded based on which relationship variable needs to be analysed. This simulated output is an example of why architects rarely use processing as a platform for design, as the visual results are limited in terms of aesthetic manipulation. However, if read correctly there is no reason why this tool cannot inform the structure of design, and provide a methodology for determining functionality, before aesthetic considerations come in to action.
The second output of this system is a set of data, detailing the exact variables of each unit, as described by the visual simulation. This output provides the means to recreate the relationships in any desired format by which architectural realisation can take place, with no stylists biases associated with the data.

Essentially this system offers a method of providing location data to construct desired architectural relationships with no preconceived notion of how the architecture should be formed, making it adaptable to any culture, any place, and any time. As long as the desired relationship structures are known, a spatial representation can be simulated and then explored through architecture.

Part III - Results

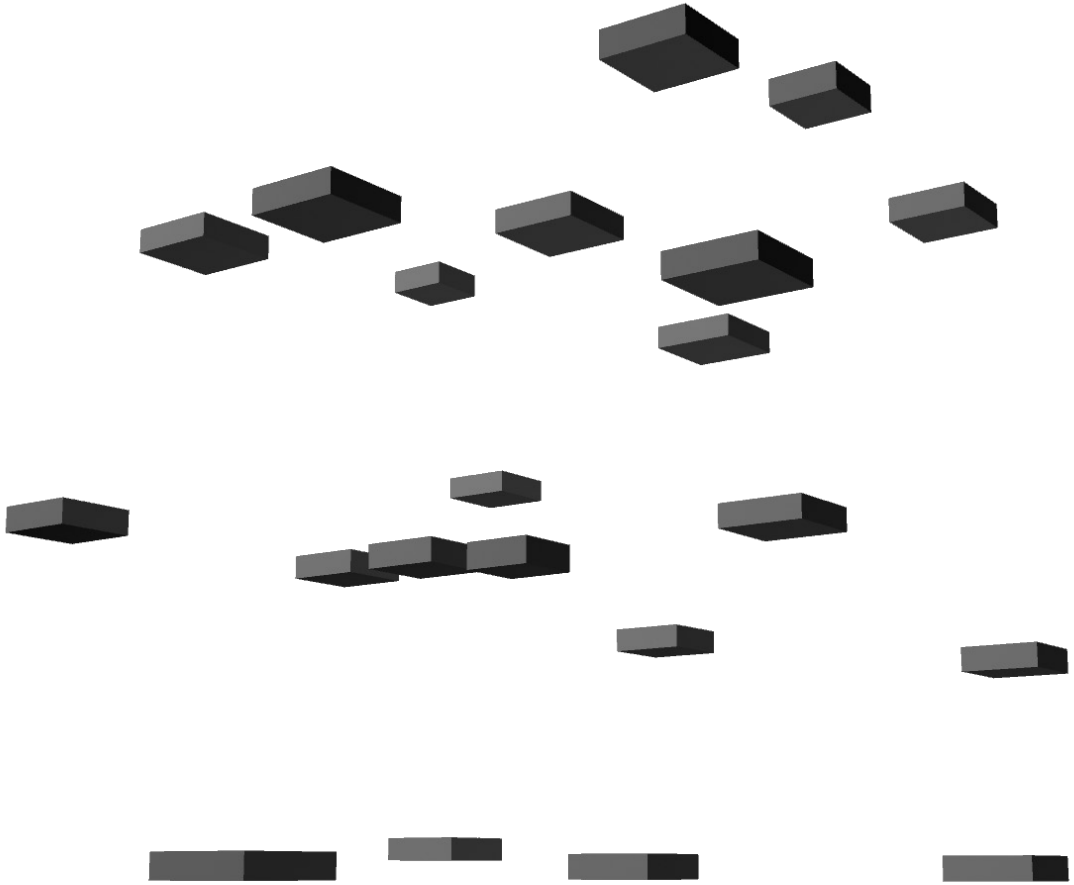
CHAPTER EIGHT:

INTERPRETING OUTPUT

As the output takes the form of a visual simulation and data series, it can be quite difficult to comprehend that it is in fact informed by the relationship systems set in place. Therefore in order to visualise the active connections between the units, further simulations have been carried out. This interpretation could be carried out in Processing, or in other applications like Autodesk Maya, using nParticle simulation techniques. Both simulations have been carried out in order to visualise different aspects of the system.

Processing has been used to visualise the strength of the relationships at their final locations, simulating the network of iwi, familial and generational relationships in action. Maya has been used to simulate which relationships are the most meaningful to each unit, by using springs between units to draw them closer to other units with which they have the strongest relationships. This example is no longer a precise simulation of the desired relationship structure, rather a manipulated simulation where the degree of separation is not enforced as absolute, and compounding degrees of attraction can operate at a greater extent than in the original simulation.

Figure 81: Final Data Use
The data produced can be input into any medium from which relationship structuring can be replicated for further design stages.



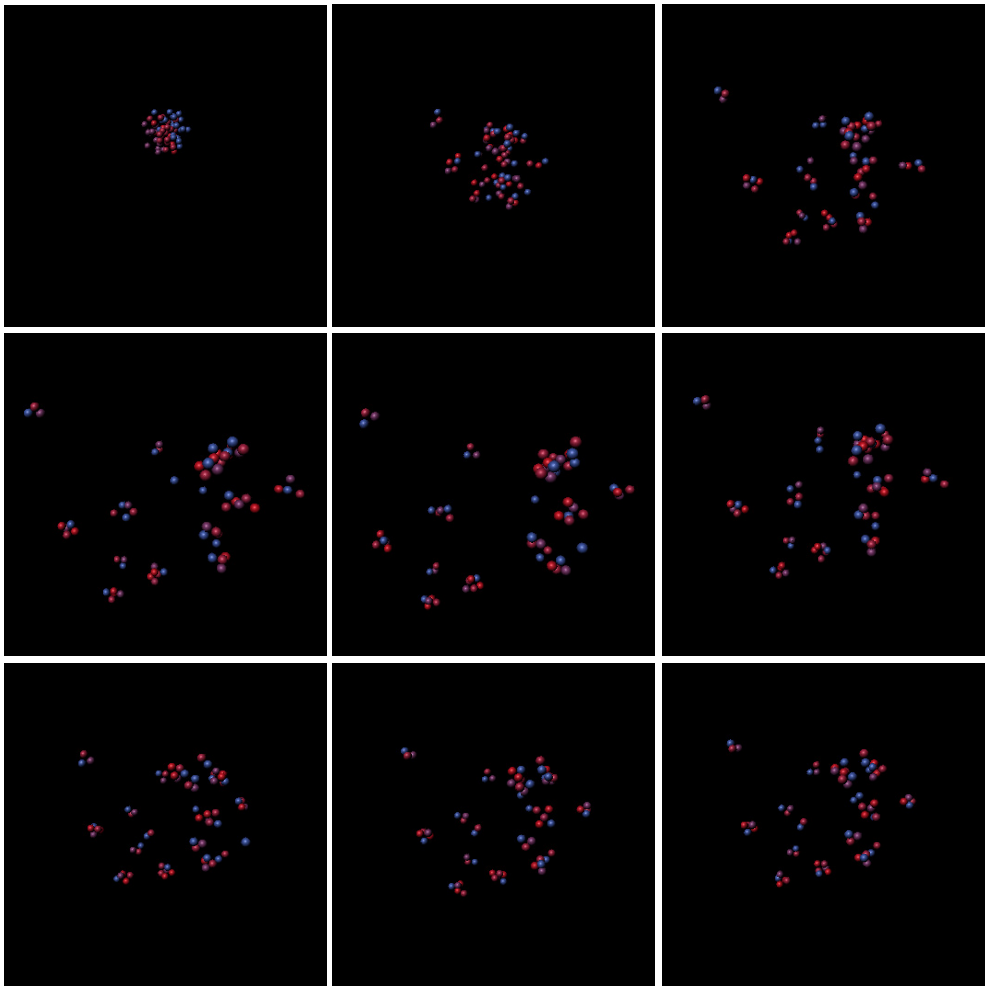
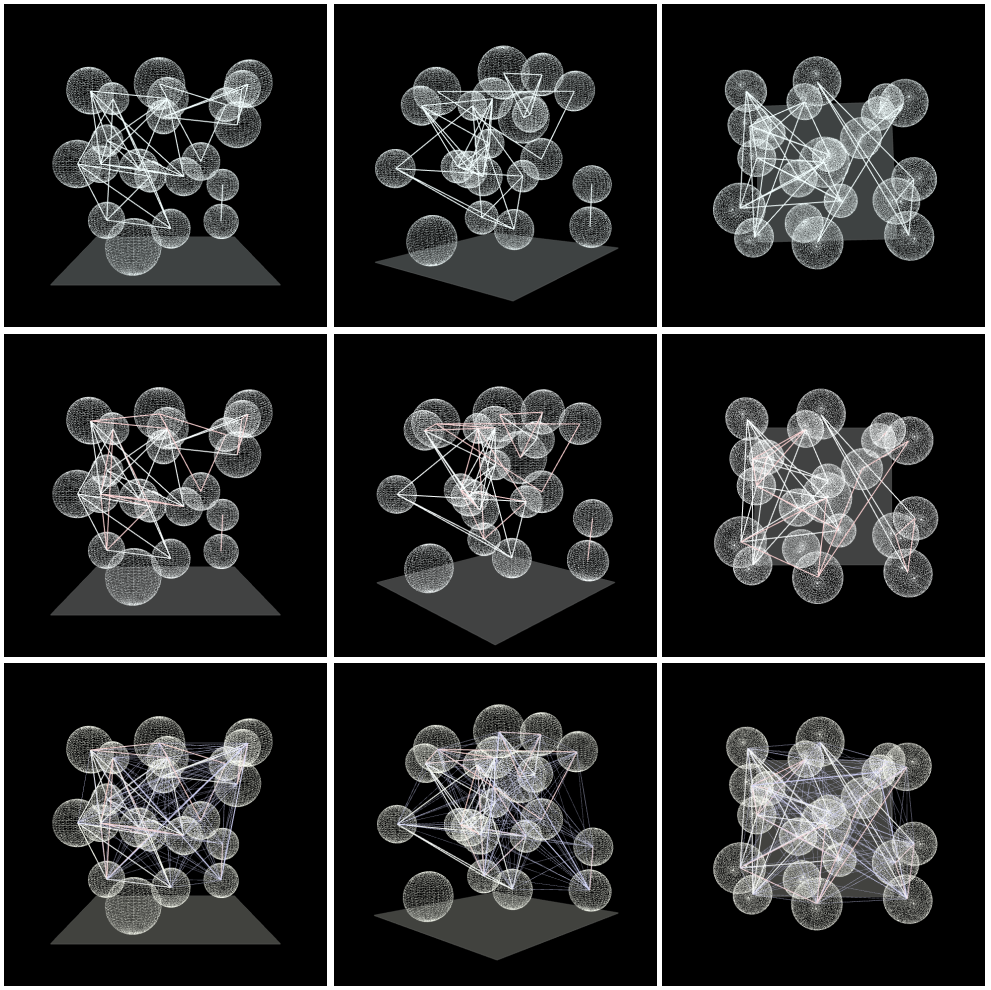


Figure 82a - i (L to R, Top to Bottom):
Maya Spring Dynamics
Inputting the data and relationships into Maya nParticle spring dynamics identifies which relationships are the strongest by which offers the most deviation from location. When all relationships are in effect, certain units will be more likely than others to associate with units a great distance away, and this is a way of testing the degree to which this occurs. If deviation is too large then the initial system simulation needs to be run again, allowing more time for units to align themselves with precisely who they wish. Rendered outputs of the system do not identify the springs in action, rather the effect, therefore Appendix Item 13 has been included to demonstrate how the springs function.

Figure 83a - i (L to R, Top to Bottom): Relationship Identification

As a means of evaluating the data output, the scene has also been reintroduced to processing and the relationships which initially organised the system have been identified. By identifying the relationships in layers we can see which has the greatest bearing on the scene. The first layer identifies the iwi relationships, the second the familial relationships, and the third the relationships based upon generational aspects. It becomes apparent that the generationally driven relationships provide the most complication to the system and are likely to be what has the greatest bearing in terms of intricacy of location, whereas iwi and familial relationships inform the more generalised setting.



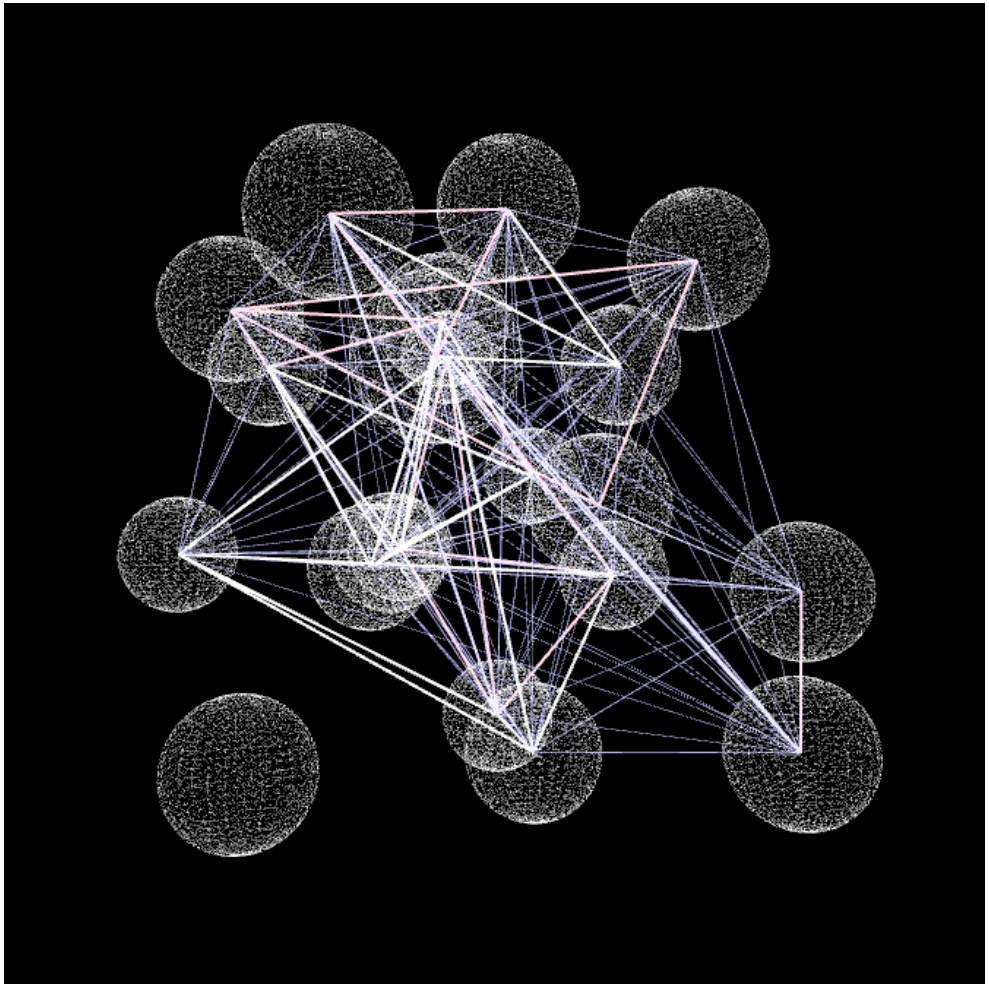
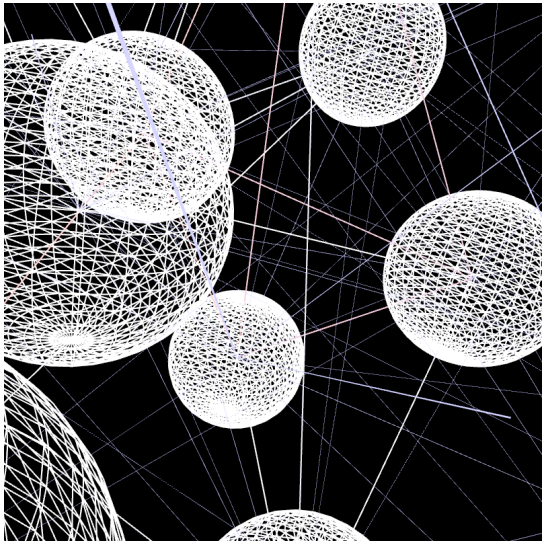
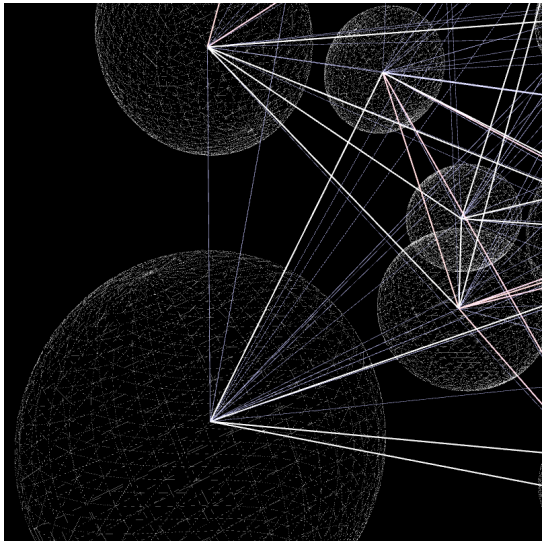
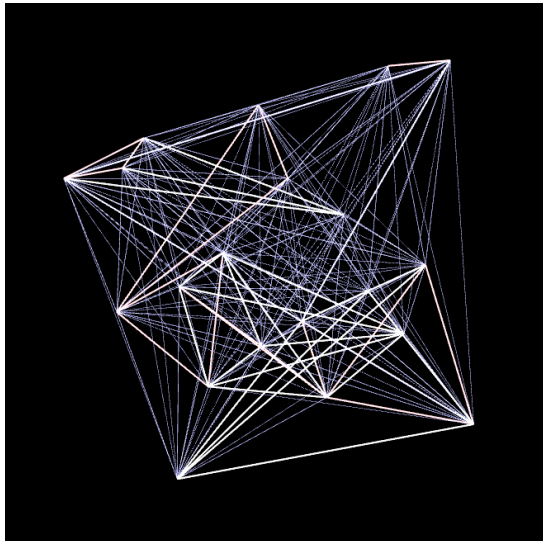


Figure 84, 85 a - c (L to R): Final Relationship Network
This simulation illustrates the final relationship structuring of the hypothetical settlement with which I have been working, and identifies the relationships that have been structured within it. The degree of influence of each relationship to each unit dictates the strength of each thread in the system, creating a system that uses a small amount of specific information to predict a great amount in terms of human behaviour through space.
84. Network as a whole
85a - c. Network with varying degrees of focus on spatial occupation and relationship establishment.

The final processing images show how the matrix of behaviours taken from Māori cultural values can be transformed into a network of relationships with spatial implications that can then be translated into architecture. Just as processing systems can model traffic systems that are translated into urban interventions, they can simulate cultural relationships that can be translated into built form.







Part III - Results

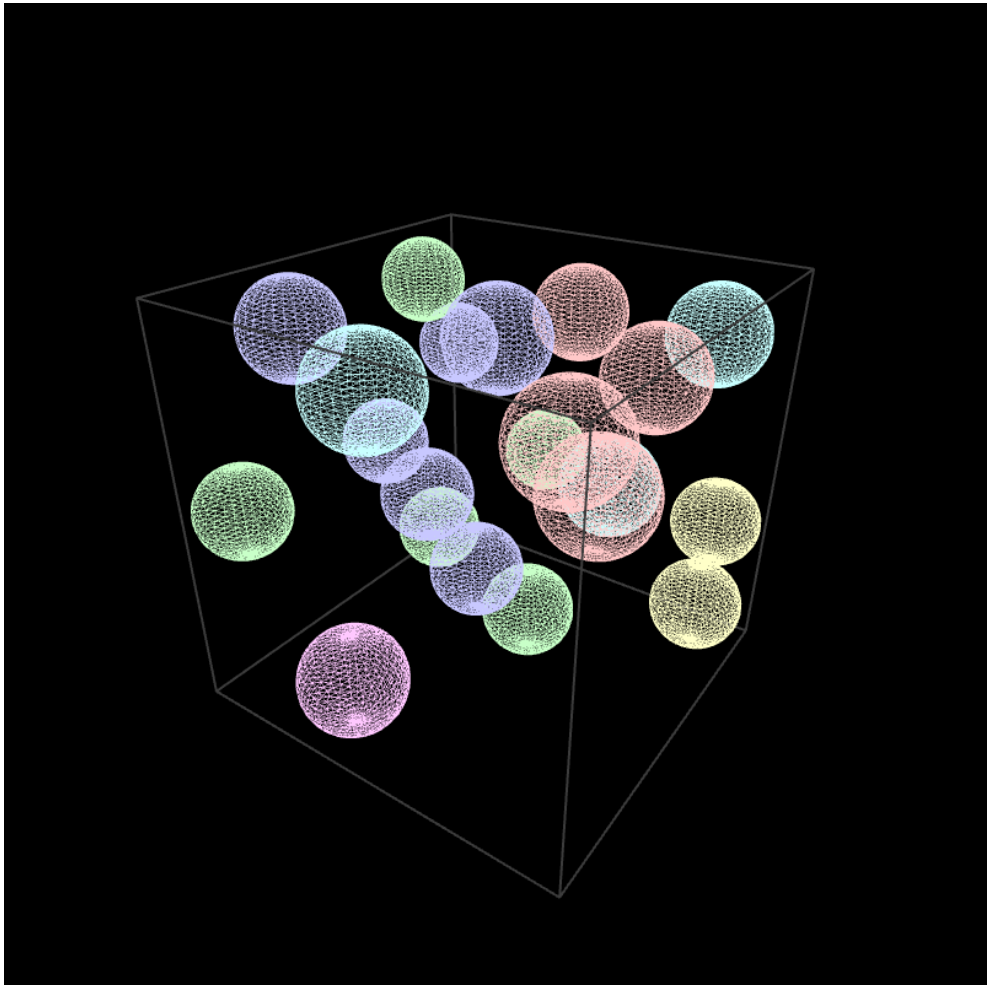
CHAPTER NINE:

ARCHITECTURAL IMPLICATIONS

The significant architectural implications that emerge from this research are twofold; there are implications on the extent of architectural design and the idea of architecture as an organisation system, and implications in the form of a design tool.

It is commonly expected that architectural design will produce structure through form, however this need not always be the case. Why be limited to producing structure through form when you can produce structure through a system that can then create multiple forms, and inform numerous other systems by which more successful interpretations of architecture can be produced? This research looks to realise architecture with an output based on the latter idea, therefore the architectural implications are abstract examples of architecture rather than traditional in nature.

Figure 86: Direct Output
The raw imaging and data that describe it are the direct output, through correct interpretation these become useful as architectural tools.



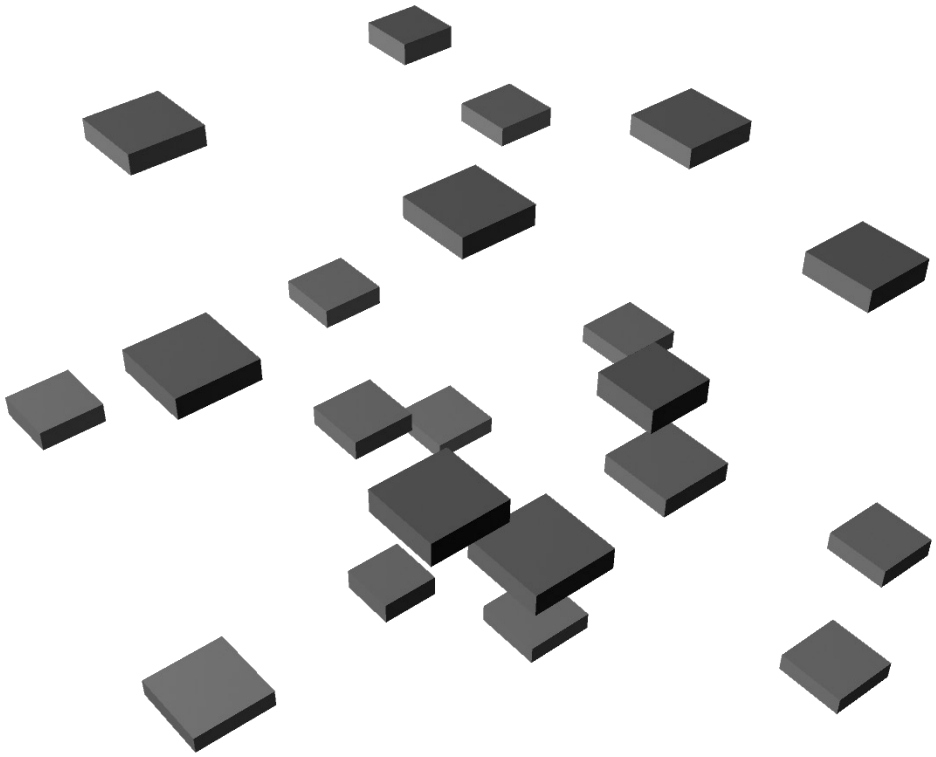


Figure 87: Interpreted in a Traditional Format
In order to create a more traditional format to design with, the data needs to be understood in terms of how it is translated to more workable and editable formats, like that of SketchUp and Maya which have been implemented in this project. By identifying two of the extremes in terms of computer programming capability, the more user friendly, free-ware of SketchUp, compared to the more advanced, professional software of Maya, it is clear that this type of data production does not limit design preferences.

AS A DESIGN SYSTEM

The product of this investigation is a style-less architecture, one that cannot be interpreted as having particular connection to a time or place, one that is entirely open to interpretation. This is essential to ensuring that as a tool, there is no bias to any specific type of architectural project that may devalue this as a source of knowledge to any particular scenario; the only influences on the system are the specific relationships from which it is built.
This challenges the idea of what motivates architectural design, and at which point in the process does architecture emerge. Through this research you could argue that architecture emerges at the point where structure is formed in relation to spatial considerations. Some may argue that in order for structure to emerge there must be an associated form, however form is subjective, and where perception plays a part, so does bias, and a biased system of design does not make for an adaptable tool, indicating that form can devalue architecture in terms of the creation of design methodology and practice.

I argue that this investigation has inadvertently been an exploration into open architecture, without association to form or constraints, and challenging preconceived ideas of architectural understanding. The systems that create architecture are indeed part of architecture itself, and they are too often underrated in comparison to the visible form of architectural products.

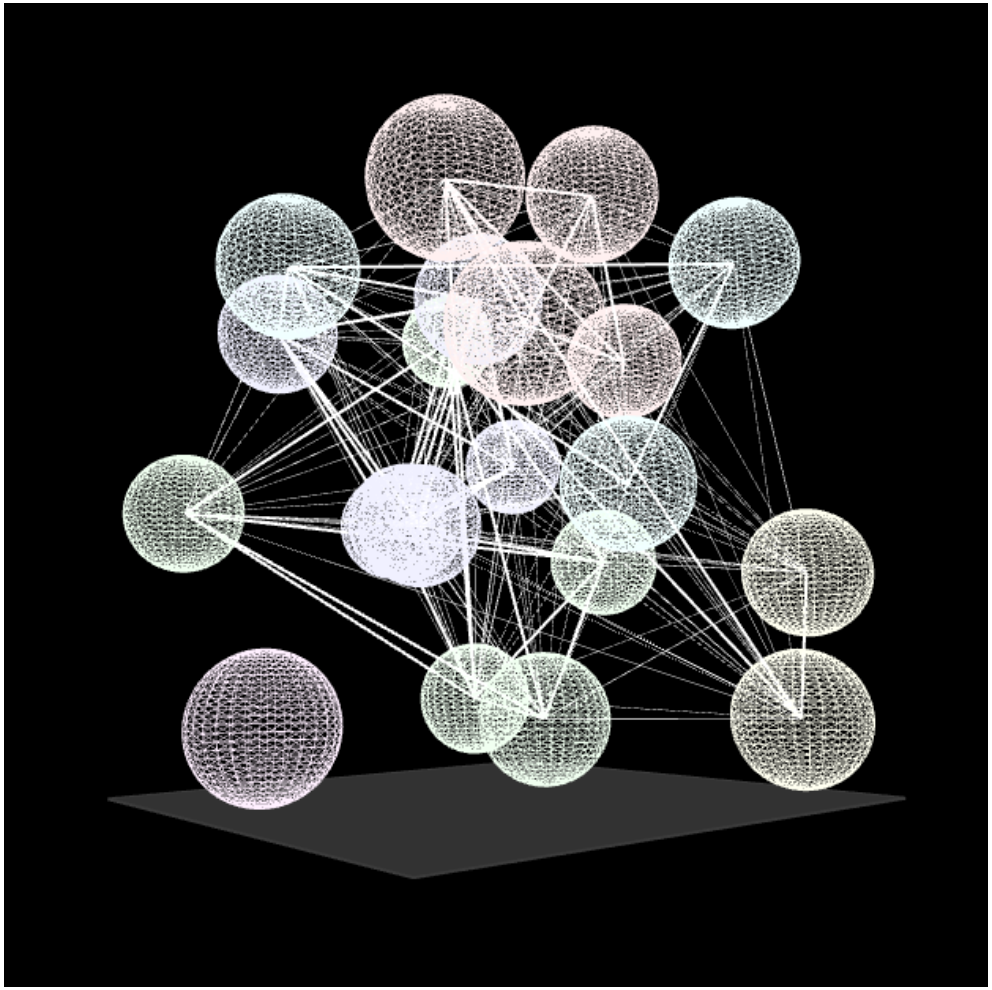
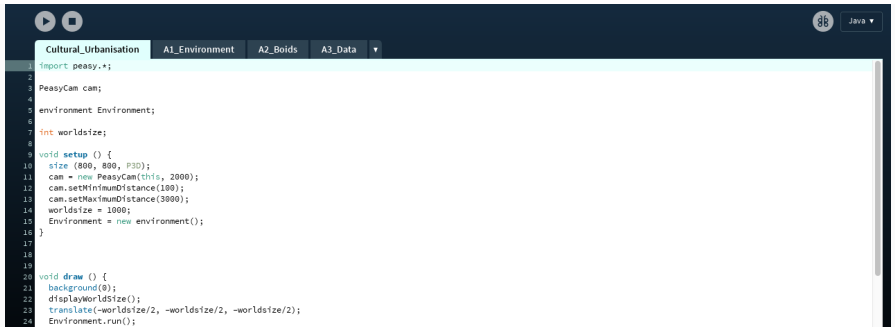


Figure 88, 89: Network Model and Design Tool
88. Demonstrates how the system acts as a networking tool. The ability to construct relationship markers is included as Appendix Item 14 and 15. Item 15 allows relationships to be monitored as they are formed, and Item 14 provides the ability to identify relationships in previously established spatial organisations.
89. Offers an insight into the user-face of the system, using Excel csv files in conjunction with Processing to create all the required data from relationship modelling.

AS A TOOL

This system is intended to operate as a digital tool for architectural design, it is the purpose for which it was built, and hence why it has been honed to the point at which flexibility and ease of use are paramount. The design of digital architectural tools is both an ever evolving, and very limited field of design. It is ever evolving in that new programs, plug-ins and scripts are being produced every day, however it is limited in that the majority of these are made with western design in mind. Preprogramed design tools are generally scripted in a way that caters to designs of western influences, looking into aspects concerning form, building science and other similar western focuses. Very few however concern themselves with relationship structures as this tool does. As architecture is as much about the user as it is about what is built, this tool could prove to be very effective in simulating user behavioural relationships that operate within a community, before the form is built. The tool and method of use is included as Appendix Item 12 of this document in the hope that it may be implemented, or extended for further use.

	A	B	C	D	E	F	G	H	
1	Agent Number	Area	Iwi Number	Family	Additional Family	Children	Young Adults	Adults	Eld
2	1	50	1	1	0	0	0	0	
3	2	75	3	4	0	1	0	2	
4	3	100	1	1	0	2	0	2	
5	4	75	4	6	0	1	0	0	
6	5	100	1	5	0	2	1	2	
7	6	75	2	1	2	0	0	2	
8	7	100	3	3	0	0	1	1	
9	8	50	2	2	0	0	1	2	
10	9	100	3	4	0	1	2	2	
11	10	75	1	1	0	2	0	1	
12	11	100	4	6	0	1	1	1	
13	12	75	2	2	0	0	0	1	
14	13	100	2	7	0	2	1	2	
15	14	50	3	4	0	0	0	0	
16	15	100	5	8	7	1	0	2	
17	16	75	1	9	0	1	1	1	
18	17	100	2	7	0	2	1	2	
19	18	75	5	8	0	0	2	2	
20	19	100	5	8	0	0	0	0	
21	20	75	2	10	0	1	1	0	
22	21	150	99	99	99	99	99	99	



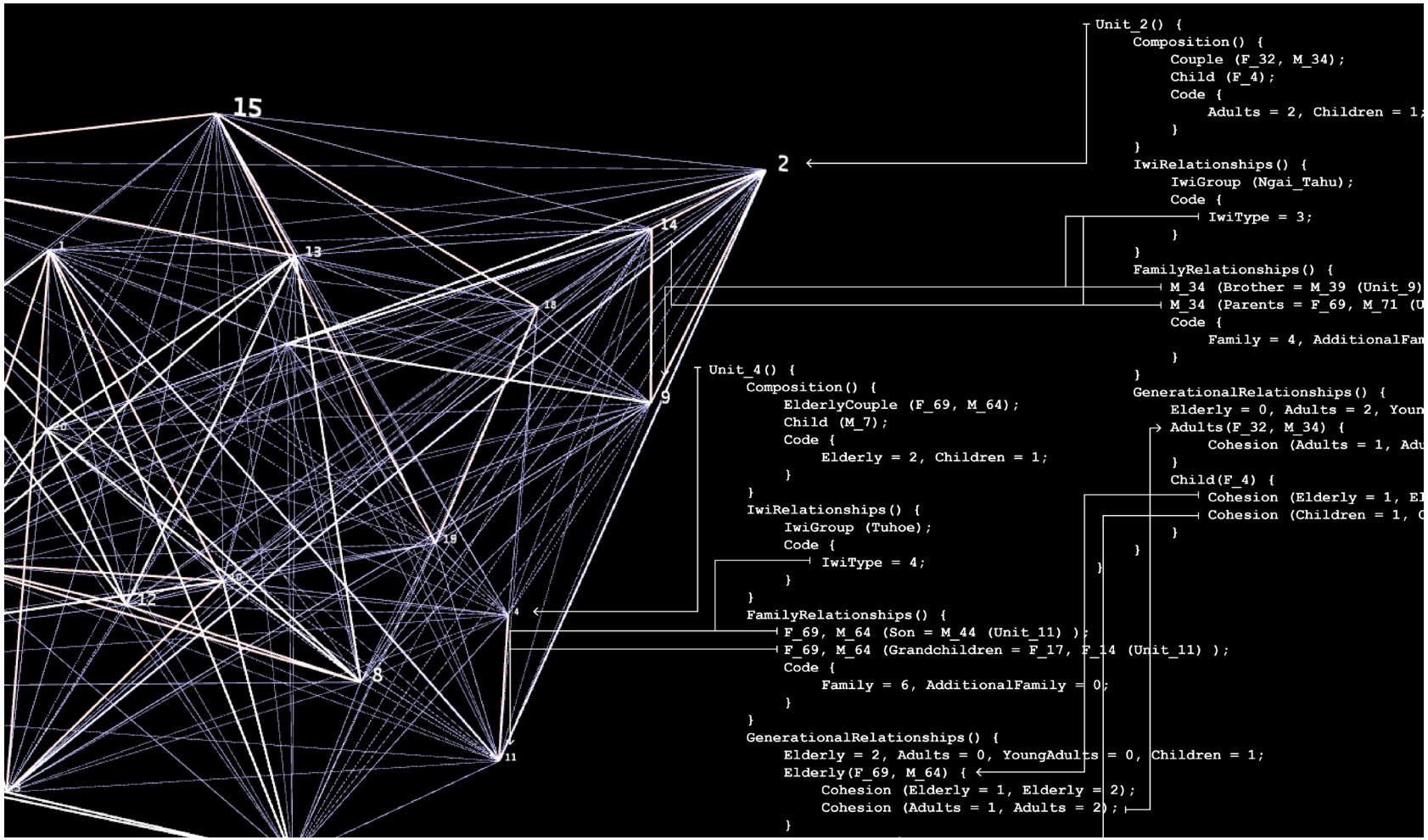
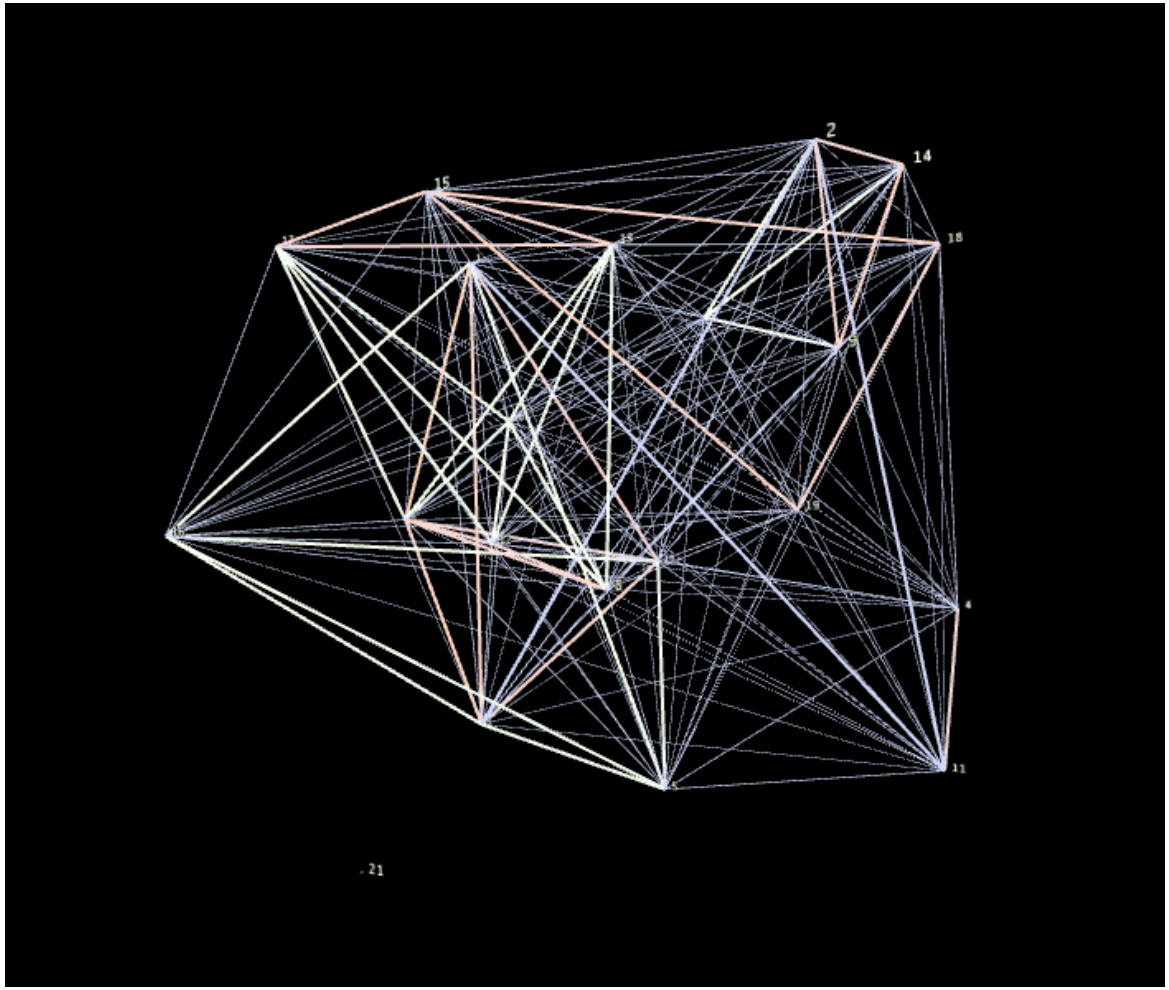

15
2
14
13
1
18
9
19
12
4
8
11
Unit_2() {
Composition() {
Couple (F_32, M_34);
Child (F_4);
Code {
Adults = 2, Children = 1;
}
}
IwiRelationships() {
IwiGroup (Ngai_Tahu);
Code {
IwiType = 3;
}
}
FamilyRelationships() {
M_34 (Brother = M_39 (Unit_9)
M_34 (Parents = F_69, M_71 (U
Code {
Family = 4, AdditionalFam
}
}
GenerationalRelationships() {
Elderly = 0, Adults = 2, Youn
Adults(F_32, M_34) {
Cohesion (Adults = 1, Adu
}
Child(F_4) {
Cohesion (Elderly = 1, El
Cohesion (Children = 1, C
}
}
}
Unit_4() {
Composition() {
ElderlyCouple (F_69, M_64);
Child (M_7);
Code {
Elderly = 2, Children = 1;
}
}
IwiRelationships() {
IwiGroup (Tuhoe);
Code {
IwiType = 4;
}
}
FamilyRelationships() {
F_69, M_64 (Son = M_44 (Unit_11));
F_69, M_64 (Grandchildren = F_17, F_14 (Unit_11));
Code {
Family = 6, AdditionalFamily = 0;
}
}
GenerationalRelationships() {
Elderly = 2, Adults = 0, YoungAdults = 0, Children = 1;
Elderly(F_69, M_64) {
Cohesion (Elderly = 1, Elderly = 2);
Cohesion (Adults = 1, Adults = 2);
}

Figure 90 (Opp.), 91 (R): Reading the Personalised Data

By activating the identification of unit codes we can establish which unit is which, allowing the designer to know precisely which variables each unit is to be designed for, and where each unit is to be placed within the system, reintroducing human value to the data. This brings the system to a full circle: from a group of people, to a string of connected data, and then an actively functioning group of people that respond and relate to each other as they desire, by virtue of their spatial arrangement.



CONCLUSION

Architectural design has made great advancements through the use of digital design tools, yet we have also failed to use them to understand the most fundamental of human behaviours. We have mastered the creation of technology that can investigate the relationship between a piece of steel and a sheet of glazing within a building, and all the intricate connections and materials between them, however we do not have systems that can investigate the intricate relationships that form between people. As an example, take a single 28 year old male with a 7 year old daughter, and his neighbours; an elderly couple, 78 and 81 years old respectively, who own a small dog and would like to see their grandchildren more often. Empathy would immediately tell us that between these two units there would be many relationship formations. We may assume the elderly couple are likely to be happy to have a child next door (as she may remind them of their grandchildren), and the single father would be glad to have neighbours that have experience with children and can offer support when needed. The young girl may enjoy having a small dog next door to play with, as well as having additional people to care for her. We personally understand how all of these relationships may emerge due to the fact that we understand human nature, however we do not have the means to interpret them on a large scale with our own minds. Therefore the ability to instruct a computer to understand these relationships, and provide spatial interpretations based on their organisation could prove paramount to understanding how people respond to each other through architecture.

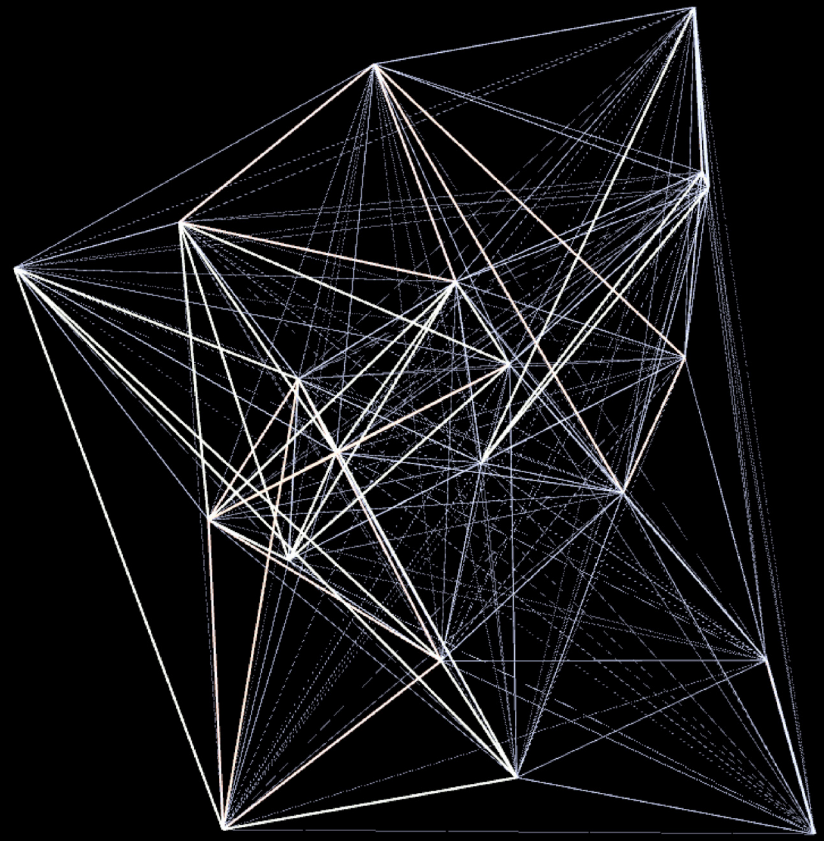
While this research began with the notion of investigating how Māori papakainga could be translated into urban environments, it became apparent that the issue was not with the environment to which it was being translated, rather the issue was with the components of the papakainga that were being used. Western understandings of architecture led to the idea that translating the physical form to the urban environment would create a successful transition, putting value in the material aspects of Māori papakainga. Consequently marae, communal areas, and other built forms that are prevalent within traditional Māori communities were constructed in urban settings in order to establish urban Māori papakainga. When that created isolation rather than integration the idea emerged that the built form wasn't the priority, rather community composition was the focus, and iwi groups built shared community blocks designed to western standards. Again, this didn't capture the essence of a traditional Maori papakainga. The reason behind this is that while the traditional built form is not vital, it is a highly successful physical manifestation of traditional Māori values, and these are what form the foundation of a Māori community. This shifted the focus of investigation, no longer was the priority how to integrate Māori into the urban environment, but rather, to investigate how traditional Māori community values manifest within architecture, and how these values can be simulated in any given scenario in order to create a system by which Māori values can be established in any environment, providing opportunities for settlement development.

This focus created a product, by which Māori values are simulated as relationships between each other and with the surrounding environment, and from which a spatial organisation system emerges. Throughout the research it has become clear that while this system was built with the Māori culture in mind, it can essentially be adapted to act as a spatial organisation system based on any relationship structure. The intention was always to relate this investigation to issues of loss of identity within non-western cultures that are under-represented within architecture, as a great deal of indigenous cultures suffer from the same misalignment of values that plague Māori within western design. Because of this, the system has been created to be highly adaptable to cater to any relationship structure, in order to create a greater understanding for all communities, real or imagined, within the architectural realm.

The modern world is currently heavily weighted in favour of western behaviours, and unless a way is found to integrate other cultures into this modern environment the divide will become even greater, perhaps insurmountable for some cultures, which would be an unfortunate loss in terms of global cultural enlightenment. This tool works to try and counter this divide, by providing a greater understanding of the values and relationships formed between cultures that are currently an unknown within the modern design world.

Figure 92: Final System Product

Technology is now advanced to the point where a group of people can be taken, their behaviours identified, and through coding can be organised in a way that their behaviours and desires work in harmony with the group. This creates a virtual spatial community, that can then be turned into built form, and a real community can emerge. This kind of architecture emerges from the fundamental behaviours of humans, and can be expressed in the form of a relationship network.



GLOSSARY:

Awa: *A river, stream or creek.*
Hapu: *A tribal or sub-tribal group.*
Iwi: *A tribal group, typically larger than a hapu.*
Mana:
Marae: *The area in front of a wharenui, however typically refers to the collection of buildings around the marae, the wharenui, wharekai and marae atea.*
Maunga: *A hill or mountain.*
Papakainga:
Papatuanuku: *The earth, mother of the gods and all of creation, wife to Ranginui, the sky-father.*
Tikanga: *Codes, plans, or protocol.*
Tipuna: *Ancestors or grandparents.*
Turangawaewae: *Place where a person has a right of residence through their tipuna and whakapapa.*
Waka: *Boat or ship. Usually refers to the boat on which a tipuna came to Aotearoa.*
Whakapapa: *Lineage, history and ancestry.*
Whanau: *Family, immediate or other.*
Whangai: *Adopted, not in the legal sense, rather an emotional connection.*
Wharekai: *Building on the marae for the preparation and consumption of food.*
Wharenui: *Building on the marae for multiple living functions, most notably for prayer and a sleeping house.*
Whenua: *The land.*

REFERENCES:

BlueThen (2011) *Curtain (Fabric Simulator)*. Web. OpenProcessing sketch 20140. Accessed Feb 2017.

Burkhardt, L. and Swallow, N. (2014) *Papakainga Development – Turning aspiration into reality*, Resource Management Journal, November 2014. Resource Management Law Association. Web. Accessed Sept 2016, 11-15.

Cropp, M. (2015) . Radio New Zealand. Web. Accessed Sept 2016.

Daniel, L., Soebarto, V., Zuo, J. (eds) (2016) *Fifty years later: Revisiting the role of architectural science in design and practice: 50th International Conference of the Architectural Science Association 2016.* The Architectural Science Association and The University of Adelaide, Adelaide, Australia, 79-88.`

Dupuis, A. and Lysnar, P. (2015) *Meeting the housing needs of multi-generational households*. Building Research Association of New Zealand. Web. Accessed Sept 2016.

Durie, E. (1987) '*The Law and the Land*'. Jock Phillips, ed. *Te Whenua Te Iwi, The Land and the People* . Allen & Unwin and Port Nicholson Press, Wellington, 78– 81.

Fabian, A. and Goodyear, R. (2014) *Housing in Auckland: Trends in housing from the Census of Population and Dwellings 1981 to 2013*. Statistics New Zealand, Wellington, 66-67.

Fitzgerald, E., King, P., Wadegrave, C. and Walker, T. (2006) *Māori Housing Experiences: Emerging Trends and Issues*. The Family Centre. Web. Accessed Sept 2016, 51-56, 120.

Getliff, J. M., Lowrance, G. F., Scott, R. S. (1999) *A Suitcase of Letters: the history of a family of Wilsons told by them in the letters they wrote from the 1790s - 1930s.* R. S. Scott, Richmond, New Zealand, Supplementary genealogical table.

Henderson (2011) *Attraction/Repulsion*. Web. OpenProcessing sketch 18798. Accessed Feb 2017.

Joynt, J. L. R., Lysnar, and P. Tuatagaloa, P. (2016)
and initiatives. Auckland Council Technical Report.

Kokkugia.com (2016) *Kokkugia – About*. Web. Kokkugia. Accessed June 2016.

Martin, D. R. (1997).). University of Otago, Dunedin, New Zealand.

Mead, H. (2016). . Huia (NZ) Ltd, Wellington, New Zealand.

Resnick, M. (1994) Massachusetts Institute of
Technology, United States of America, 68-74.

Reynolds, C.W. (1987) *Flocks, Herds, and Schools: A Distributed Behavioural Model*. Computer Graphics, ACM SIGGRAPH
'87 Conference Proceedings, Anaheim, California, 25-34.

Roche. F. (2016) *New Territories*. Web. New Territories. Accessed June 2016.

Schrader. B, (2013) . Te Ara - the Encyclopedia of New Zealand. Web. Accessed March 2017.
1-5.

Weiss, G. (1999) .
Massachusetts Institute of Technology, United States of America, 1-9.

LIST OF FIGURES:

Figure 1: Individual Complexities.
Figure 2: Past Interpretive Timeline of Western and Māori Architecture.
Figure 3: Projected Interpretive Timeline of Western and Māori Architecture.
Figure 4: Angas, George. *Potatau Te Wherowhero's pa*, 1847. Artwork. New Zealand History, Ministry for Culture and Heritage. *Potatau Te Wherowhero's Pa*. Web. Updated Jul 2014. Accessed Feb 2017.
Figure 5: Current Papakainga Composition Diagram.
Figure 6: Goldie, Charles, *All 'e same t'e Pakeha*. 1905. Dunedin Public Art Gallery, New Zealand Museums. *All 'e same t'e Pakeha (Te Aho-o-te-Rangi Wharepu, Ngãti Mahuta) ; Charles Goldie; 1905; 3-1936*. Web. Accessed Feb 2017.
Figure 7: Angas, George. *Potatau Te Wherowhero's pa*. See Figure 4. With Authors own Overlay of Iwi Structure.
Figure 8: Gully, John. *Cable Bay from Maori Pa*. 1882. Artwork. The Bishop Suter Art Gallery, New Zealand Museums. *Cable* Web. Accessed Feb 2017.
Figure 9: Angas, George. *Potatau Te Wherowhero's pa*. See Figure 4.
Figure 10: Authors own Family Tree. (Getliff, Lowrence and Scott, 1999).
Figure 11: Generic urban community. ***Cul De Sac Home Guide***. Web Accessed Feb 2017.
Figure 12: Generic property information drawing. *S & G Ramos Realty*. Web. Accessed Feb 2017.
Figure 13: Generic city scape image. . Web. Accessed Feb 2017.
Figure 14: Isolation Model Diagram.
Figure 15: Assimilation Model Diagram.
Figure 16: Communal Relationship Structuring.
Figure 17: Iwi Relationships.
Figure 18: Familial Relationships.

Figure 19: Individual Relationships.
Figure 20: Unified Relationship Structuring.
Figure 21: Flocking: Separation. Authors representation of Flocking Principles (Reynolds, 1987).
Figure 22: Flocking: Alignment. Authors representation of Flocking Principles (Reynolds, 1987).
Figure 23: Flocking: Cohesion. Authors representation of Flocking Principles (Reynolds, 1987).
Figure 24: *Batman Returns* Bat Simulation.
Figure 25: Traffic Navigation System (Resnick, 1994).
Figure 26: Work of Kokkugia (Kokkugia, 2016).
Figure 27: Swarm Flocking. See Appendix Item 3 (adapted from Henderson, 2011).
Figure 28: Curtain (Fabric Simulator). See Appendix Item 4 (BlueThen, 2011).
Figure 29: Rain without Gravity Code.
Figure 30: Rain Without Gravity Simulation.
Figure 31: Exploring Reynolds' Principles Through Code. Authors Work (Reynolds, 1987).
Figure 32: Working with Separation.
Figure 33: Working with Alignment.
Figure 34: Working with Cohesion.
Figure 35: Working with Separation, Alignment, and Cohesion in Unison.
Figure 36: Reynolds' Principles Unified Harmoniously.
Figure 37: Mapping Iwi Relationships.
Figure 38: Mapping Multiple Iwi Relationships.
Figure 39: Mapping Multiple Iwi Relationships (System). Appendix Item 7.
Figure 40: First Phase of Spatial Mapping.
Figure 41: Second Phase of Spatial Mapping.
Figure 42: Identifying Relationships in Second Phase of Mapping.
Figure 43: Current Output Simulation.

Figure 44: Moving into a Three Dimensional System.
Figure 45: Refining the Basic Principles for a Three Dimensional World.
Figure 46: Working with Area.
Figure 47: Refining Iwi Group Cohesion.
Figure 48: Refining Family Type Cohesion and Separation.
Figure 49: Adding External Factors.
Figure 50: Example of Part II Coding.
Figure 51: Example of Part II Coding.
Figure 52: Final Simulation Model.
Figure 53: Traditional Spatial Interpretation of Final Sketch.
Figure 54: Refining Code to Date.
Figure 55: Relocating the Source Code.
Figure 56: Relocating the Source Outcome.
Figure 57: Finalising Iwi Type Simulation Code.
Figure 58: Finalising Iwi Type Simulation Outcome.
Figure 59: Generational Attraction Code.
Figure 60: Generational Attraction Outcome.
Figure 61: Dependant People Code.
Figure 62: Dependant People Outcome.
Figure 63: Familial Relationships Code.
Figure 64: Familial Relationships Outcome.
Figure 65: Addition of Communal Spaces Code.
Figure 66: Addition of Communal Spaces Outcome.
Figure 67: Final System Code.
Figure 68: Final System Outcome.

Figure 69: Code of the Final Product.
Figure 70: Input Variables 1.
Figure 71: Input Variables 2.
Figure 72: Input Variables 3.
Figure 73: Behavioural Coding 1.
Figure 74: Behavioural Coding 2.
Figure 75: Behavioural Coding 3.
Figure 76: Output Directives 1.
Figure 77: Output Directives 2.
Figure 78: Output Directives 3.
Figure 79: Data Files
Figure 80: Final Product of System.
Figure 81: Final Data Use.
Figure 82: Maya Spring Dynamics.
Figure 83: Relationship Identification.
Figure 84: Final Relationship Network as a Whole.
Figure 85: Final Relationship Network in Parts.
Figure 86: Direct Output.
Figure 87: Interpreted in a Traditional Format.
Figure 88: Network Model.
Figure 89: Network Model as a Design Tool.
Figure 90: Reading the Personalised Data in Detail.
Figure 91: Reading the Personalised Data as a Whole.
Figure 92: Final System Product.

APPENDIX

APPENDIX ITEMS:

1. Papakainga Composition Research
2. Urban Papakainga Exemplars
3. Flocking
    Attached File: 'Item_3_Flocking.mp4'
4. Curtain (Fabric Simulator)
    Attached File: 'Item_4_Curtain(Fabric_Simulator).mp4'
5. Working with Separation, Alignment and Cohesion
    Attached File: 'Item_5_Sep_Ali_Coh.mp4'
6. Reynolds Principles Unified
    Attached File: 'Item_6_Unified_Principles.mp4'
7. Mapping System
    Attached File: 'Item_7_Mapping.mp4
    Attached Folder: 'Item_7_Mapping'
8. Family and Housing Typology Research

9. 2D Simulations
    Attached File: 'Item_9_2D_Simulations.mp4'
10. 3D Simulations
    Attached File: 'Item_10_3D_Simulations.mp4'
11. Unit Specifications
12. Final System
    Attached File: 'Item_12_ Final_System.mp4'
    Attached Folder: 'Item_12_Final_System'
13. Spring Systems
    Attached File: 'Item_13_Springs.mp4'
14. Established Relationship Network
    Attached File: 'Item_14_Established_Relationship_Network.mp4'
    Attached Folder: 'Item_14_Established_Relationship_Network'
15. Generating Relationship Network
    Attached File: 'Item_15_Generating_Relationship_Network.mp4'
    Attached Folder: 'Item_15_Generating_Relationship_Network'

This is an investigation into the design of a tribally based, mixed use, residential apartment block using digital design tools to inform design methods.
The research revolves around the creation of a system of design, looking into how multi-agent systems can be employed to navigate complex social and cultural scenarios within pan-tribal communities, and from that inform spatial configurations and create building envelops that are based on specific cultural behaviours, therefore bringing an interpretation of traditional rural living values into the urban setting.

What is an Urban Papakainga?

A traditional papakainga is essentially a settlement of a particular iwi or hapu (tribal groups and subgroups).
Compare to:

- Rural Papakainga
- Past Precedents of attempts at the Urbanisation of Papakainga (Opp.)

Rural Papakainga:

Rural papakainga are generally composed of housing, with easy access to areas of cultural amenity - Marae, Urupa, and often with a Kura/Kohanga Reo. They usually have the ability to function independently of the outside world (Mead, 2016).

- Iwi orientated

- Multi generational living

- Great diversity in family types, huge fluctuations, hard to design for a specific group, considering the dynamics are ever changing

- Marae centric

- Open environment

- Connection to the land through Maunga/Awa etc. As well as a form of self sufficiency as a source of food

- Community environment

Urban Setting:

- Largely disconnected tribally. The majority of Māori in the area will be from many different iwi, with no cultural base

- Some Manawhenua/Ahi Ka present (people of the land, or residing iwi)

- Tends to be split, older generations in the fringe suburbs, younger generations grouped together, generally very stable

- No cultural unifiers - Marae/Kohanga etc.

- Confined, limited space

- Land connection partially lost, in some cases there are still the Maunga/Awa connection, however sourcing food off the land is rare, and landmarks have mostly been built up

How to create a papakainga with the following constraints of an urban setting:

- Both manawhenua and tribally displaced Māori are living within the same area

- Bringing traditional papakainga family typologies into an area of limited living styles

- No pre established cultural unifiers, they need to be included within the papakainga

- Spatial confines do not allow for traditional layout of a papakainga, this must be translated into a more urban appropriate design.

- No land established connection with the land. Land is seen as a source of profit rather than a tool and a treasure

- Urban areas typically hold 'lost' areas of Iwi significance, and while these are gone to most, they are very important to a few, and should therefore be respected on the grounds that in the Māori culture places of importance are cherished for the history they hold

Greta Point

The Te Aro Pā papakainga development was completed in March 2016 and operates as Wellington's first urban Papakainga. Interest lies in how it compares to a rural papakainga.

- Iwi specific, the papakainga is owned by the Te Aro Pā trust and is built with the intent of housing descendants of this tribe
- Intended for multi-generational living, however all apartments are two storied - may cause some issues
- No Marae
- Closed environment, no connection between houses
- Attempt at connection to land and community spirit with planters, not on the rural scale however

Images redacted for copyright reasons

Waiwhetu

The Waiwhetu papakainga development was the first Wellington development to result from the Port Nicholson Trust Bill, returning Iwi land for residential development, this granted 24 homes for the Iwi.

- Currently Iwi specific, however have been granted the right to expand their papakainga, and upon doing so, they plan to diversify if capacity allows for it
- Multi-generational living
- Marae for community use and by request, as well as Iwi runanga, a medical centre, radio station etc.
- More open than most urban settings, however more of an attempt to insert rural into urban settings, rather than adapting to the urban lifestyle

Images redacted for copyright reasons

Pūkaki Papakainga

Pūkaki Trust Papakainga is located in Mangere, Auckland. It is still under development, aiming to expand from 12 to 22 dwellings

- Iwi specific, multiple land owners so dwellings will most likely stay in the families that originally purchased them
- Multi generational living for many family types, 4 housing typologies
- Located next to their Marae and in close proximity to the Urupa
- Open environment, on the outskirts of the urban centre, more like a rural papakainga in actuality
- Development intent was for the Iwi to reoccupy their land, the relationship established is blossoming

Images redacted for copyright reasons

Appendix Item 3. Flocking

Attached File: 'Item_3_Flocking.mp4'

Appendix Item 4. Curtain (Fabric Simulator)

Attached File: 'Item_4_Curtain(Fabric_Simulator).mp4'

Appendix Item 5. Working with Separation, Alignment and Cohesion

Attached File: 'Item_5_Sep_Ali_Coh.mp4'

Appendix Item 6. Reynolds Principles Unified

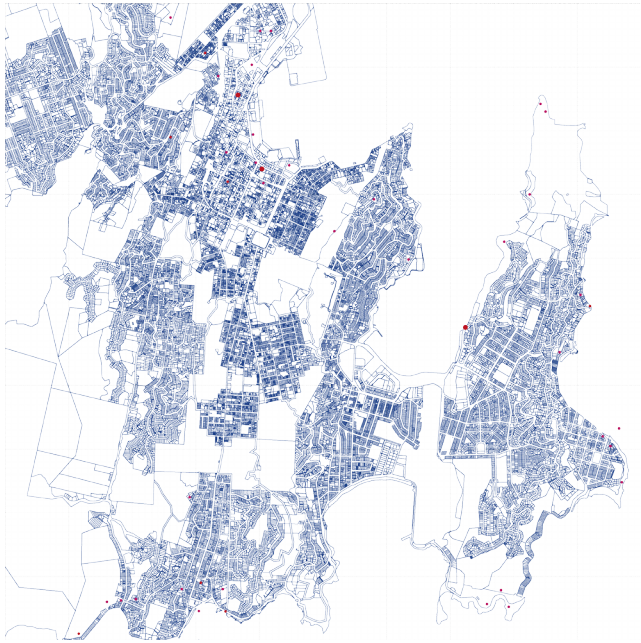
Attached File: 'Item_6_Unified_Principles.mp4'

Appendix Item 7. Mapping System (Opp.)

Attached File: 'Item_7_Mapping.mp4

Attached Folder: 'Item_7_Mapping'

Appendix Item 7. Mapping System

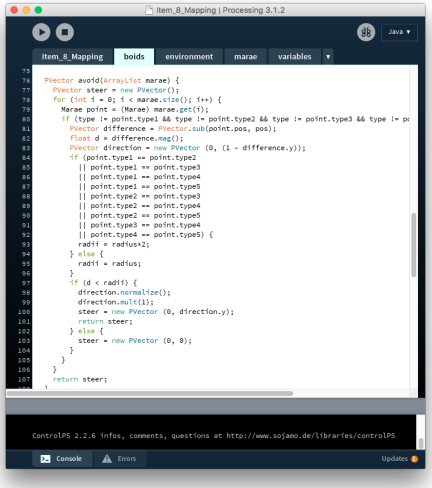


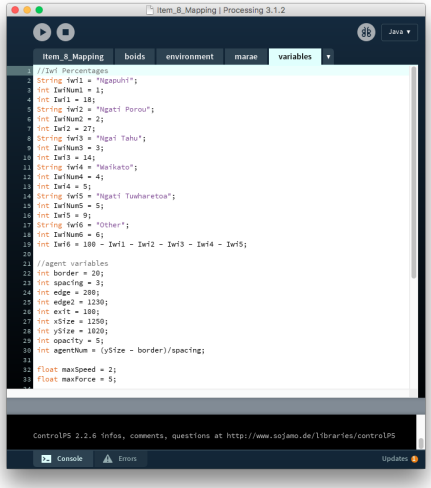
	A	B	C	D	E	F	G
1	262	22	1	2	3	10	10
2	401	43	1	4	10	10	10
3	417	43	3	10	10	10	10
4	315	80	6	10	10	10	10
5	388	97	3	10	10	10	10
6	535	114	2	10	10	10	10
7	366	143	2	2	2	1	3
8	260	210	5	1	10	10	10
9	390	206	2	3	10	10	10
10	348	234	1	10	10	10	10
11	350	280	6	10	10	10	10
12	394	250	6	10	10	10	10
13	404	260	3	3	2	4	10
14	406	282	5	10	10	10	10
15	447	254	2	1	10	10	10
16	518	358	6	10	10	10	10
17	580	308	1	10	10	10	10
18	634	402	6	10	10	10	10
19	290	776	1	10	10	10	10
20	723	508	3	3	10	10	10
21	116	990	6	2	3	10	10
22	160	950	2	10	10	10	10
23	182	938	1	10	10	10	10
24	206	936	6	10	10	10	10
25	278	920	1	2	10	10	10
26	308	910	5	10	10	10	10
27	305	955	4	2	4	5	10
28	342	920	2	10	10	10	10
29	347	955	6	10	10	10	10
30	632	402	3	10	10	10	10
31	842	158	5	1	5	10	10
32	850	170	5	10	10	10	10
33	825	300	2	3	10	10	10
34	785	374	6	10	10	10	10
35	905	456	6	2	1	10	10
36	872	548	3	10	10	10	10
37	966	668	4	10	10	10	10

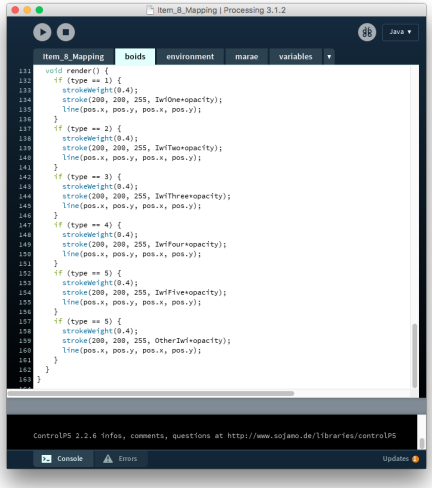
Item_8_Mapping | boids | environment | marae | variables

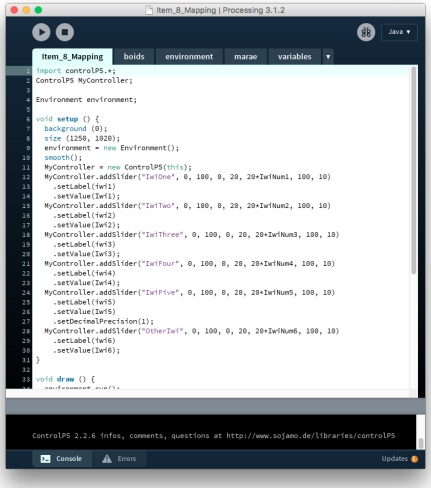
```
class Environment {
  ArrayList agents;
  ArrayList marae;

  Environment() {
    initAgents();
    initMarae();
  }

  void initAgents() {
    agents = new ArrayList();
    for (int i = (border/spacing); i < agentNum; i++) {
      for (int j = 1; j < 7; j++) {
        Boid agent = new Boid(0, (i*spacing + 0.1), j);
        agents.add(agent);
      }
    }
  }


  void initMarae() {
    float Ix;
    float Iy;
    int Mt1, Mt2, Mt3, Mt4, Mt5;

    marae = new ArrayList();
    String lines[] = loadStrings("Marae.csv");
    String [][] csv;
    int csvWidth=0;
    for (int i=0; i < lines.length; i++) {
      String [] chars=split(lines[i], ',');
      if (chars.length>csvWidth) {
        csvWidth=chars.length;
```

The input for this system is based on locating the areas that hold Māori significance on a 1000x1000 point map, then translating that information into a csv file. X and Y locations on the map are placed in the columns A and B, and columns C - G indicate which iwi are affiliated with each location based on iwi codes of 1 - 6. This file is then called upon in the sketch code, the name and location of which must correspond with the call-out, in this case the file "Marae.csv" located in the data folder of the sketch. Adjusting the system to other locations simply requires the adjustment of this csv file.



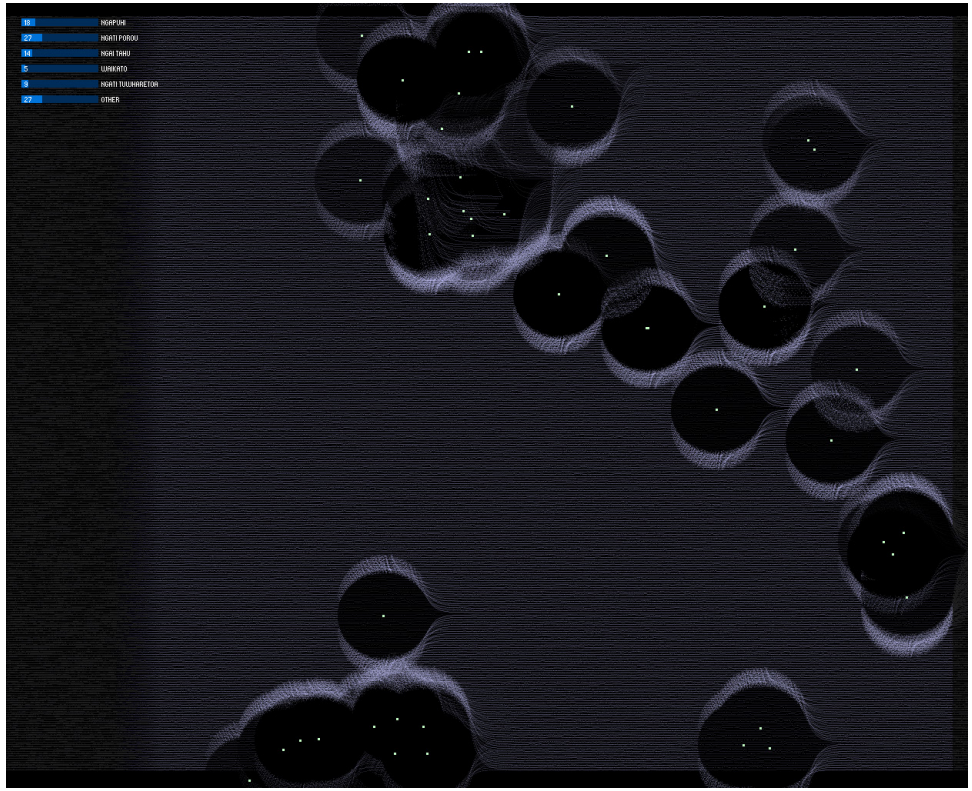


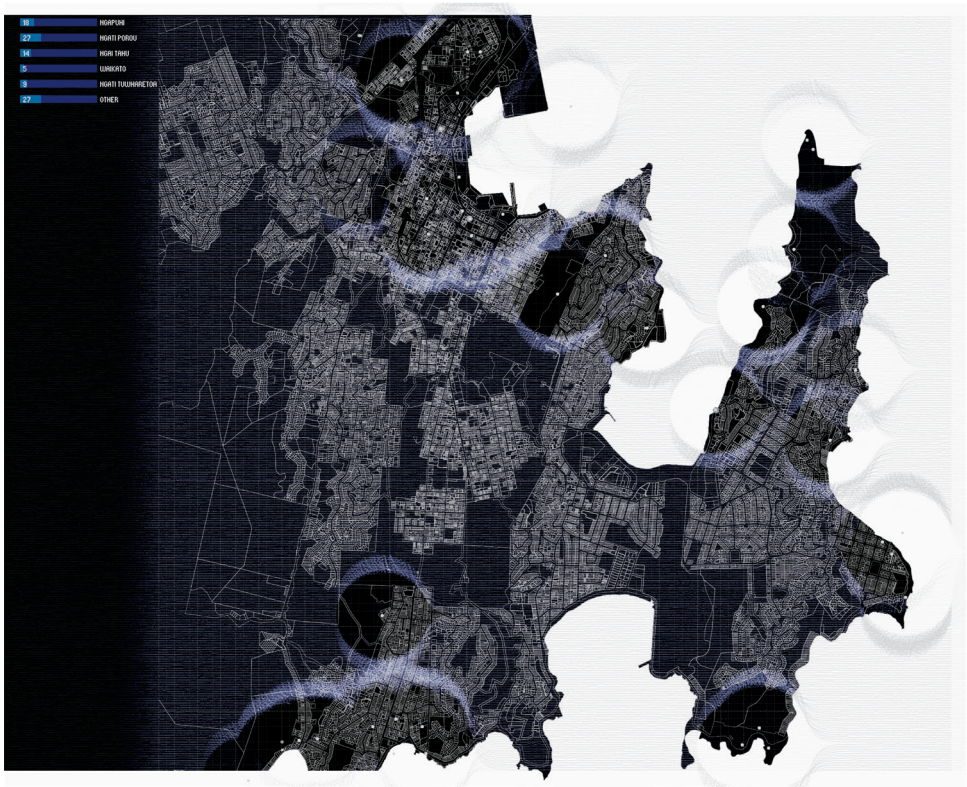




In order to work with different iwi, more iwi or less iwi, the sketch code needs to be updated throughout. This type of manipulation is less easily carried out than the previous, because it is less likely to require adaptation. In this scenario 5 major iwi groups are considered as having iwi affiliations with the area, with a 6th group encompassing all iwi that have no affiliation with the land. The percentage associated with each iwi in the second image is a reflection of the ratio of that particular iwi in the group looking for potential settlement locations.

Please note, sketch will not run without the control.p5 library loaded. This can be accessed from Sketch > Import Library > Add Library, then searching and downloading control.p5.





By using the resulting mapping simulation as an overlay for the initial map, a graphic is provided where potential areas for relocation have been identified, and areas that should be avoided are left undisturbed. This can be used in any scenario provided the data for areas of iwi significance can be gained.

IWI

One of the most significant Cultural Parameters with which to base this research on is the significance with which the Māori community treat their Iwi affiliations. An Iwi is a tribe, and within that Iwi are many Hapū (sub-tribes) and the relationships formed within these are very important to the Māori culture.

Iwi Living

The Māori culture has very significant Iwi relationships that are fundamentally rooted in their way of life. As well as the western understanding of a familial relationship, Māori have an Iwi relationship which is as ingrained in every day life as their immediate family connections are.
Traditionally, individual Iwi relationships are how residential settlements are formed, with people of the same Iwi/Hapū residing in the same area.
This research however is looking into how mixed Iwi can co-reside, rather than using one particular Iwi as the unifying living factor, this is exploring how the desire to maintain strong Iwi relationships can bring a particular settlement together, regardless of which Iwi one may have connections with. The necessity for cohabitation comes out of the lack of demand within any particular Iwi for them to stage this type of intervention alone, and may lead to a more successful model, as the desire to merge for the retention of Iwi living overcomes the traditional separation due to historical conflicts and gripes.

Traditional Papakainga

A traditional Papakainga is a rural settlement, low-rise and wide spread.
The settlement is generally located in an area of historical value to the Iwi, with many locations of significance within close proximity.
The composition of the settlement generally is that housing surrounds general community buildings. These building may include:

- Marae
- A hall or church
- Education facilities like a Kohanga (Māori pre-school) or a primary school
- Or perhaps other buildings that serve a community service

Generally these settlements are relatively self sustaining, with education up to a certain level (generally not past primary school, with an exception of composite schools), which would include the teaching of more traditionally Māori values and language.

Land Value

Of particular importance to Iwi in terms of settlement locations, are areas of historical significance. By this understanding, traditional settlements are formed near areas that are significant to the Iwi being settled. When this translates to creating a settlement in a foreign area it adds a complication in the form of Iwi that do have a rightful territorial claim on the wider region. Due to this complication, an unknown area should not be settled in without firstly locating areas of significance to all iwi within the area, and building out of range of any of them.

Interactions

The most architecturally important social value within a traditional settlement is that of the social interactions that form in and around the community in terms of generational divide, or in the case of Māori communities, the lack of divide. Multi-generational living is not only common, it is expected, in that the community operates on the basis of supporting each other. This means that family compositions are always in a semi-permanent state, with the possibility of fluctuations and alterations that can be completely unpredictable, however due to this understanding, Māori living has a more flexible understanding of architectural implications than the more rigid western understanding.

FAMILY TYPES

Identifying 'typical' Family Types, and analysing how cultural qualities can affect these typologies in term of adding outlying factors.

Single Person	Any of the house type options
Single Parent, Primary Caregiver	Any house type with more than one bedroom
Single Parent, Secondary Caregiver	Any of the house type options
Young Couple	One of the smaller house types
Couple without Children	Any of the house type options, generally a two bedroom or larger
Couple with Children	A two bedroom house minimum, more commonly a three bedroom
Elderly Couple	Any of the house type options
Elderly Single Person	Any of the house type options
Flatting Household	Three bedroom household minimum

Children

Within Māori communities it is not unusual to whangai children, in that they are adopted (usually unofficially) into a family with which there is a previous relationship. This is done for a variety of reasons, either as a result of issues with the current living situation of the child, to provide stability, for education opportunities, or even for the benefit of the family who are to take the child in.

Requirements: Possible addition of bedroom.

Elderly

The idea of shared caring responsibilities extends from children to the elderly. Within cultures where multi-generational living is common, care of the elderly is very important. When living alone becomes difficult or even less desirable than living with people, it is almost a given that a family member will open their home to the permanent addition of an elderly person, for the benefit of all involved.

Requirements: Accessible bedroom.

Dependant Person/s

Due to the shared caring responsibility, it is very uncommon for a person who needs help to look after themselves to be left without help from their family and community. Whether the dependency is financial, physical or mental, it will have an effect on the composition of the family and house type, as well as an investment of income and time.

Requirements: Bedroom. Potentially accessible.

Short-Term Additions

Short term additions are most common within elderly households, with a child of the elderly person or couple living with them for a while to provide help during times of pressure. Also very relevant for seasonal workers, who may have two or more places where they live short term, in order to seek working opportunities in a particular field. This is least common type of addition that would emerge in this type of development.

Requirements: Probable addition bedroom.

Long-Term Additions

Long-term additions address the situation where a single adult may choose to live with family rather than alone, this could be for financial reasons or personal choice, both of which are common with situations where a family with children has an aunty or uncle live with them, relieving some child care pressures for the parents, and providing a welcoming home environment to the unattached adult.

Requirements: Bedroom, larger floor area is ideal.

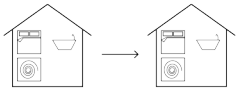
UNIT TYPES

Using the Family Types that emerged from the consideration of Cultural Parameters, we can edit standard residential apartment typologies to be more fitting Unit Types for the collective at hand. Most apartments just have an increased floor area in common rooms and secondary bedrooms, others have more significant differences, and others have no change or aren't required at all in this set up. Brief descriptions of the changes are indicated to the right, and the data is part of the 2D and 3D spatial programming, which is expanded upon in the later section on coding.

Studio Apartment

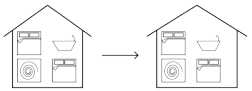
Not required in this model as cultural tendencies indicate that occupants likely to use a studio are more likely to live with family

Single Bedroom Apartment



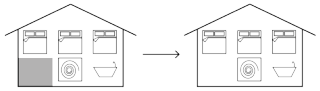
Single bedroom only required for those who don't need extra space, so unit stays the same

Two Bedroom Apartment



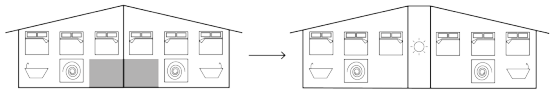
This apartment has an increased second bedroom size and an increased living room size for housing of fluctuating residents

Three Bedroom Apartment



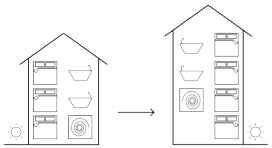
Apartment has large bedrooms in case of cohabitation, with a larger living room for temporary living demands

Duplex



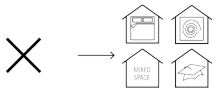
Larger floor areas for potentially larger family composition, added communal outdoor space

Penthouse Apartment



Penthouse suite targets luxury, which in this case is a spatious dwelling. Extra bedroom, larger areas

Extra Communal Spaces



Extra communal spaces are required to fit in with the iwi model. Wharenui, wharekai, kohanga/hall

One Bedroom

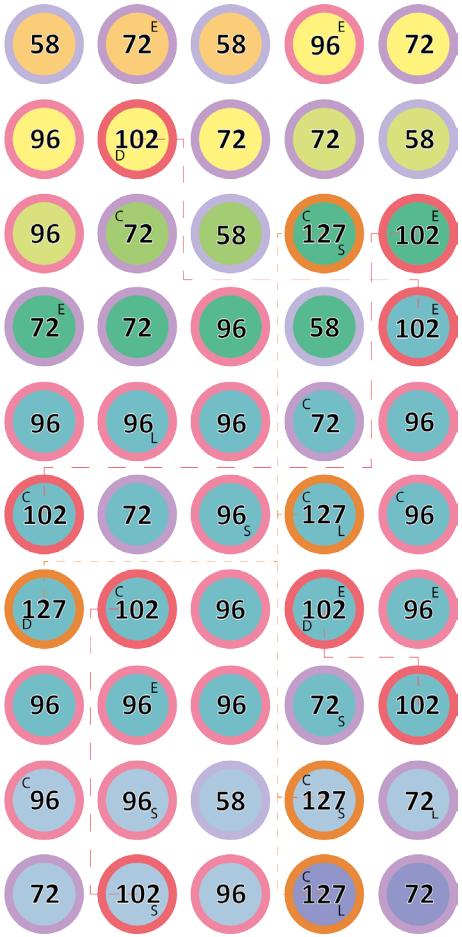
Two Bedroom

Three Bedroom

Four Bedroom

Five Bedroom

- Single Person
- Single Parent Main Caregiver
- Single Parent Secondary Caregiver
- Young Couple
- Couple without Children
- Couple with Children
- Elderly Couple
- Elderly Single
- Flatting Household
- Single Bedroom Apartment 58
- Two Bedroom Apartment 72
- Three Bedroom Apartment 96
- Duplex 102
- Penthouse 127
- Children
- Elderly
- Dependant Person/s
- Short Term
- Long Term



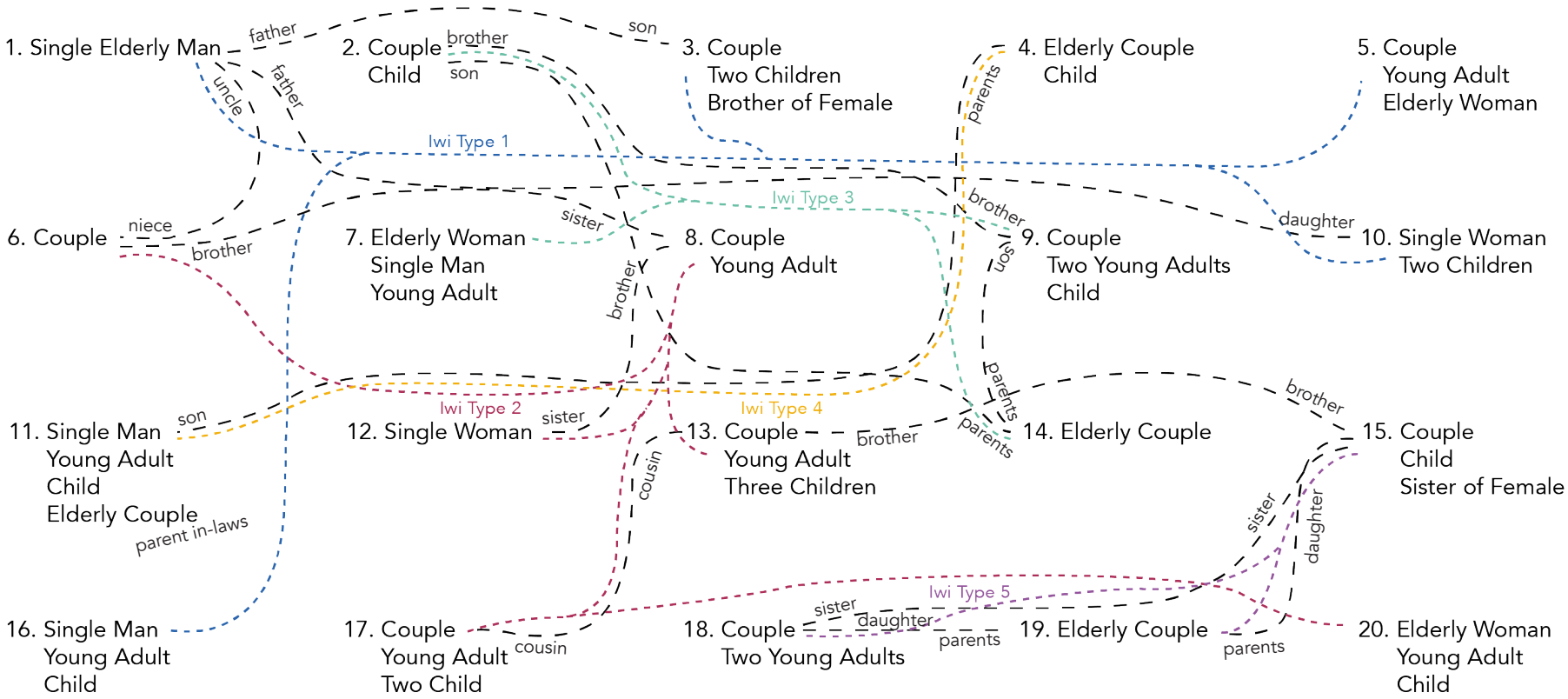
Appendix Item 9. 2D Simulations

Attached File: 'Item_9_2D_Simulations.mp4'

Appendix Item 10. 3D Simulations

Attached File: 'Item_10_3D_Simulations.mp4'

Appendix Item 11. Unit Specifications



Appendix Item 11. Unit Specifications

	A	B	C	D	E	F	G	H	I	J
1	Agent Number	Area	Iwi Number	Family	Additional Family	Children	Young Adults	Adults	Elderly	Dependant Persons
2	1	50	1	1	0	0	0	0	1	0
3	2	75	3	4	0	1	0	2	0	0
4	3	100	1	1	0	2	0	2	0	0
5	4	75	4	6	0	1	0	0	2	0
6	5	100	1	5	0	2	1	2	1	1
7	6	75	2	1	2	0	0	2	0	0
8	7	100	3	3	0	0	1	1	1	0
9	8	50	2	2	0	0	1	2	0	0
10	9	100	3	4	0	1	2	2	0	0
11	10	75	1	1	0	2	0	1	0	0
12	11	100	4	6	0	1	1	1	2	1
13	12	75	2	2	0	0	0	1	0	0
14	13	100	2	7	0	2	1	2	0	0
15	14	50	3	4	0	0	0	0	2	0
16	15	100	5	8	7	1	0	2	0	1
17	16	75	1	9	0	1	1	1	0	0
18	17	100	2	7	0	2	1	2	0	0
19	18	75	5	8	0	0	2	2	0	0
20	19	100	5	8	0	0	0	0	2	0
21	20	75	2	10	0	1	1	0	1	0
22	21	150	99	99	99	99	99	99	99	99

Unit specification data begins as a set of families with stories and connections and through numerical interpretation becomes this set of data that is used within the final system simulations. From unit 16, a widowed father with his two children, 16 and 11, looking for support by settling with fellow iwi members, to data row 17, an agent ready to find a spatial location to establish a home.

Appendix Item 12. Final System

Attached File: 'Item_12_Final_System.mp4'

Attached Folder: 'Item_12_Final_System'

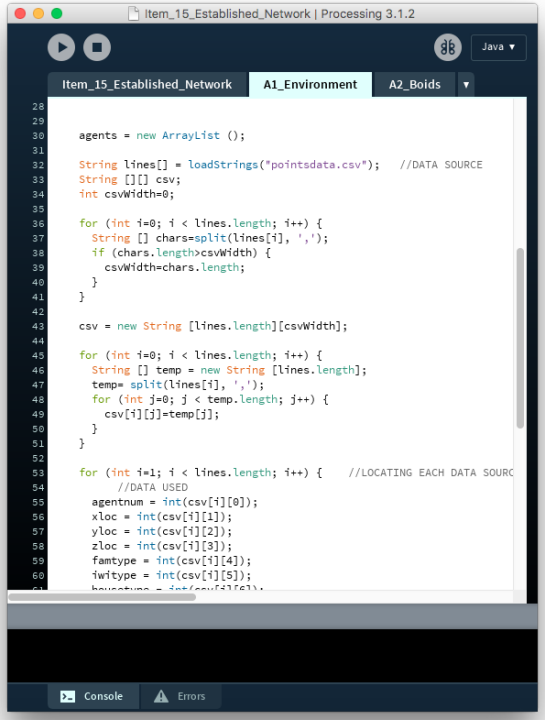
Appendix Item 13. Spring Systems

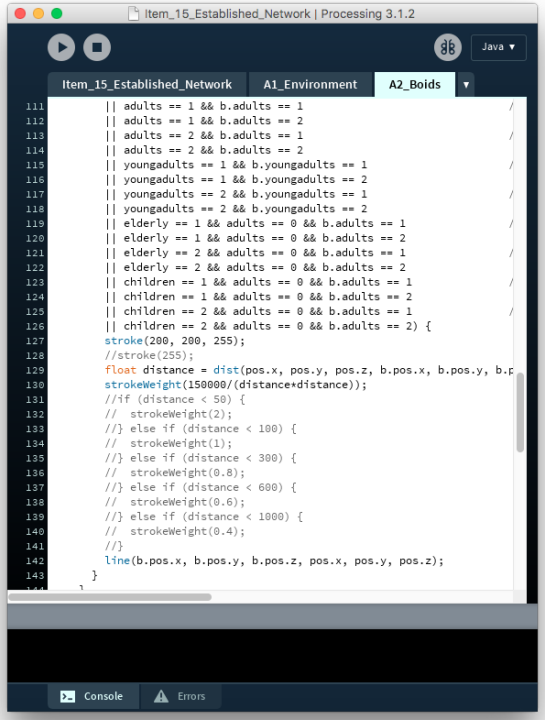
Attached File: 'Item_13_Springs.mp4'

Appendix Item 14. Established Relationship Network

Attached File: 'Item_14_Established_Relationship_Network.mp4'

Attached Folder: 'Item_14_Established_Relationship_Network'

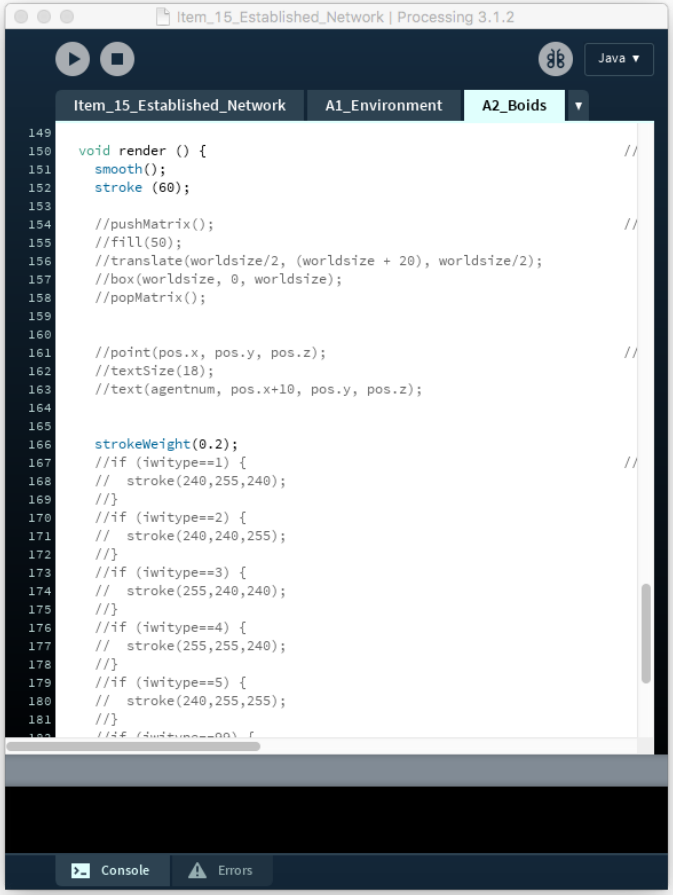




This system is based upon a csv file in the same format as the output of the Final System of Appendix Item 12, so the output file may be copied and pasted in the data folder of this sketch, the name changed to "pointsdata.csv" and the sketch will be ready to replicate the Final System simulation, and identify the relationships within the system.

The relationships within this sketch are grouped into the three original relationship categories, that of iwi relationships, familial relationships, and individual relationships, and colour coded for clarity. The individual relationships vary in simulated strength according to proximity, identifying which relationships were most effectual in the location manipulation of each agent.

Appendix Item 14. Established Relationship Network




```
149
150  void render () {                                                    //
151    smooth();
152    stroke (60);
153
154    //pushMatrix();                                                   //
155    //fill(50);
156    //translate(worldsize/2, (worldsize + 20), worldsize/2);
157    //box(worldsize, 0, worldsize);
158    //popMatrix();
159
160
161    //point(pos.x, pos.y, pos.z);                                     //
162    //textSize(18);
163    //text(agentnum, pos.x+10, pos.y, pos.z);
164
165
166    strokeWeight(0.2);
167    //if (iwitype==1) {                                               //
168    //  stroke(240,255,240);
169    //}
170    //if (iwitype==2) {
171    //  stroke(240,240,255);
172    //}
173    //if (iwitype==3) {
174    //  stroke(255,240,240);
175    //}
176    //if (iwitype==4) {
177    //  stroke(255,255,240);
178    //}
179    //if (iwitype==5) {
180    //  stroke(240,255,255);
181    //}
```

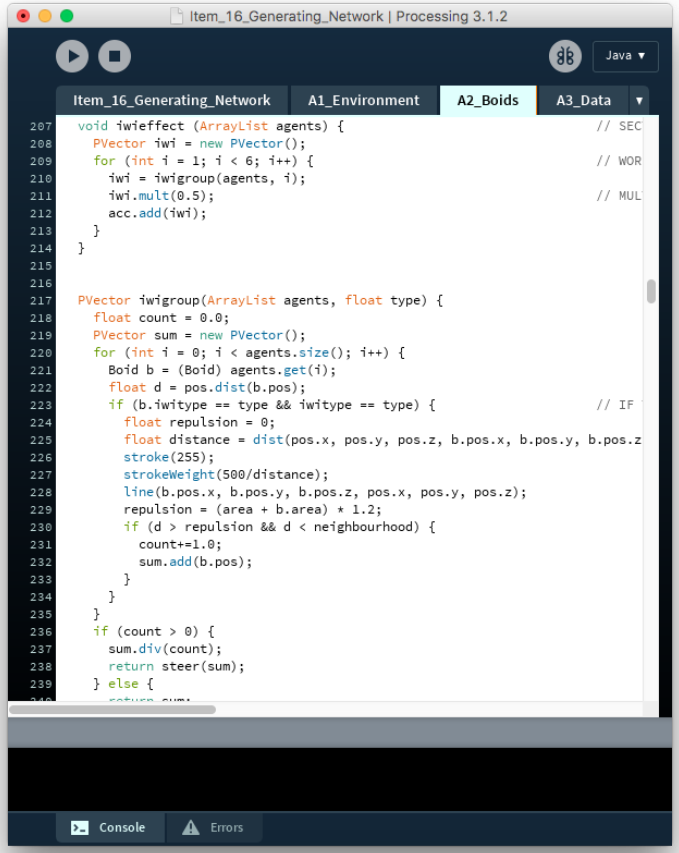
```
166    strokeWeight(0.2);
167    //if (iwitype==1) {                                               //
168    //  stroke(240,255,240);
169    //}
170    //if (iwitype==2) {
171    //  stroke(240,240,255);
172    //}
173    //if (iwitype==3) {
174    //  stroke(255,240,240);
175    //}
176    //if (iwitype==4) {
177    //  stroke(255,255,240);
178    //}
179    //if (iwitype==5) {
180    //  stroke(240,255,255);
181    //}
182    //if (iwitype==99) {
183    //  stroke(255,240,255);
184    //}
185
186
187    noStroke();                                                       //
188    //stroke(255);
189    //fill(255);
190    noFill();
191
192
193    pushMatrix();
194    translate(pos.x, pos.y, pos.z);
195    sphere(area);
196    popMatrix();
197  }
198 }
```

The visual output has several simulation options programmed for various methods of interpreting relationships. From spatial simulation and relationship networks, to one or the other, or just locations, this system provided the ability to simulate stable data as needed, editing and re-simulating a constant scenario to provide greater levels of understanding.

Appendix Item 15. Generating Relationship Network

Attached File: 'Item_15_Generating_Relationship_Network.mp4'

Attached Folder: 'Item_15_Generating_Relationship_Network'



This system acts as a combination of the two previous systems. The functionality of the Final System (Appendix Item 12) working in conjunction with the visualisation of the second. The scripting operates in the same way as the previous two operations.