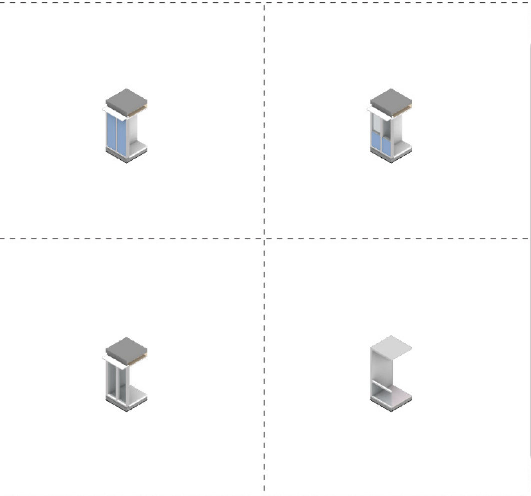
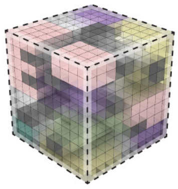
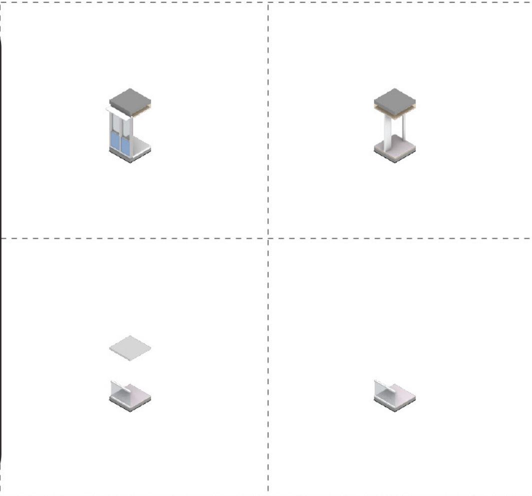
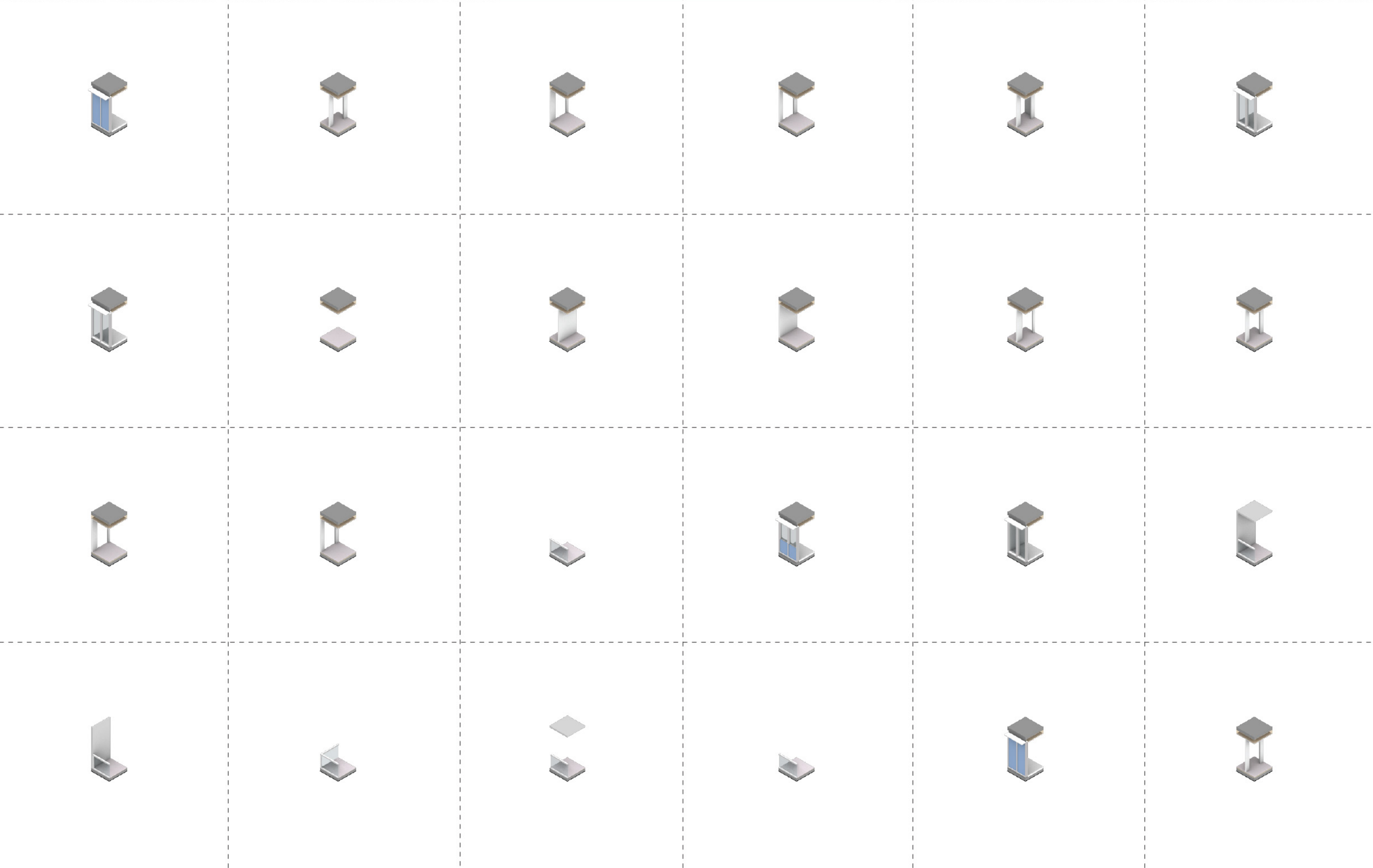


GEN-ARCH



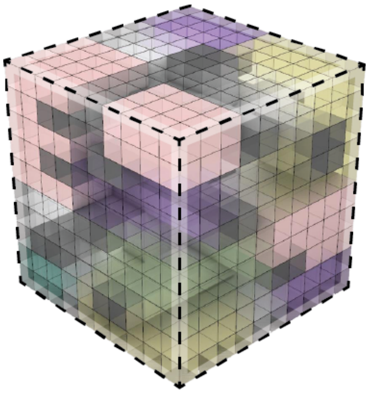
PLATFORM FOR DEVELOPING ARCHITECTURAL CONFIGURATIONS USING GENERATIVE DESIGN METHODOLOGIES.





GEN-ARCH

Platform for developing Architectural configurations using generative design methodologies.



Aditya Soman - 4999487

MSc Building Technology
Building Technology Graduation studio

Mentors:
Dr.ir. P. Nourian
Prof.ir. Thijs (M.F.) Asselbergs
ir. S. Azadi
June 28, 2021

Abstract

The Constructing industry around the world is growing exponentially. In the Netherlands itself there is a need of constructing 1 million affordable new homes by 2030. Such a massive demand for new buildings has given rise to a need for a solution for mass customization of architectural configurations. Simple repetition or a badly configured building is not ideal and can lead to a lot of social and environmental problems. Generative Design can offer a digital solution for this mass customization problem. It can utilize the power of Artificial intelligence in design to generate a system where users can create custom solutions rapidly. The second part of the customization problem is the user participation in determining the customization goals. More often than not the end users of the built space do not get an opinion in the configuration process and this leads to unsatisfactory and undemocratic design solutions producing urban inequality.

In this thesis a comprehensive digital solution by means of a digital platform is provided in an attempt to solve the 3D layout problem in a participatory manner. The platform built in the thesis is intended for designing Open buildings at a scale of support structures. The concept of Open buildings calls for separation of the structure and the infill layer making the buildings open for modification and customization.

The 3D-Layout configuration problem in its full sophistication is a classical example of a wicked problem. So in this thesis the problem is methodically broken down into a group of nested smaller problems with local goals to achieve the global spatial planing goals and various computational methods are tested for solving the problems. The nature of the smaller problems is often iterative hence a robust decision support system is also designed for stakeholder participation and control over the problem solving method.The computational methods used in the thesis are inspired from various other disciplines like Computer science, Industrial engineering, Operations research, etc. where they are used to solve similar problems in their respective domains. Techniques like Multi-agent systems, Multi-criteria decision analysis, Techniques for solving the clustering and packing problems in operations research are explored to solve the various nested smaller problems in the 3D-Layout problem.

Finally all the methods developed in the project are used to solve a test case for the design of a mixed used Open building consisting of Housing and Commercial spaces in the Buiksloterham region which is one of the first circular neighbourhoods in Amsterdam to showcase the potential of the platform and the feed forward design process developed in the thesis.

Keywords: Generative design, Mass customization, 3D-Layout problem, Participatory design, Multi-Agent systems, Python programming.

PREFACE

The interest for developing this project roughly began during my undergraduate thesis where my thesis supervisor Dr.Anil Darshetkar introduced me to a book that he had written for helping the students formulate research and design questions called as 100+ decisions. The concept of Architect as a decision maker and the final design as a consequence of those decisions was introduced to me at the time and has been a subject of interest since then.

The need for such an approach was solidified when I worked as an architect on a large scale healthcare projects where there were so many decisions to take and numerous stakeholder requirements to cater to that analysing a configuration for the requirements or even checking the validity of it became a very difficult task in itself, let alone the generation of design based on them. Designing such critical spaces with no way of knowing the performance of it in meeting the needs and choices of the users can prove to be disastrous in the long run. This cemented the idea for the need for developing such a methodology. Configuration problems are very difficult questions in Architectural computation, but I can proudly say that in this thesis I took the first steps in trying to address the problem.

During the course Earthy(Computational Design for Earth Architecture) taught by Dr.ir. P. Nourian and ir. S. Azadi, I was introduced to the concept of configuration design and scientific methodical thinking about architectural design. In this course we developed a design game for developing the social housing for the Zatari refugee camp in Jordan. This gamified approach towards collective decision making was the main inspiration behind starting this project. The course Earthy, Dr. Nourian's doctorate work and the work done at Genesis lab by the students and the researchers form the background for the computational design theory and the subsequent work done in this project.

The core architectural theory and one of the most important subjects over which this proposal is based on is the Open building movement initiative and the work done by my second mentor Prof.ir. Thijs (M.F.) Asselbergs. The consultation and discussions with him helped me develop this project considering the architects point of view in design of these mass customized configurations.

Acknowledgements

The journey for this thesis has made me indebted to a lot of people for their guidance and support. Firstly I would like to take the opportunity to thank my mentor team Dr.ir.P.Nourian, Prof.ir. Thijs (M.F.) Asselbergs, ir. S. Azadi for their constant support, communication and timely advice throughout the project. Completing the graduation studio with them was one of the best experiences I had in my journey at TU Delft.

The topic of architectural configuration dealt with a lot of mathematical formulations and philosophical questions and I would like to thank Dr.ir.P.Nourian for the clear explanations, good references and for always guiding me in the right direction to avoid getting into rabbit holes. I thoroughly enjoyed not only the technical parts but also the philosophical parts of the amazing discussions that we had throughout the course of this project.

I would like to thank Prof.ir. Thijs (M.F.) Asselbergs for his critical questions about designing architectural configurations and for introducing me to the concept of open buildings. I would also like to thank him for organising interesting educational seminars like the infill festival where I was introduced to a new way of design thinking and was a source for a lot of the references used in the project. I am also indebted to him for his willingness to introduce my project to professionals in the related field for future development.

I would like to thank ir.Shervin Azadi for helping me learn Python programming and for always showing me the right tools and paths for solving the configuration problems. I would also like him for introducing me to the library Topogenesis which he developed with Dr.ir.P.Nourian which formed the basis for solving a lot of the problems stated in the report.

I would also like to thank all my fellow students in the Genesis lab for the fruitful discussions and the previous students for their inspirational lectures and projects.

I would like to thank my brother Siddharth for helping me build a website for my project and for explaining me some of programming concepts inspite of his busy schedule. Finally I would like to thank my family and friends for their constant support and motivation which allowed me follow my passions and chase my dreams.

Contents

ACRONYMS

GD: Generative Design.

ABM: Agent Based Modelling.

TCR: Total Closeness rating.

MIP: Mixed Integer programming.

SCIP: Solving Constraint Integer Programs. It is a solver in part of the Google OR tools library.

OR: Operations research

REL-CHART: Relationship Chart

MCDA: Multi criteria decision analysis

TOPSIS: Technique for Order of Preference by Similarity to Ideal Solution. It is a technique for MCDA.

CORELEAP: Computerised relationship planning

ALDEP: Automated Layout Design program

GH: Grasshopper a parametric modelling plugin for the 3d modelling software Rhinoceros.

WFC: Wave function collapse. An algorithm used in procedural content generation.

BIM: Building Information modelling.

CRAFT: Computerized Relative Allocation of Facilities Technique.

GSL: Generative Space Layout.

HOY: Hour of the Year.

LIST OF TERMINOLOGY USED:

1.Lattice: a numerical field within a discrete 3-dimensional space

2.Voxel/cell: Voxel represents a value within a lattice

3.Mass: A group of voxels representing the (maximum) volume of the buiding.

4.Massing: The process of obtaining the mass from a given design space

5.Envelope: Boundary of the mass

6.Zone: A group of voxels within a mass representing a specific attribute like spatial function.

7.Zoning: The process of obtaining the zone from a given mass.

8.Agent: An agent is anything that can be considered able to perceive its environment through sensors and act on this environment through actuators.A typical agent has a set of actions it can take and an id to trace its activity.

9.Spatial Agent: An spatial agent is an autonomous entity which acts, directing its activity towards achieving its spatial goals, upon an environment.

10.Agent Seed: An agent seed is the voxel where the agent starts performing its behaviour.

11.Agent Behaviors: The actions which the agent can perform to achieve its goals

12.Multi-Agent system: An agent-based system also can exist with several agents having the characteristics as described in the previous paragraph. Such a system is called as a multi-agent system

13.Stencil: The neighborhood definition in a discrete 3d space

14.Environment Field: Set of scalar values derived from an environmental simulation on a 3D grid having the same structure as the voxel grid.

15.Value Lattice: Set of scalar values corresponding to a certain simulation or calculation done on a 3D grid having the same structure as the voxel grid.

16.Desirability Lattice: Set of scalar values representing the result of the MCDA process done on a 3D grid having the same structure as the voxel grid indicating the desirability of a certain voxel for a certain agent.

17.Occupancy Lattice: Set of scalar values corresponding to the agent ID indicating the occupancy status of a voxel by the agents in the system on a 3D grid having the same structure as the voxel grid.

18.Availability Lattice: Set of boolean values indicating the availability of a voxel for occupancy on a 3D grid having the same structure as the voxel grid

19.Environment Field: Set of simulation scalar values on a 3D grid having the same structure as the voxel grid

20.Decision variable: A decision variable is a quantity that the decision-maker controls

21.Decision space: The range of values these variables can take on

22.Global decision variable: A decision variable that affects the performance of the entire design space
23.Local decision variable: A decision variable that affects the performance of an aspect of the design space

24.Design criteria: Attributes of the design space Design space: The range of values that these variables can take on

25.Performance indicator: An aggregated value that quantifies the performance towards a goal

26.Global performance indicator: A performance indicator that informs an entire design criterion.

27.Local performance indicator: A performance indicator that informs part of a design criterion

28.MCDA: Multi-Criteria Decision Analysis is a sub-discipline of operations research that explicitly evaluates multiple conflicting criteria in decision making

29.Function: mathematical function

30.Spatial function: Space usage or occupation type

31.Environmental factors: The conditions found in the environment which can have an impact on the design like noise, solar radiation, wind, precipitation, moisture etc.

32.Radiance: The density of radiant flux per unit of surface area and unit of solid angle.

33.Irradiance: The density of radiant flux (power) per unit of surface area

34.Unit: Unit in the context of spatial configuration can be defined as a entity in the space program which comprises of a self contained set of rooms collectively catering to a common function

35.Visibility: amount of unobstructed view from a point of interest expressed in percentage

36.Quiteness: Simplified calculations for the amount of sound perceived at a point in space based purely on the euclidean distance values from the noise source and the context geometry.

37.Closeness: The Euclidean or Manhattan distance between two points in space.

38.Actor: The stakeholders involved in the project who can make spatial decisions are termed as actors.

LIST OF ALGORITHMS:

[1] Mesh Voxelization Algorithm
The algorithm to convert a mesh into voxel array of specific size.

[2] Facade Closeness Lattice generation Algorithm
The algorithm to generate spatial quality matrix of closeness to a surface.

[3] Quietness Lattice generation Algorithm
The algorithm to generate spatial quality matrix of Quietness with respect to a source of sound.

[4] Sun-Access Lattice generation Algorithm
The algorithm to generate spatial quality matrix of Sun-Access of the voxel array for a certain list of HOY's.

[4] Visibility Lattice generation Algorithm
The algorithm to generate spatial quality matrix of Visibility of the voxel array towards a list of points of interest.

[6] Agent Occupy behaviour Algorithm
The algorithm to simulate the Agent occupy behaviour of the computational spatial agent.

[7] Attraction/Repulsion Agent behaviour Algorithm
The algorithm to simulate the Agent behaviour of attraction/repulsion towards another computational spatial agent.

[8] Agent origin behaviour Algorithm
The algorithm to simulate the process for selection of the origin for the computational agent.

1 Introduction

1.1 Background

The Global construction industry is growing exponentially. The world is expected to build 230 billion square metres in new construction, in the span of next 40 years[(Thibaut Abergel & Dulac, 2017)]. This puts a tremendous pressure on the construction industry to produce a lot of buildings in a very short amount of time. Often the scale of the building projects in developing cities around the world are massive and maintaining the design quality and providing a comfortable and sustainable solution for the projects becomes a complex and a challenging task. This often leads to repeating solutions which are identical and mass produced and which cannot respond to the variation in the context of the locations where they are constructed and the requirements of its many inhabitants. This old method of repetitiveness leads to a lot of social problems and there is a need for mass design customization to address this issue where the modularity is addressed differently.

Similar complexity can also be experienced in dense urban areas where often multiple functions are mixed together catering to the varying requirements of the users and the city. The complexity in the Architectural configuration can be said to have multiple layers like multi-dimensional (spatial complexity), multi-criteria (Multiple conflicting design criteria), multi-actor (Stakeholders and their requirements), multi-value (Socio-cultural, environmental value etc.) and addressing all the layers simultaneously is the key to solving this problem .In today's digital world where powerful computing power is easily accessible to the designers of the built environment, this problem of mass design customization opens up an opportunity to develop a digital solution to deal with addressing the complexity of architectural configuration in buildings.

Role of Informatics in construction has completely changed the way buildings are designed and built. Buildings today are first digitally built on a computer. This offers a tremendous opportunity for the stakeholders to evaluate the potential challenges that they will face in the realisation of the project. Strong simulation tools have also been developed to enable the designers to predict the performance of the buildings and optimize them for the requirements of the project. Inspite of this the solutions generated using these tools will only be as good as the problem solving capacity of the designer using them. Informatics can truly impact the design quality of these complex projects when it is involved in the design thinking itself and not as a means of improvement of the analogue design.

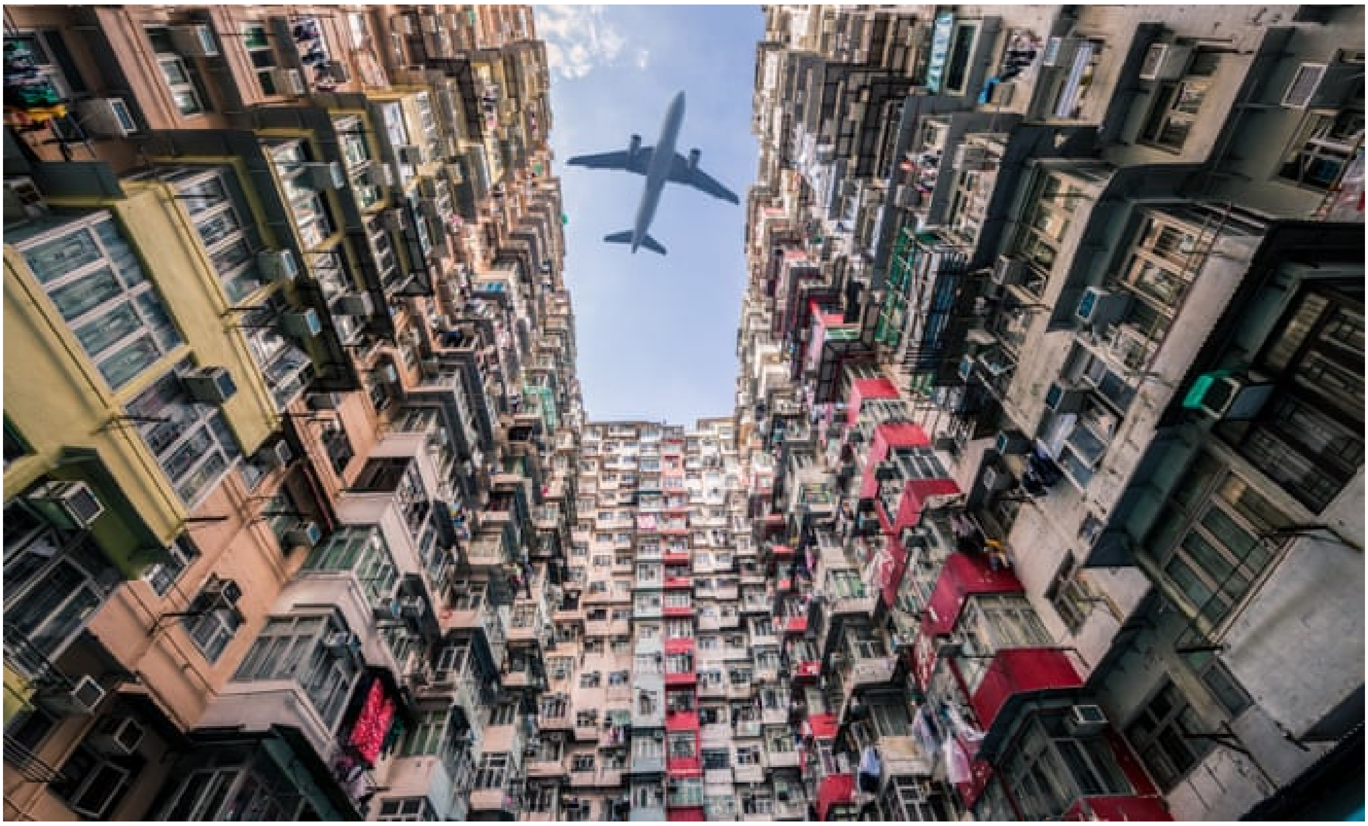


Figure 1.1: An example of a problematic configuration for a residential complex in Hong Kong where light, ventilation, privacy and quality of life of the occupants is affected by a badly configured building.

1.2 Motivation

The main task which Architects are entrusted with is to generate design options for a project which resolves and satisfies all the layers of complexity of Architectural configuration as described in [1].An example of this would be solving a spatial configuration problem where the performance aspects like energy, climate, comfort, functionality are satisfied as well as the preferences of the client and the occupants while maintaining the architectural concept. For a complex project involving large scale program of requirements and many occupants it becomes a mammoth task to iterate through all the possible design options. A digital tool using Generative design can offer a chance to computationally solve this problem at an early design phase itself.

The design decisions taken during the early design phase are crucial, because they have an enormous impact on the final building performance (how well the building will meet the requirements) as well as other aspects such as costs. Design changes occurring in later stages often come at a major time cost, producing delays in further stages or reducing the amount of time available for properly addressing certain features.Therefore, choosing the desirable design direction during the early design phases is essential.It is also challenging as it means to find a proper way to combine and integrate functional, technical, aesthetical, social, financial aspects of the project. A systematic and methodical way of approaching the configuration problem will lead to a system where the impact of decisions taken at each stage is traceable and can be improved. It will not only produce design configuration variants that can be compared and quantified but it will make the Architects and engineers more aware of the consequences of their design decisions.

The design approach proposed in the thesis proposes a methodical and traceable way of designing buildings. Through the tool developed in this project Design informatics will be directly involved in the design thinking of a project by feed forwarding the design process and helping Architects produce high performing architectural configurations.

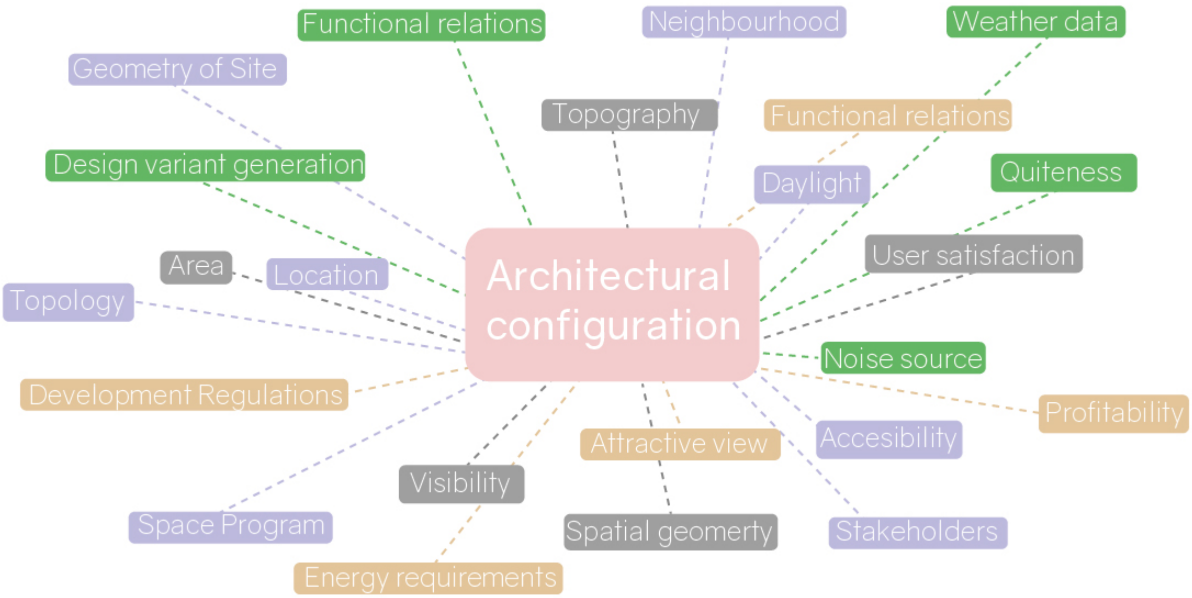


Figure 1.2: Factors to consider while solving the Architectural configuration problem.

2 Research Framework

2.1 Problem Statement

In the process of Architectural design spatial configuration plays a crucial role in the performance of the design both in terms of its qualitative (Spatial quality, interactivity and accessibility etc) and quantitative aspects (Energy performance, Energy generation capacity, etc).If the spatial configuration part is not analysed and designed correctly it might lead to a problematic building. Some examples of problematic buildings include buildings that are too costly to operate due to their energy demand. Unoccupied office and retail spaces in buildings due to poor accessibility. Isolated empty spots in large buildings which are not safe. Excessive use of mechanical systems to deal with climatically uncomfortable spaces. Unoccupied spaces in buildings due to poor functional planning.

The challenge of producing good performing design variants for complex large scaled projects by an analogue process becomes a very difficult task for the designer since a lot of factors need to be considered simultaneously. *Solving this 3d layout problem involves solving the puzzle where the zoning(placement of spaces based on their closeness requirements with each other) ,routing(providing efficient means of access to all the spaces in the configuration), spatial(geometric, aesthetic and comfort) and environmental (energy performance,natural lighting, quietness) goals need to be met at the same time.* A definitive computational method which can address all the goals together has not been devised yet and needs to be addressed if the challenge is to be solved. Since the problem is case dependant the solution devised should be modular and flexible to adjust to the requirements of the case a computational design solution would be appropriate for solving the problem.

The second problem which the thesis deals with is the problem of *urban inequality in terms of a lack of opportunity for the occupants to participate in the design of the spaces that they are going to inhabit.* More often than not the Architectural design and configuration is planned by the architects to satisfy the architectural vision and the requirements of the stakeholders having the maximum monetary investment in the project. Such an undemocratic approach lacks the inclusive element in the design of the spaces for its occupants creating unsatisfactory solutions which leads to modification in spaces by the occupants after they are built, leading to a lot of resource wastage.

If the Architects and engineers of tomorrow want to design and build sustainable, comfortable and contextually best suited buildings then the issue of problematic buildings due to poor spatial configuration and non-inclusiveness of the occupants in design process should be addressed.

2.2 Sub–problems:

1.Selection of an appropriate computational approach for solving the problem: The formulation of the 3d layout problem and searching for the appropriate computational method which is flexible and adaptable for various situations to solve the problem is a crucial step towards solving the problem.

2.Creation of a participatory framework for the stakeholders: Determining the level of participation and the means of participation for the various stakeholders involved in the project needs to be formulated along with the level of impact that their decisions have on the final outcome of the design needs to be developed.

3.Methodology for analysing the design variants: There is a need to develop a formal evaluation framework which can inform the user on how close the generated design variant has come to achieving its goals as defined in the problem statement.

2.3 Research Objectives:

The main research objective is to develop a design methodology which can solve the 3d layout problem in its full sophistication considering all the layers of complexities in a methodical manner.The methodology would be implemented by development of an interactive design tool which will assist the designer to generate a range of suitable design variants for a given set of constraints and improves the awareness of the designer in terms of the consequences of design decisions taken in the process of spatial configuration on the final performance of the design.The tool would be intended to act as a proof of concept rather than an optimised solution in terms of its computational performance.

2.4 Expected End Products:

The expected end product will be an interactive tool which can solve a 3d layout problem considering the choices and decisions of the various stakeholders involved in the project.The tool developed in the graduation project would be a python language based tool with jupyter notebooks providing the interactivity for the user. The tool developed will be tested on a case and the methodical step by step process of design will be explained by a collection of Jupyter notebooks and 3D Rhino models with grasshopper scripts.These notebooks and models will be adaptive enough to change them for any desired use case or a design problem. The future roadmap for the development of the different modules of the tool will also be proposed.

2.5 Scope and Limitations:

Scope:

The tool developed in the thesis aims to be a proof of concepts of the various features developed and discussed in the thesis and does not intend to be optimized in terms of software development or computer programming.The 3d Layout problem in its full sophistication is an extremely complicated and a large problem and solving the whole problem is very difficult in the given time frame of the project. So in the graduation thesis the framework for solving the complete problem will be developed and the actual programming will be done only for a certain set of modules in the framework due to the limited time. The methodology developed in this project assumes that the user of the tool has the knowledge of 3d modelling techniques and is able to provide the necessary inputs to the program at the various stages. This ensures that the deign process is not limited or hampered by the proposed methodology but provides the designers options which can be possible in the proposed concept of the building. The tool developed in this thesis will be developed on the base of the selected generative design method and is bound by the limitations of the same.

Limitations:

The approach considered in the thesis does not claim to be the only way of solving the problem of Architectural configuration since the problem is abstract and can have multiple solutions. The approach taken can be classified as a 'Means-orientated designing' which is rather a journey of exploration, in search for unknown design solutions for goals yet unknown when the goal-generating context changes.

The performance criteria modules defined in the thesis is based on the specific case under consideration. It tries to be as modular as possible so that they can be adapted to different cases, but can potentially vary according to the conditions of the case and the preferences of the designer.

The performance criteria modules implemented in the tool does not claim to be the most accurate way of simulating that particular aspect but is intended to provide rough estimate for the decision-making process. It is assumed that if the tool is further developed the calculations and simulations done will be more accurate and would likely follow some pre-defined standards.

This is due to the fact that the tool intends to show a new methodology and not dive deep into the specifics of the performance criteria considering the limited time span available for graduation.

2.6 Methodology:

The problem of Architectural configuration is a complex layered one as mentioned in [1.1].The methodology for solving the problem in the project will be as follows:

Understanding the problem complexity: The first step in the process would be to understand all the layers and their inter-relations.This step will be followed by a literature study of previous attempts at solving the 3d layout problem.

Case studies on Participatory design: Case studies will be performed on participatory design projects to understand the stakeholder involvement in various stages of a design proposal.

Literature review on design criteria: The various design criteria in a design project will be identified and literature review will be done to understand the way in which they can be calculated or quantified.

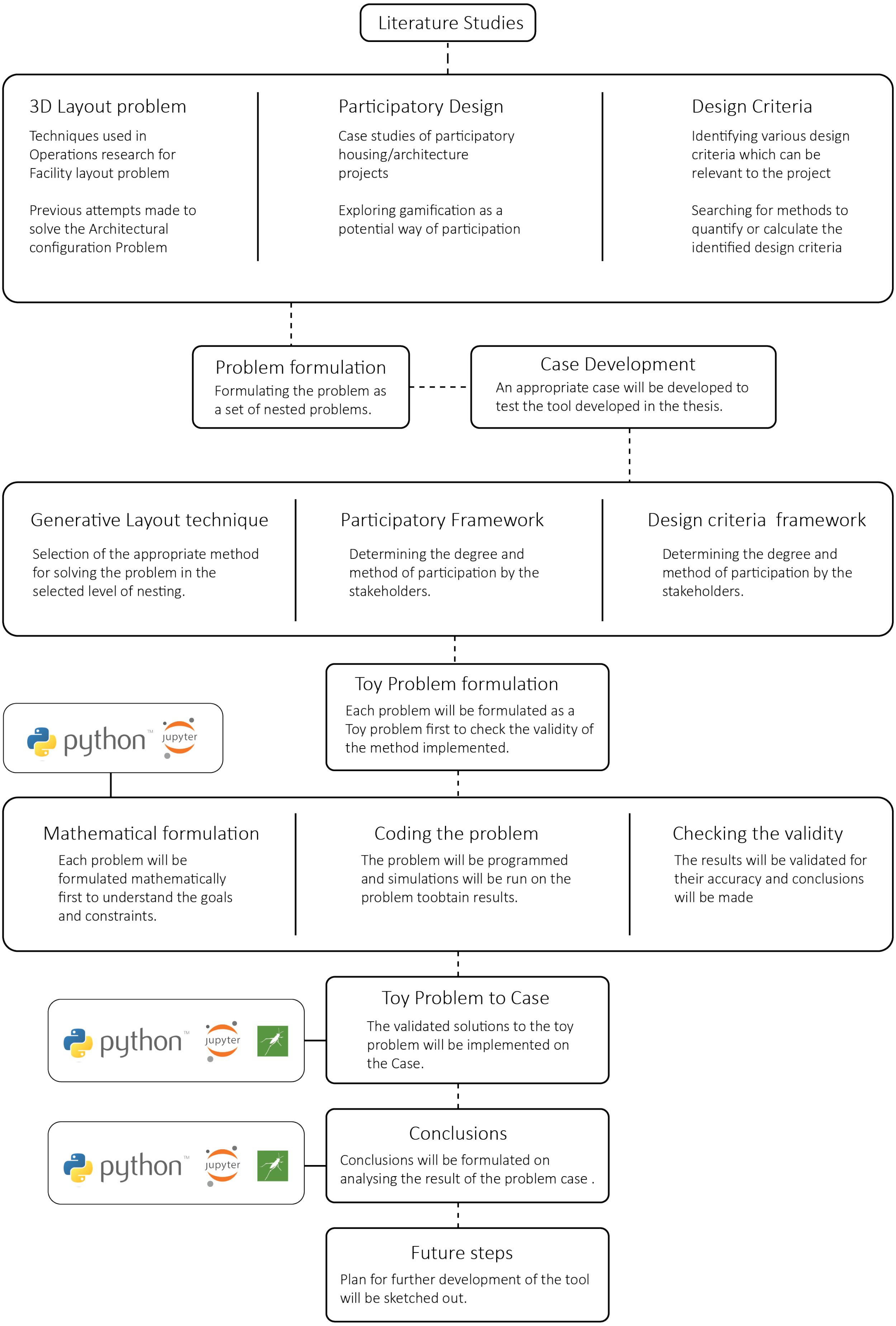
Generating a nested set of problems: Once these background steps about literature research are completed then the 3d layout problem will be methodically broken down and formulated as a nested group of smaller problems.

Defining a case: An example case will be formulated to showcase the developed tool and methodology.

Toy problem formulation: The next step will involve formulating a toy problem for each one of the smaller problem and devising a mathematical formulation to solve it.When the toy problem is resolved and can offer a proof of concept then the problem will be translated into the selected case.

Solving the case: The conclusions from the toy problems will be used to solve the case defined previously and the results and conclusions will be made

Further development:A future road map will be proposed for further development of the tool.



Literature Studies

3D Layout problem

Techniques used in Operations research for Facility layout problem

Previous attempts made to solve the Architectural configuration Problem

Participatory Design

Case studies of participatory housing/architecture projects

Exploring gamification as a potential way of participation

Design Criteria

Identifying various design criteria which can be relevant to the project

Searching for methods to quantify or calculate the identified design criteria

Problem formulation

Formulating the problem as a set of nested problems.

Case Development

An appropriate case will be developed to test the tool developed in the thesis.

Generative Layout technique

Selection of the appropriate method for solving the problem in the selected level of nesting.

Participatory Framework

Determining the degree and method of participation by the stakeholders.

Design criteria framework

Determining the degree and method of participation by the stakeholders.

Toy Problem formulation

Each problem will be formulated as a Toy problem first to check the validity of the method implemented.

Mathematical formulation

Each problem will be formulated mathematically first to understand the goals and constraints.

Coding the problem

The problem will be programmed and simulations will be run on the problem toobtain results.

Checking the validity

The results will be validated for their accuracy and conclusions will be made

Toy Problem to Case

The validated solutions to the toy problem will be implemented on the Case.

Conclusions

Conclusions will be formulated on analysing the result of the problem case .

Future steps

Plan for further development of the tool will be sketched out.

Figure 2.1: Research Methodology Diagram

3 Literature Research

3.1 Previous attempts at solving the 3D layout problem:

Several attempts have been made over the years to solve the 3d layout problem from an architectural point of view as well as the facility planning point of view.The facility layout planning problem in the industrial engineering discipline has close relations to the architectural space layout one which this project is trying to solve. In the literature research about the previous attempts some common algorithms in facility layout problem were studied along with the most commonly used methods for solving the generative space layout problem as studied by (Du, Turrin, Jansen, van den Dobbelsteen, & Fang, 2020) and (Lobos, 2010) in their respective papers which reviewed the attempts made so far.

3.1.1 Physics simulation based method:

In the physics simulation based method space configurations are generated by application of physical forces to the spaces. The layout generation process is transformed to calculate the equilibrium between different forces for example the attraction and repulsion in a spring system. In this method, a space is represented as a circle or rectangle, and the connection between spaces is represented by the string between circles or rectangles. In order to perform a topological resolution, spaces are represented as circles, and attraction and repulsion forces are applied to strings until the equilibrium is reached. For the geometric resolution part space locations are manually changed by the user. This process also removes overlaps and gaps and can bring a change in adjacency and connections. The final layout is also chosen manually by sorting the options based on certain performance criteria defined by the user.

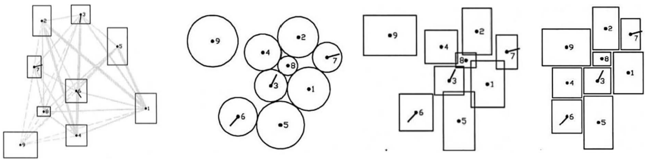


Figure 3.1: Physics simulation based method example as seen in (Arvin & House, 2002)

3.1.2 Constraint based non-linear method:

In a constraint based non-linear method multiple optimised solutions can be generated inside a rectangular floor plate like a lot of non-linear methods. The constraints consist of physical and functional requirement of spaces. Dimensional constraints determine the size and the functional constraint determine the position.The constraints are converted into mathematical formulations which are then used in the solver to obtain multiple valid options as seen in (S. Li, Frazer, & Tang, 2000)

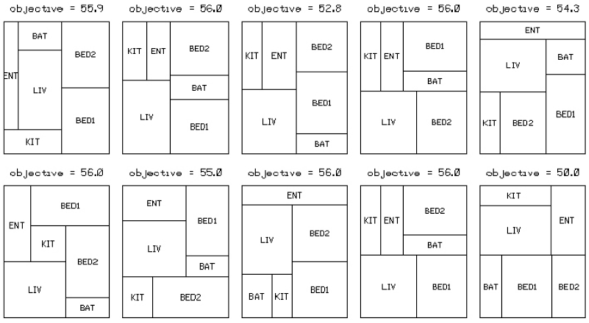
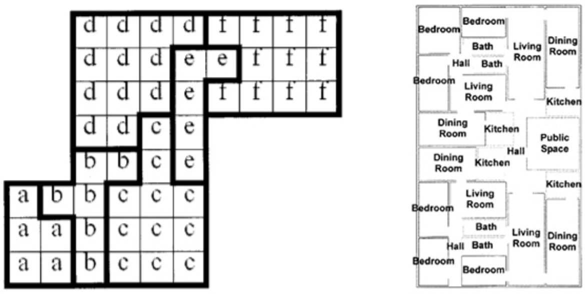


Figure 3.2: Constraint based non-linear method example as seen in (S. Li et al., 2000)

3.1.3 Gradient based and Evolutionary algorithms based:

This approach allows for the inclusion of mathematical optimization and subjective decision making in the conceptual design stage. Gradient based algorithms and Evolutionary algorithms are used in the process of finding the optimal solution in the global decision space. This approach defines the available space as a set of grid squares and uses an algorithm to allocate each square to a room activity. Mathematical optimization allows the user to interact in the design process without worrying about the background complex operations through an "object-oriented representation" of it. Designers can change objects and constraints during the process.

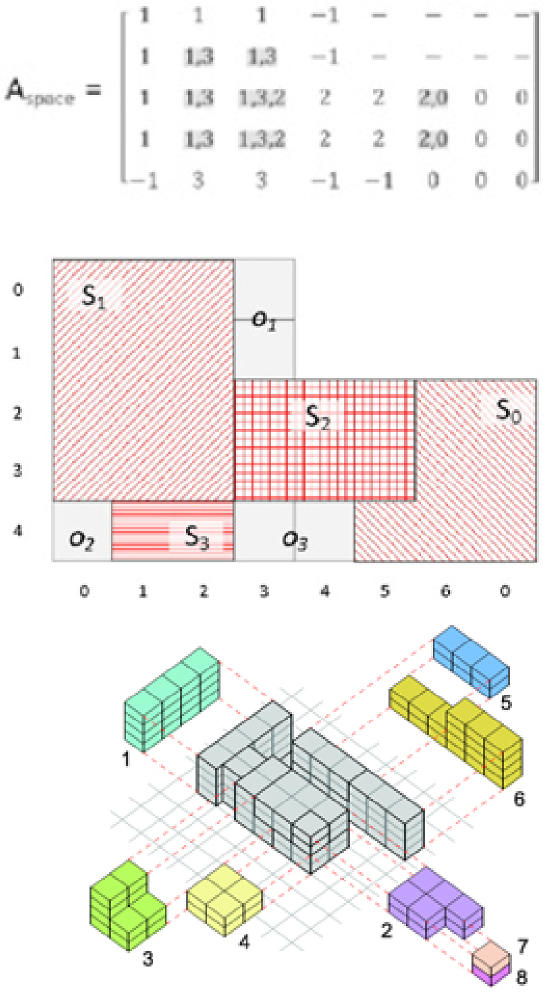


		d	d	d	d	f	f	f	f
		d	d	d	e	e	f	f	f
		d	d	d	e	f	f	f	f
		d	d	c	e				
		b	b	c	e				
a	b	b	c	c	c				
a	a	b	c	c	c				
a	a	b	c	c	c				

Figure 3.3: Sample fixed gird allocation layout and the optimized variant of it as seen in (Michalek, Choudhary, & Papalambros, 2002)

3.1.4 Cell assignment method:

In this method the building geometry is predefined and the whole mass is divided into a 3d cell grid or a voxel grid. First a matrix is defined which represents all the cells in the building, and the value in the matrix represents which space is assigned to the corresponding cell. Secondly, spaces are assigned to the cells in the building geometry correspondingly. The variant generation is done, by changing the values in the matrix, the feasible layout can be obtained satisfying both geometric and topological requirements.



$$A_{space} = \begin{bmatrix} 1 & 1 & 1 & -1 & - & - & - & - \\ 1 & 1,3 & 1,3 & -1 & - & - & - & - \\ 1 & 1,3 & 1,3,2 & 2 & 2 & 2,0 & 0 & 0 \\ 1 & 1,3 & 1,3,2 & 2 & 2 & 2,0 & 0 & 0 \\ -1 & 3 & 3 & -1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3.4: Example of Cell assignment method as seen in the work of (Dino, 2016)

3.1.5 Space splitting method:

In this method, a predefined floor plan is split recursively following a sequence, which is stored in a data tree. The node in the data tree represents a space, and the value in the node represents the dimensional information for where the splitting line locates, like the space area. First, a floor plan is defined by users.Second, space dimensions and adjacency's are coded into a data tree, which can be varied for layout alternatives. Third, the initial layout is recursively split based on the tree data finally, the final layout is generated after all splits. There are different slicing methods like slicing by distance, slicing by ratio, and slicing by area. Some splitting strategies can help to generate irregular spaces.

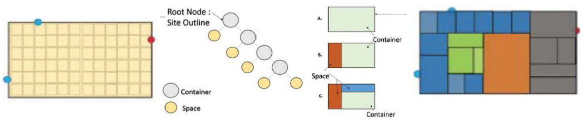
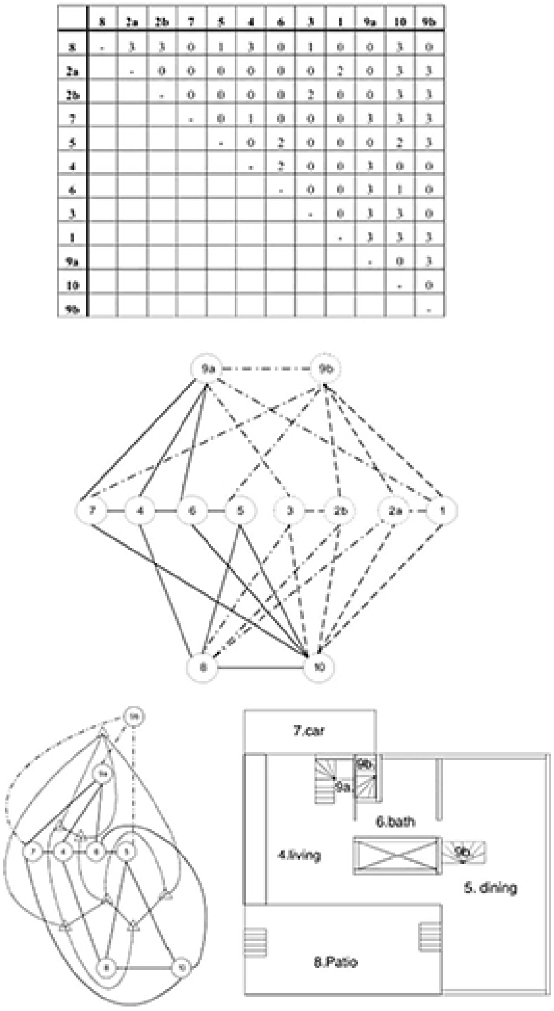


Figure 3.5: Example of Space splitting method as seen in the work of (S. Das, n.d.)

3.1.6 Planar graph method:

In this method, space adjacency's are transformed to a planar graph, and algorithms for graph theory are used to convert the planar graph into a feasible space layout. The process of space generation has two clear divisions of topological and geometric resolution in this method .First, the space adjacency preferences are stored in a 2D matrix, which can be varied for alternatives then, the matrix is transformed to a planar graph, in which nodes represent spaces and links represent connections algorithms are used to convert the planar graph to a graph which can be converted to a feasible layout, like a dual graph, in which the links can be divided into multi-floors .The final space layout is obtained by inserting geometric information to the graph.



	8	2a	2b	7	5	4	6	3	1	9a	10	9b
8	-	3	3	0	1	3	0	1	0	0	3	0
2a		-	0	0	0	0	0	0	2	0	3	3
2b			-	0	0	0	0	2	0	0	3	3
7				-	0	1	0	0	0	3	3	3
5					-	0	2	0	0	0	2	3
4						-	2	0	0	3	0	0
6							-	0	0	3	1	0
3								-	0	3	3	0
1									-	3	3	3
9a										-	0	3
10											-	0
9b												-

Figure 3.6: Example of Cell assignment method as seen in the work of (Dino, 2016)

3.1.7 Occupant-trace based method:

In this method, a space layout is generated based on occupant tracks, which are obtained by simulating occupant movements. First, occupant movements are simulated, which are controlled by external forces of attraction and repulsion, and affected by the environmental elements, like obstacles and destinations second, the simulated occupant tracks are used as circulation paths third, the circulation paths are meshed and converted to feasible spaces. Finally, the left-over spaces are used as the volumes to accommodate functional spaces.

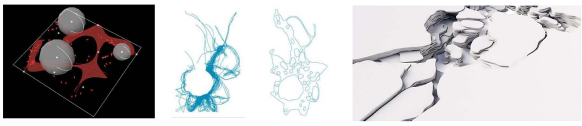


Figure 3.7: Example of occupant trace as seen in the work of (Ghaffarian, Fallah, & Jacob, 2018)

3.1.8 Machine learning method:

In this method, a model of machine learning is trained based on the dataset with real cases of space layouts, then the trained model is used to generate space layouts with certain inputs. The machine learning method is a method to mimic the decision-making process of architects based on their expertise and experience, without the need to understand thoroughly the logic behind the experience.

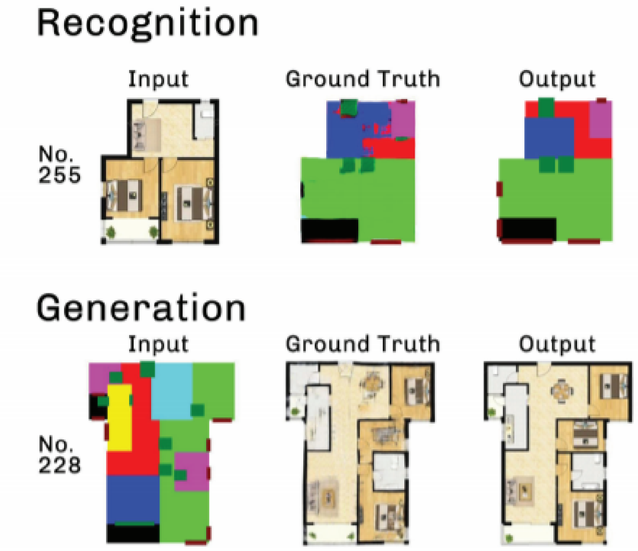


Figure 3.8: The machine learning based method for space layout generation as implemented by (W.Huang, n.d.)

3.1.9 Facility layout problem Generation method:

This method is used to design facilities which are yet to be built.The most commonly algorithms developed for this method involve the concept of TCR or Total Closeness Rating. One such algorithm in this method is the construction algorithm CORELEAP (Computerised Relationship layout planning) It calculates the Total closeness rating as the closeness rating given to a facility or a function based on the activity chart developed for the complete facility.Then the algorithm systematically places the facilities around the main facility having the maximum closeness rating.The main goal of this method is to optimize the process flow in a facility my minimising the handling costs of the resources.

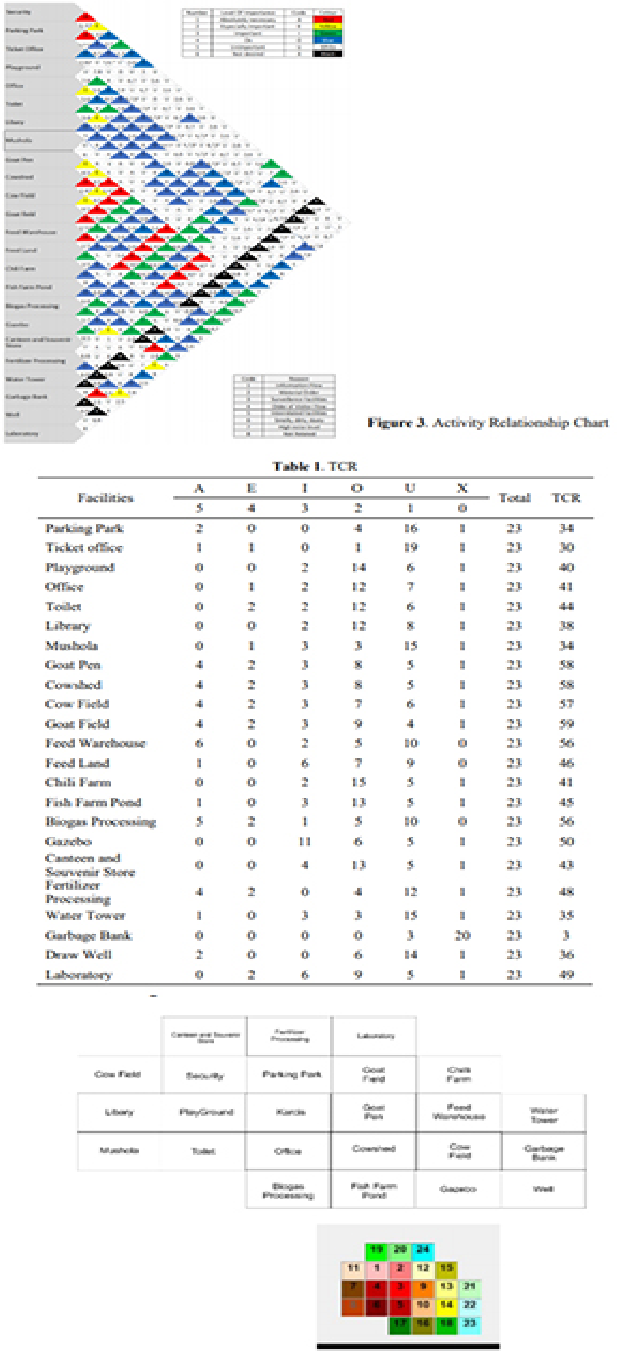


Table 1. TCR

Facilities	A	E	I	O	U	X	Total	TCR
	5	4	3	2	1	0		
Parking Park	2	0	0	4	16	1	23	34
Ticket office	1	1	0	1	19	1	23	30
Playground	0	0	2	14	6	1	23	40
Office	0	1	2	12	7	1	23	41
Toilet	0	2	2	12	6	1	23	44
Library	0	0	2	12	8	1	23	38
Mushola	0	1	3	3	15	1	23	34
Goat Pen	4	2	3	8	5	1	23	58
Cowshed	4	2	3	8	5	1	23	58
Cow Field	4	2	3	7	6	1	23	57
Goat Field	4	2	3	9	4	1	23	59
Feed Warehouse	6	0	2	5	10	0	23	56
Feed Land	1	0	6	7	9	0	23	46
Chili Farm	0	0	2	15	5	1	23	41
Fish Farm Pond	1	0	3	13	5	1	23	45
Biogas Processing	5	2	1	5	10	0	23	56
Gazebo	0	0	11	6	5	1	23	50
Canteen and Souvenir Store	0	0	4	13	5	1	23	43
Fertilizer Processing	4	2	0	4	12	1	23	48
Water Tower	1	0	3	3	15	1	23	35
Garbage Bank	0	0	0	0	3	20	23	3
Draw Well	2	0	0	6	14	1	23	36
Laboratory	0	2	6	9	5	1	23	49

	Canteen and Souvenir Store	Fertilizer Processing	Laboratory		
Cow Field	Security	Parking Park	Goat Field	Chili Farm	
Library	PlayGround	Kantin	Goat Pen	Feed Warehouse	Water Tower
Mushola	Toilet	Office	Cowshed	Cow Field	Garbage Bank
		Biogas Processing	Fish Farm Pond	Gazebo	Well

Figure 3.9: CORELEAP algortihm implementation to design a facility as seen in the work of (Jati, Rahayu, Salsabila, & 'Azzam, 2020)

3.1.10 Facility layout problem Generation with improvement step method:

This method is used to improve the existing facility or used in combination with a generative algorithm as explained in previous paragraph to improve its effectiveness.The most commonly used algorithm in this method is called as the CRAFT (Computerized Relative Allocation of Facilities Technique) algorithm. The algorithm starts with an initial layout and improves the layout by interchanging the department's pair wise so that the transportation cost is minimized. The first step involves studying the effect of interchanging the departments under consideration on the efficiency of a given layout. Then step by step the departments are interchanged till a point where interchanging the departments doesn't improve the performance of the facility.

3.1.11 Selection of an appropriate method:

Choosing an appropriate methodology is essential as a first step towards developing the spatial configuration logic.The following factors could be considered for the method selection in a space layout problem as described by [(Du et al., 2020)].

1.Feasibility: whether the generated layouts are feasible or not, considering the requirements for practice. User-friendliness: whether the method is easy to be controlled by designers.
Generation speed: how fast the method can generate layout solutions.
2.Variance: how easy the method is used to generate variants.
3.Capability of multi-floor: how easy the method is used to generate multi-floors. This is important, as in practice most buildings have multi-floors.
4.Capability of irregularity: whether the method can generate an irregular boundary or space, except for rectangle. The more space forms the method can create; the more options designers can have.
5.Necessity of predefined boundary: whether the method needs a predefined boundary or not. In practice, the boundary design might happen before or after space layout design, and it can also be the result of interior space layout design. This requires that the GSL method is capable to use a layout boundary predefined by designers, as well as to generate the layout boundary by itself.

3.1.12 Conclusion:

Considering the intent of the project as a platform for Architectural configuration exploration and combinatorial design variant generation for mass customisation using modular strategies so the following properties become very important while choosing a generative space layout method.

1.The idea is to generate Architectural configurations and not two-dimensional floor plans. The capability of the chosen method to generate three dimensional options or multi-floor options is critical for the project.

2.Flexibility is also very important in terms of design exploration and the configuration form so the property of irregularity capacity and changing geometry also becomes very important for the project.

3.The project is based on the premise that the software program developed will not replace the designer but help in generating configurations in the conceptual 3d blocks developed by the designer. The project intends for the designer to interact and work with the program to explore design variants so the constraints of fixed boundaries are necessary for the generative space layout method to respect the design developed by the user.

The problem in its full sophistication will not be solved by a single method since the requirements for each level of nesting of the problem is different.There needs to be combination of selected methods to solve a specific problem in the configuration problem as a whole.It would be interesting to explore the *cell-assignment based method to solve the zoning level problems in the configuration and other methods like space-splitting or the facility layout method would be interesting to look at more at a floor plan level.*

Multi-agent systems and cell assignment method as a Generative space layout strategy:

Currently Agents are one of the main fields of interest in computer science, artificial intelligence (AI), and complex system theory. Agent or agent-based modelling has been used in many diverse applications and does not have a fixed definition. According to [(Stuart Russell, 2010)] an agent is anything that can be considered able to perceive its environment through sensors and act on this environment through actuators. [(Macal, 2016)] described the essential characteristics an agent should have:
Identifiable: discrete individual with a set of features and rules (mathematical or logic) that govern behaviour and decision-making capacity.
Locatable: settled in an environment with which it interacts and also in which interacts with other agents

Goal driven, Self-contained, and be Flexible, and have the ability to learn and adapt its behaviour through time-based experiences.

An agent-based system also can exist with several agents having the characteristics as described in the previous paragraph. Such a system is called as a multi-agent system. [(Rocha, 2017)] describes multi-agent systems as 'a loosely coupled network of problem-solving entities (agents) that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity (agent).

In the thesis project multi-agent system can be used in the cell assignment method of generative space layout to generate the architectural configuration variants. The agents can be trained to possess the necessary qualities and intelligence which can solve the multi-layered complexity of the problem of architectural configuration as mentioned in the previous chapter.

Examples of Multi-agent systems and cell assignment method:
A multi-agent approach combined with evolutionary solver was used by [(Guo & Li, 2017)]in a project which explored generative space layout using cell assignment method.Their paper argued that unlike any other design constraints in space layout topological relationships are fundamental for architectural configuration. A space can still be usable if the geometry is not of the proper shape or size but if the connection between the spaces is missing that can prove to be disastrous to the configuration of the building. The chances of such a situation increases with an increase in the number of the rooms. Hence starting an optimization process with a correct topological relationship between the spaces will narrow the search space for the optimization process.

The topology finding process was implemented through a multi agent system, where rooms were represented as bubble-like agents and connections as strings linking the agents. The bubble-like agents were divided into two types a sphere like agent with centre point and a buffer distance and a capsule shaped agent with end points and buffer distance. All the linear spaces in the building like corridors and staircases were represented by capsule like agents and individual spaces were represented by spherical agents.

In order to control the behaviour of the agents during the interaction stage rules were defined based on three operations move, push and swap. The operations modified the positions and orientation of the agents in the system. The move operation [A,$\vec{m}$] was defined as moving the internal geometry of agent A along vector $\vec{m}$ .The pushing operation [A,P$\vec{m}$] is pushing the agent A along point p on its geometry with a vector $\vec{m}$ as seen in 3.11. The swap operation swaps the position of the agents as seen in fig 3.12.

The rules were as follows:
Attraction: For a pair of connected agent A_1 and A_2, the call operations are Push [A,$P_1\vec{m}$]and Push[A,$P_2\vec{m}$] where P_1 and P_2 are the closest points between the geometries of A_1 and A_2,and is the vector heading from P_1 to P_2. The length is positively related with the distance between P_1 and P_2. This rule is applied to all connected agents in the system.

Repulsion: For a pair of unconnected agents, A_1 and A_2 are in the same level when the distance between them is less than the sum of their buffer distance, and the call push operation is Push [A,$P_1\vec{m}$] and Push [A,$P_2\vec{m}$]. The meanings of the parameters are the same as the meanings in the attraction rule, except that the length of vector $\vec{m}$ is negatively related to the distance between P_1 and P_2.

Swap: For any two pairs of connected agents that are at the same level, one agent from each pair is chosen if the connection strings intersect. The chosen agents are A_1 and A_2. Thus, the call operation becomes Swap (A_1 , A_2)

Compression: This is an optional rule that may be involved if minimizing the volume of the entire building is necessary. The definition of this rule is that, for every agent A, the call push operation is Push (A,p,-mag.p), where p is the position of the agent, and mag is the magnitude that reduces by a specified reduction rate in each iteration.

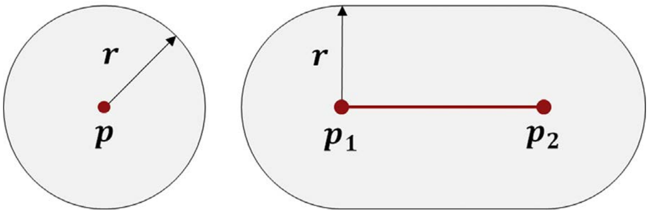


Figure 3.10: Left: sphere-like agent.Right:capsule-like agent. (Guo & Li, 2017)

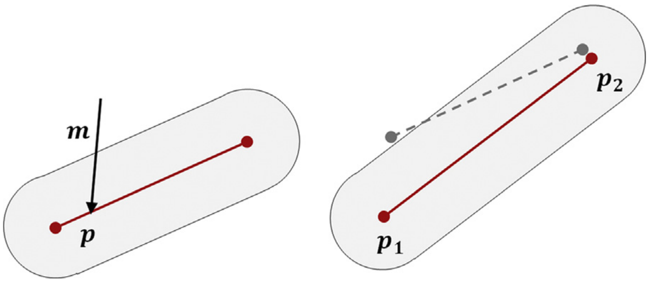


Figure 3.11: Effect of the push operation on a horizontal capsule-like agent. (Guo & Li, 2017)

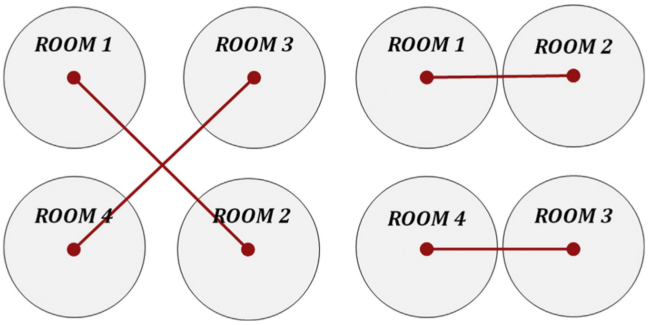


Figure 3.12: Swap operation between Rooms2 and 4 avoids the incorrect topology caused by the intersection.(Guo & Li, 2017)

Based on the rules the simulation generated topologically acceptable configurations in an abstract manner with no geometry defined. This model was then converted into a 3D cell-based model where the conversion applied the principle of the Voronoi diagram. Cells are assigned to the closest agent on the current level according to the horizontal distance.[3.14] Vertical capsule-like agents, such as stairs, can consume cells in different floors. This cell will not be assigned to any rooms if the closest agent of a cell is one of those extra agents that represent exterior spaces. This strategy ensured the connectivity between entrances and exterior spaces.

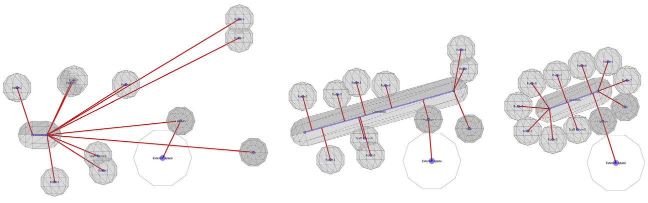


Figure 3.13: Example of the multi-agent system. **Left**:initial status. **Middle:**status during the interaction. **Right:**stable status.(Guo & Li, 2017)

Following the conversion of the model an evolutionary optimization process was implemented which considered the converted model as the parent which is further replicated and mutated into child layout. Then the child layout is evaluated if its better than the parent one it will replace the parent one or else it will be discarded.The mutation is achieved by pushing the faces of rooms. A set of co-planar faces is randomly selected and pushed forward or backward random steps. Cells that are affected by the movement change their owners. If only one face is selected, this face may be divided into two, and only one divided face is pushed. After a push operation, all faces refresh their pair references. 3.13 illustrates a 2-D version of this process, where the middle face is selected and pushed toward the left for one step. Thus, the number of cells owned by the left room increases, whereas the number of the right decreases. The mutation may also contain a global adjustment to explore the search space more effectively. The global adjustment has a certain probability to randomly select and swap the position of rooms and thus alter the layout significantly which is not possible by just pushing the faces.

The quality of the mutated layout is determined by the cost values returned by evaluators. Each evaluator corresponds to one architectural criterion, and the cost values are weighted summed. If the result is great, then the quality of the mutation is worse. Four criteria are adopted (topology, shape, dimension, and aspect ratio and building shape) despite the many criteria used in actual architecture design. Other requirements, such as climate, energy, and cultural issues, may be involved in the different implementations of the evaluator. Given that the evaluation result is calculated through the weighted sum of all cost values, the number of the evaluators is flexible, and new evaluators can be easily involved. The calculation can be expressed as:

$$E(x) = \sum W_i * E_i * (x)$$

Where x is the layout being evaluated, Ei denotes the evaluators corresponding to different criteria, and Wi is the corresponding weights.

The following limitations were observed in this approach. Generation of non linear or curvilinear spaces was not possible in this method. The proposed approach dealt with only the criteria for geometry and topology which does not represent the complete set of complexity involved in the configuration of real architectural projects. The process of optimization involves calculation and evaluation of each option which becomes very time consuming if environmental factors like daylight are concerned. Lastly more often than not architectural configuration has two groupings one higher level grouping where relationships between typology of spaces are defined and a lower-level grouping where connections and relations between the spaces in the individual typology of spaces are defined. This factor is not considered in this approach.

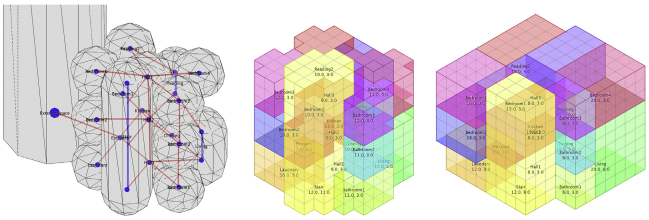


Figure 3.14: Screenshots of the entire process.**Left:** generated multi-agent layout.**Middle:** converted grid system.**Right:** Optimized grid system.(Guo & Li, 2017)

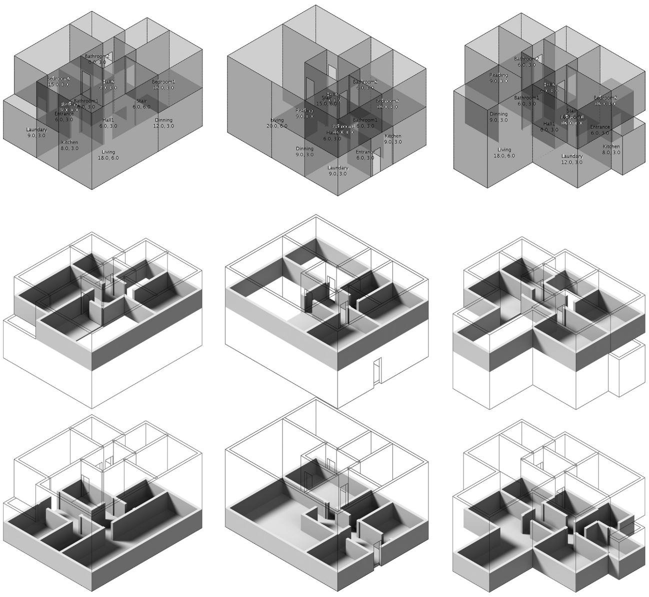


Figure 3.15: Generated layout from the same architectural program rendered as separated layers.(Guo & Li, 2017)

Another interesting study based on cell assignment method was done by[(Dino, 2016)] where a novel approach of Precedence-Based Layout Configuration Heuristics (P-LCH) was introduced. The author developed a tool called Evolutionary Architectural Space layout Explorer (EASE) for early architectural design that aims to facilitate better informed decision-making under spatial constraints using evolutionary optimization. EASE tackles multi-floor, unequal-area layout problems starting with arbitrary a priori architectural forms. The intention of this method is to generate a finalized layout, but a number of low-resolution solutions that provide insight into the solution space and initiate further exploration. The starting point for the tool involves discretising the building form into equal sized voxels. This voxelated form is represented by a 3d Boolean matrix (A_{init}) . The spaces in the architectural brief also needs to be discretised in the same manner such that the total number of spatial units is equal to the total number of voxels in the building. This results in transformation of the layout problem into a space allocation problem where each space with n number of units needs to be matched with n number of voxels. As the spaces are assigned to voxels, the space indices that occupy the voxels are registered in a second 3D matrix (A_{space}) with the same voxel structure as (A_{init}) . P-LCH takes as input the building representation (A_{init}) and the space sizes. Thereon, three steps are repeated for each layout instance which are as follows:

Step 1: P-LCH first generates for each space Si a rectangular prism Pi that aims to approximate its total number of voxels NR_{iv} as much as possible. The prisms' dimensions are determined by generating random values over [0, 1] for their width (ratio W), depth (ratio D) and height (ratio H) ratios. Then the prism's actual width is calculated by multiplying ratio W with the cube root of the voxel size divided by its ratio W, ratio D and ratio H. The same is calculated for $heigth_i$ and $depth_i$ in the same manner. To place P_i in the building, first a centre point (C_x, C_y , C_z) is randomly generated) where:

$$0 \leq C_x^i < depth_b and 0 \leq C_y^i < width_b and 0 C_z^i < height_b$$

The building voxels that will be occupied by $P_i$in A_{init} are calculated from the upper-left to the lower-right voxels If the prism overflows the building, it is trimmed off. Eventually, the calculated spaces are assigned to the corresponding indices of (A_{space}). In the resulting configuration, an index can be occupied by multiple spaces, and/or some indices may remain empty. This conflict is resolved in the next two steps by using two permutation lists, Collision Precedence List (CPL) and Fill Precedence List (FPL), which maintain the precedence of each space in case of collision and empty voxels. The higher-ranking space has the priority to preserve the overlapping voxels or to annex the empty neighbouring voxels. The values of CPL and FPL are generated randomly for the first generation. Following generations: Recombination operators are used to generate and assign prisms to voxels, and modify the CPL and FPL.

Step 2: This step eliminates the collisions (overlap) between spaces, if any, by determining which space preserves those voxels for all voxels in Aspace, if there are multiple spaces assigned, then CPL is referred to. Only the highest-ranking space is allowed to keep the voxel island. Following,

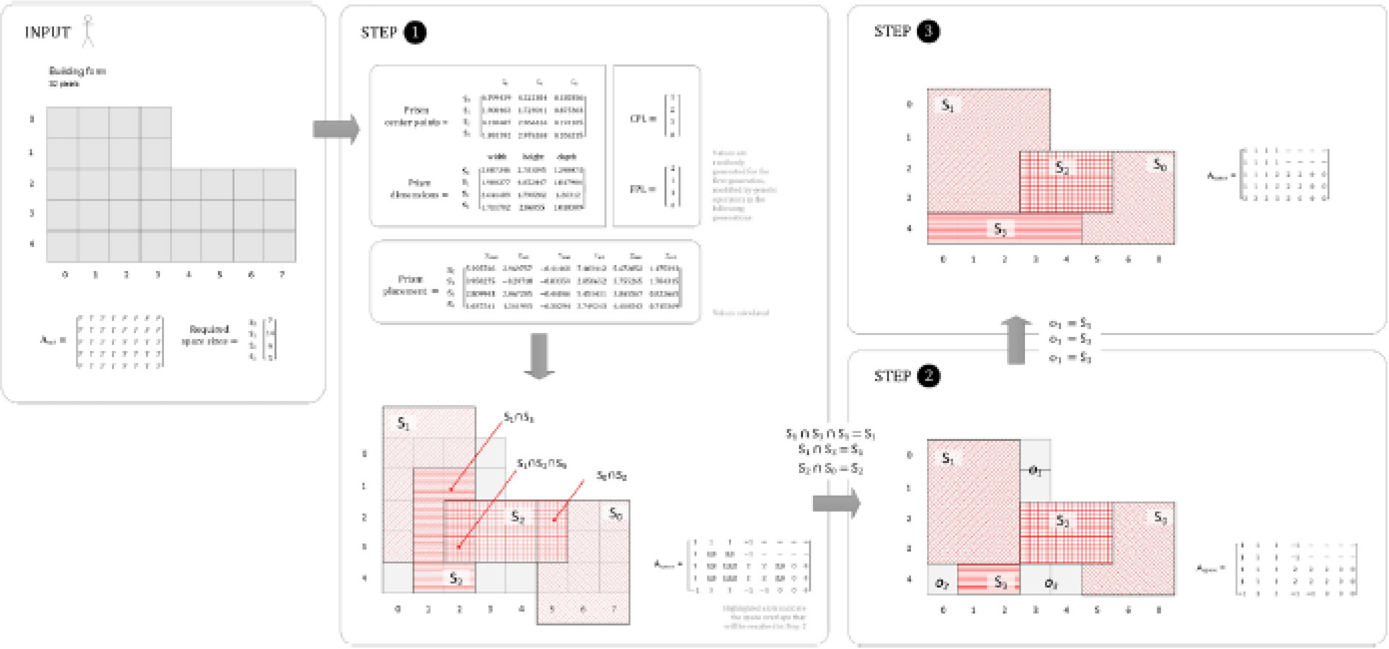


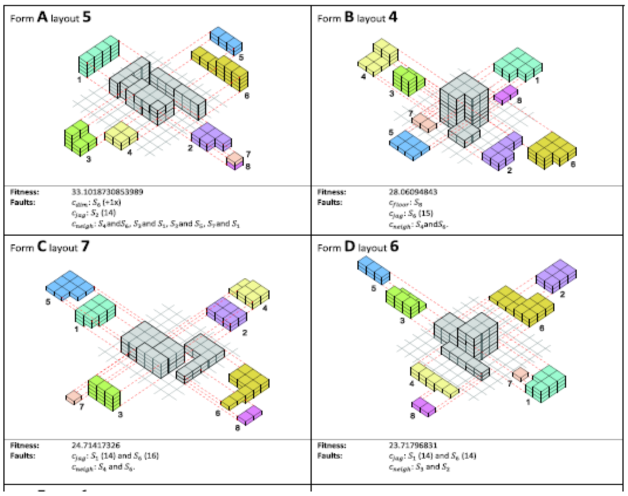
Figure 3.16: The three steps illustrated in the process of [(Dino, 2016)]

the remaining space(s) withdraw from this voxel island.

Step 3: determines which space extends itself towards the unoccupied areas, if any, remaining in the building First, the voxel islands (adjacent voxel groups) that don't have spaces assigned to them are detected with food fill algorithm. For each such island, all the neighbouring spaces that border these voxels are calculated. Amongst these, we eliminate the spaces that have already reached their required size (NRiv < NAiv). For all the remaining spaces, the highest ranking in FPL extends itself onto the island. As a result, all the empty voxels are occupied by the neighbouring spaces.
In these steps P-LCH generates spaces which do no have empty cells or overlaps but this does not guarantee that it meets all the configuration requirements. The Constraint Checker quantifies the fitness of a layout by means of penalizing constraints that evaluate spaces. Finally, weighted penalty values are aggregated into a single fitness function that is minimized towards zero. The eventual objective function can be formulated as below, where f(L) is the function to be minimized for a given layout L, and w is the user-defined penalty weights.

$$
\begin{aligned}
minimise f(L) = \sum_{n=1}^{i=1} (W_{size}C_{size} + W_{dim}C_{dim} \\
+W_{compact}C_{compact} + W_{jag}C_{jag} + W_{convex}C_{convex} \\
+W_{facade}C_{facade} + W_{floor}C_{floor}) \\
+ \sum_{A_{topo.length}}^{j=1} (W_{neigh}C_{neigh} + W_{sep}C_{sep})
\end{aligned}
\tag{3.1}
$$

In the subsequent steps EASE utilizes genetic optimization by using various operations for elitism, crossover, mutation and repair to generate an optimal solution.



Fitness:	33.1018730853989	Fitness:	28.06094843
Faults:	c_{dim}: S_6 (+1x) c_{jag}: S_2 (14) c_{neigh}: S_4andS_6, $S_3$and S_1, $S_3$and S_5, $S_7$and S_1	Faults:	c_{floor}: S_8 c_{jag}: S_6 (15) c_{neigh}: S_4andS_6.
Fitness:	24.71417326	Fitness:	23.71796831
Faults:	c_{jag}: S_1 (14) and S_6 (16) c_{neigh}: S_4 and S_6.	Faults:	c_{jag}: S_1 (14) and S_6 (14) c_{neigh}: S_3 and S_2

Figure 3.17: The optimization results from the work of [(Dino, 2016)]

Conclusions: Based on the references mentioned above the following conclusions can be made regarding the implementation of an Agent based modelling approach for a cell occupying method of generative space layout.

1.Performing simulations on the generated variants is expensive in terms of time and defeats the purpose of rapid generation of design prototypes. The simulations are also not as accurate in the early design stage in comparison to the final design.

2.Topology, Geometry and other architectural performance indicators like environmental performance need to be considered together for a comprehensive solution for the complexity of the problem.

3.The grouping of spaces into higher level (Typological) and lower level (Individual) is important before the cell assignment process.

4.Generation of curvilinear spaces is not easily possible with this method.

5.Precedence lists of the various spaces and agents are important in solving complexity and fuzziness since they make the decision-making process simpler

6.Evolutionary solvers are a good way of generating options but they take time and simplifications and pre-resolution of certain constraints go a long way in reducing the optimization time.

3.2 Participatory Design:

One of the main question which the thesis tries to address is the inclusion of stakeholders participation in the design process.To understand the role and scope of participation of the various stakeholders in the whole design process case studies about the already implemented participatory design projects were undertaken. This is followed by a small exploration in serious games or gamification of design as a means of participation.

The projects based on the open building concept are considered as part of case studies since the approach inherently is a participatory one and is in alignment with the intention of the project. Open buildings make a distinction between support and infill. The support represent the most permanent parts of the building like the structure and can be seen as a bookcase. The infill represents the adaptable part of the building or in other words the books. [(Habraken, 1961)]

3.2.1 Case study Molenvilet project:

The Molenvilet project in the city of Papendrecht in the Netherlands designed by Frans van der Werf is a classical example of the open building concept.The buildings are designed as a standard set of parallel columns and standard floors and roofs consisting of three floors and an attic. The buildings have been formed along four courtyards and are kept as empty shells with only the structure. The land parcellation and the number of units were done in consultation with the housing corporation.

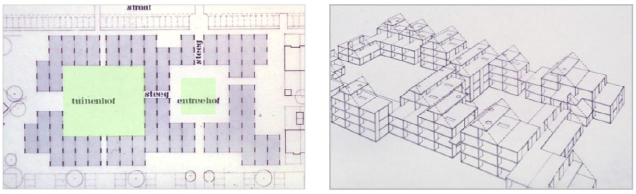


Figure 3.18: Plans from the molenvilet project [(*Molenvilet project*, n.d.)]

Once the constructions of the structure of the building started the meetings between the future residents and the infill contractor were scheduled. An empty copy of the building was kept for the discussions with the users and accompanying facade frames were added. In the first meeting the functional requirements and the required spaces for each family member with respect to their age, hobbies and preferences were discussed. After two weeks based on the first meeting drawings and details were made which where finalized with the users.In the second meeting the layout of the facade was also discussed and finalized based on the light and ventilation requirements of the users and they were also given a choice to choose the colours of the facade elements from a set of 8 different colours chosen by the architect. The resulting project became a classical example of mass customization based on user preferences.

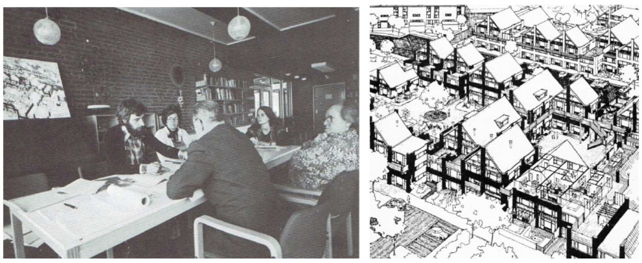


Figure 3.19: Discussion sessions with the users and infill contractors [(*Molenvilet project*, n.d.)]

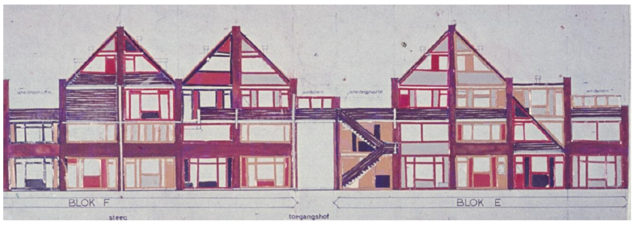


Figure 3.20: Facade sketches after the discussions with the users [(*Molenvilet project*, n.d.)]

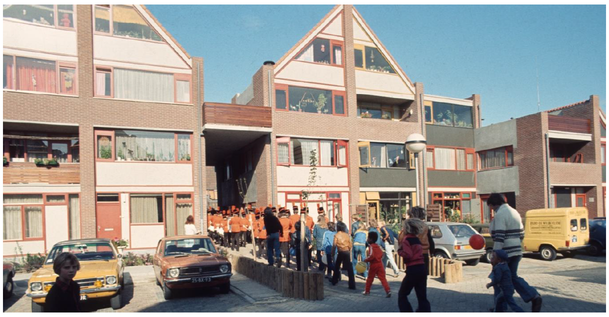


Figure 3.21: Project photo after construction [(*Molenvilet project*, n.d.)]

3.2.2 Case study Superlofts:

Superlofts is a flexible design and development framework and is developed by Marc Koehler Architects. Superlofts offers its members the freedom to customise or design and self-build their homes from scratch incorporating any hybrid function, and co-create shared spaces to build a global co-living community [(Superlofts, n.d.)].
The idea of super lofts is to adapt a open and circular framework where the lifecycles of the structure and the infill layers are independant of each other like the facades can be changed/modified after every 25 years, the HVAC installations after 10 years and the interiors after every 5.This can lead to less material wastage and efficient use of resources.

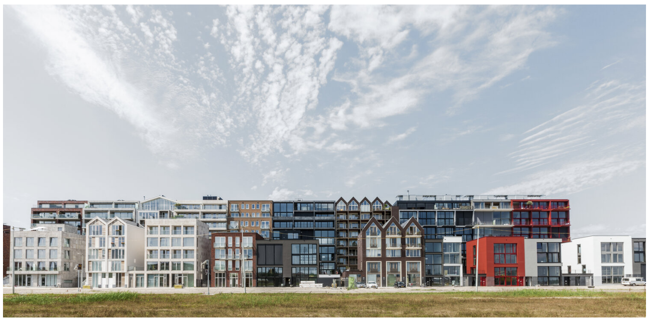


Figure 3.22: Superlofts project image [(Superlofts, n.d.)]

The base building comprises a prefabricated modular concrete framework composed of five to six meter high modules which forms the structure part of the building.The infill elements can be developed as per the needs of the owners requirements. Pre-fabricated walls and floor form one unit which are then stacked on top of one another.The project also offers it s users to grow their spaces gradually so they don't have to invest all their money at the same time and also there is a possibility of making the infill of the lofts independently of the main contractor.

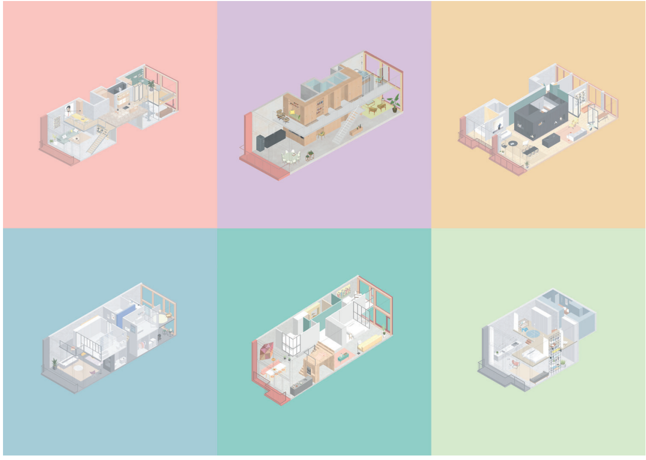


Figure 3.23: Configuration options for the infill [(Superlofts, n.d.)]

The facade system is a prefabricated aluminum modular building system that is easy to adapt to the specific demands of each user/owner. It has the intention of energy saving, natural ventilation and sun shading systems and rainwater drainage. There are several configurator tools which are under development for these projects as well like the site planning tool the house customization tool for the infill etc.which show close resemblance to the thesis project and also offers a validation to justify the need of such a tool.

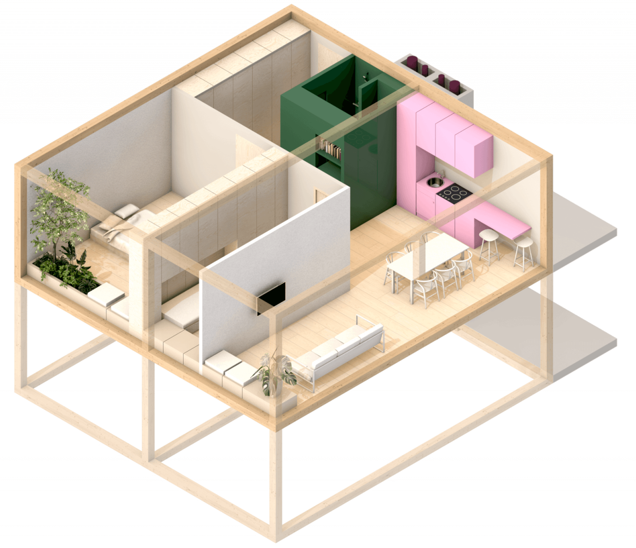


Figure 3.24: Outputs of the homemaker configurator of the superlofts project [(Superlofts, n.d.)]

3.2.3 Case study SketchBlock:

The Sketchblock project is also an example of the open residential building project designed by ANA architects for the 4Winden foundation members on Westlandgracht. In Sketchblock the position size and the layout of the dwellings can be customised for each residents needs. There is also a possibility of combining houses which are next to each other or above one another. The equipment of the home, outdoor areas and communal facilities have also been further developed by the residents.

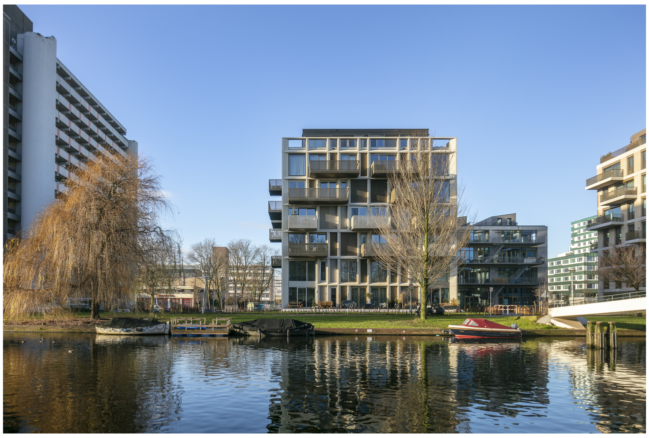


Figure 3.25: View of the sketchbook project [(architecten, n.d.)]

The process involved in this project was similar to the molenvilet project [3.2.1]. The meetings between the residents and the architects were arranged to design the layout of the apartments and finalize the detailing of the projects. Each house layout was custom designed by the architects.This in evidently lead to the creation of a catalogue of plans for each house. The architect had also clearly defined the rules for customization beforehand with the residents so that the design character and integrity of the building is maintained.

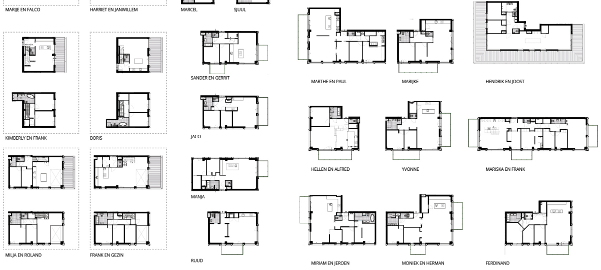


Figure 3.26: Catalogue of plans for the Sketchbook project [(architecten, n.d.)]

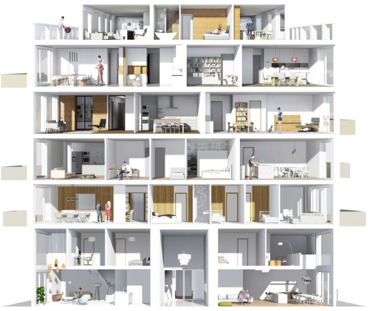


Figure 3.27: Cross-section for the Sketchbook project [(architecten, n.d.)]

The facade of the buildings was designed before so the layout of the apartments had to match the layout of the facade which was an added complexity in the project.The choice of having a balcony was left to the users but the size of it was fixed by the architect to maintain the visual appearance of the building.

3.2.4 Gamification as a strategy for participation:

Gamification or serious games have proven to be a good method for collective decision making and user participation. A serious game is defined by [(Göbel & Wiemeyer, 2016)] as a computer software which is designed not just for the purpose of playing but has other serious aspects associated with it like education, decision making and problem solving.
[(Sanchez, 2021)] sums up the medium of video games as a way of systems thinking very accurately by stating that Video games are an interactive medium where players are able to engage with the production of form and systems thinking. They have shown great potential to advance an architectural agenda that attempts to democratize access to local fabrication and community development.

Figure 3.28: Cities Skyline a city builder game which simulates real world systems for city building based on the various urban requirements [(*Cities Skylines - Cities: Skylines*, 2020)]

Games such as Minecraft that have a network of nearly 75 million players have the closest resemblance to what we could consider a social platform for spatial content. Games have been able to establish a two-way dialogue between users and developers, actively participating in the production of a network via forums, polls and streams among other forms of digital content.

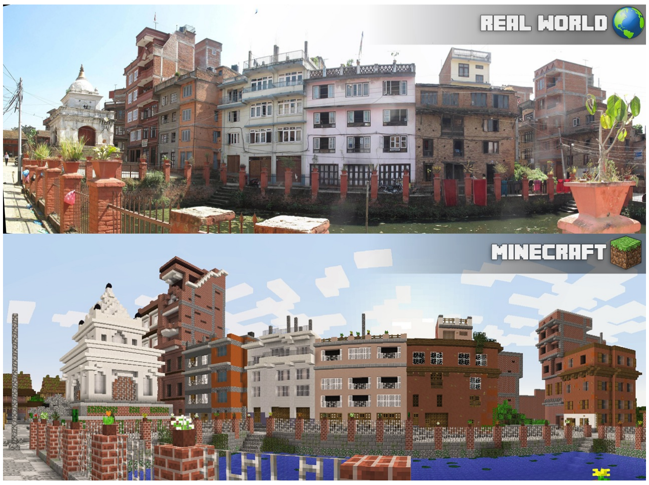


Figure 3.29: Minecraft used as a tool by UN habitat to allow public to participate in Urban design of a district. This exhibits how close to reality the gaming platforms today have developed [(blockbyblock, n.d.)]

Games also have another quality which is commonly referred to as modding. This makes it ideal to serve as a development platform for combinatorial design seen in games like Minecraft and cities skyline where it is seen that with robust in game educational protocols have led players to reach advanced game stages.

3.2.5 Example of a Participatory design game:

The relationship between computer games and Architecture has been studied by a lot of researchers around the world and many smaller design games have been developed.However there has not been any significant development in a comprehensive design game which has the ability to solve real world architectural problems and generate deployable solutions. One of the really good games that has been released which focuses on systems level thinking and combinatorial design is the game of Block'hood developed by Jose Sanchez [(*Block'hood*, n.d.)].

According to the official description[(*Block'hood*, n.d.)], Block'hood is a city building simulator video game that focuses on ideas of ecology, interdependence and decay. The game invites players to envision a neighbourhood, by building structures out of a catalogue of 200+ blocks. The player is challenged to maintain an ecological balance as each block placed will consume and produce resources of different kinds. Blocks that are not provided of their required input, will slowly decay and deteriorate to a point of collapse. Player creations will attract inhabitants, both humans and animals, that will populate your neighbourhood. It is the hands of the player to provide a positive environment for inhabitants to prosper.

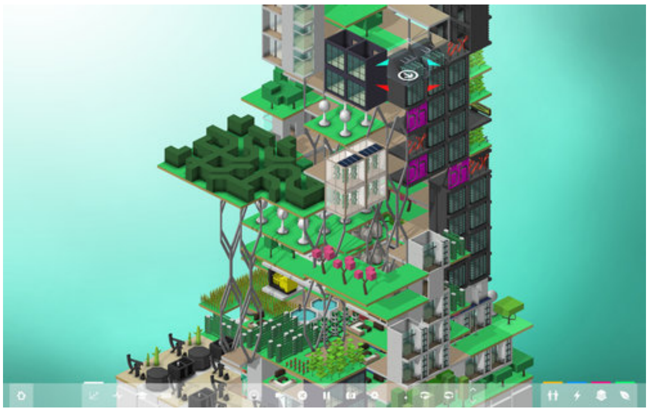
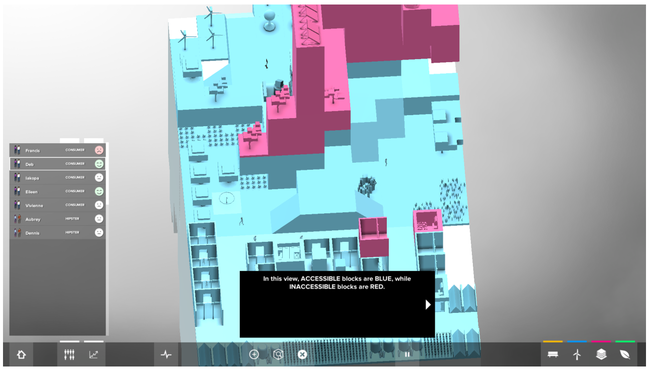


Figure 3.30: Example of a neighbourhood built on Block'hood game [(*Block'hood*, n.d.)]

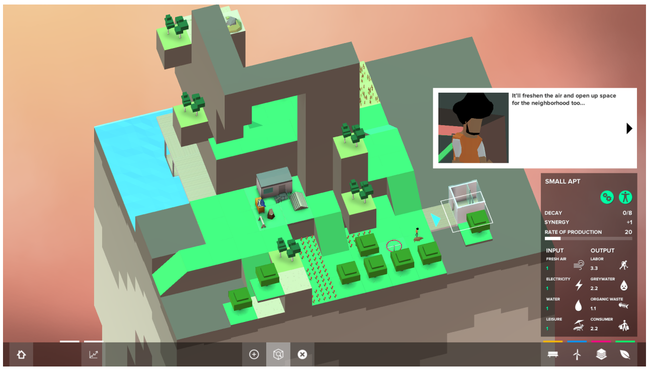
The game is both an educational and research initiative exploring the connection between games and architecture, contributing to a form of a digital infrastructure for the ecological and systems thinking that is necessary in analysing the socio-economical problems associated with their local communities in contemporary Urbanism. Platforms are utilized as a dashboard for combinatorial design aiming for the production of literacy and the development of digital infrastructure for participation and self-provisioning. At the core of these initiatives is an interest in generating architectural principles and designs that engage the world through a scope of resource management, systems thinking and ecological interdependence.



In this view, ACCESSIBLE blocks are BLUE, while INACCESSIBLE blocks are RED.

Figure 3.31: Accesibility Filter view on the gameplay highlighting the blocks that do not have an acess in pink [(*Block'hood*, n.d.)]

The challenge is to effectively create a platform that is aimed at the construction of the common knowledge and repositories of architecture alternatives.



It'll freshen the air and open up space for the neighborhood too...

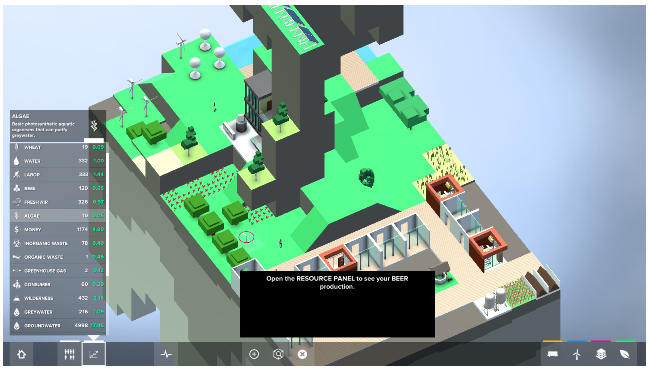
SMALL APT

DECAY	0/8
SYNERGY	+1
RATE OF PRODUCTION	20

INPUT		OUTPUT	
FRESH AIR	1	LABOR	3.3
ELECTRICITY	1	GREYWATER	2.2
WATER	1	ORGANIC WASTE	1.1
LEISURE	1	CONSUMER	2.2

Figure 3.32: The tab on the right shows the input and the output of the asset block in this case an apartment and also shows the rate of production [(*Block'hood*, n.d.)]

Important features found in the game:
1.The game can be played in a story mode/challenge mode/sandbox mode. The story mode gives an engaging and a comprehensive overview of how the game is to be played.
2.Through a narrative the player is guided through all the concepts of the game which makes it easier for everyone to understand the core concepts.
3.The controls are simple the user can pan rotate and press the cells to create or remove assets in the neighbourhoods. The assets themselves are modular blocks which can only be created, removed or rotated.
4.The connectivity requirement for each asset is displayed and appropriate connectivity elements need to be deployed for the assets to be productive and useful for the neighbourhood.Each cell in the grid has a certain number of resources available in it to harvest which is shown on an ordinal scale. The player can investigate the on this basis the development (asset block) which can be proposed on the block.



ALGAE
Basic photosynthetic aquatic organisms that can purify greywater.

WHEAT	19	0.09
WATER	332	1.00
LABOR	333	1.44
BEES	129	0.65
FRESH AIR	326	0.97
ALGAE	10	0.05
MONEY	1174	4.90
INORGANIC WASTE	78	0.40
ORGANIC WASTE	1	0.46
GREENHOUSE GAS	2	0.12
CONSUMER	60	0.29
WILDERNESS	432	2.16
GREYWATER	216	1.09
GROUNDWATER	4998	17.45

Open the RESOURCE PANEL to see your BEER production.

Figure 3.33: The bar on the left bottom shows the trends of the resources in the neighbourhood [(*Block'hood*, n.d.)]

5.The overall requirement of the Neighborhood can be seen in the resource panel and there is also a visualization filter available to visualize a certain type of data like decay, access, production, land value etc.
6.There is also a panel where the likes and dislikes of the people in the neighborhood can be seen. Based on this panel the game possesses challenges for development of the neighborhood.
7.Different agents are unlocked as the game progresses like animals and people and if certain combinations of assets are deployed in the game then it produces conducive environment for certain agents and the neighborhood is more productive.

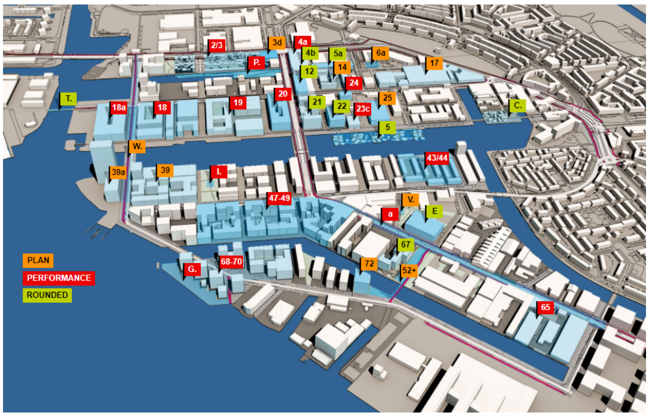


Figure 3.34: 3D layout of Buiksloterham development status [(*Block'hood*, n.d.)]

3.2.6 Conclusion:

The following conclusions were made after finishing the literature studies.
1.The problem of spatial configuration needs to be separated into various scales due to the variation in the importance of the preference of the stakeholders at each stage of the design process.
2.Clear rules need to be formulated for participation and the degree of participation needs to be determined before recording the preferences and choices of the participants.
3.The choices of one user should not affect the quality of spaces for the others or the whole building.
4.The consequences of the decisions taken should be visible to the participants and there should be a feedback from the tool pinpointing the areas which can be improved.
5.The principles of Open building movement like separation of structure and infill should be implemented in the tool for achieving its goals.
6.Gamification could be a strategy which can be implemented for the user interaction with the tool and to generate a performance feedback.

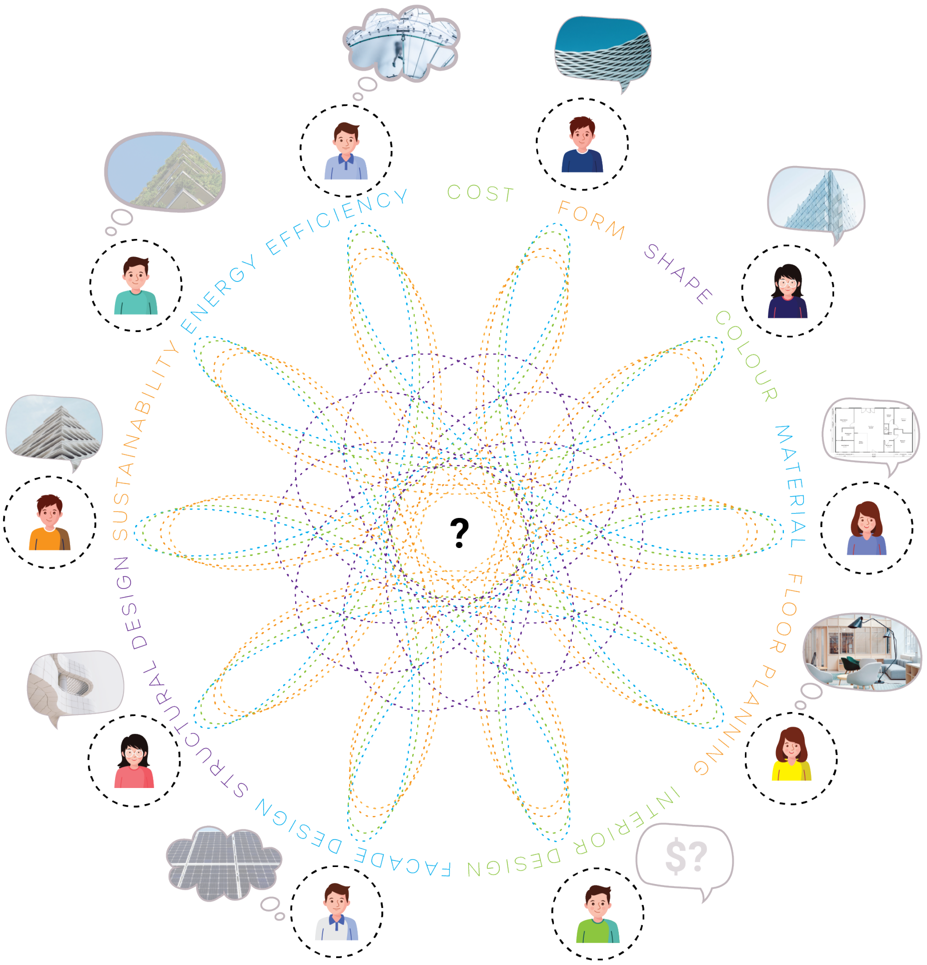


Figure 3.35: Diagram showing the multi-actor complexity in reaching to a consensus about the design goals

4 Problem formulation:

The problem of 3D layout configuration, when regarded in its full sophistication, is a classical example of a wicked problem that is at the same time a complex combinatorial problem which can be classified as a NP-hard problem [(*NP hard and NP complete problem*, 2021)] , and arguably the most difficult problem of computational design for which the level of 'control' provided by the claimed methods in the literature is limited.This is primarily because of the human/physical complexities involved in the problem.Reaching a consensus on the design goals itself between the actors themselves becomes a difficult task due to the multi-actor complexities.

A methodical approach toward solving the problem by breaking it down into a set of smaller problems and subsequent actions helps in understanding and addressing the underlying complexities like multi-dimensional, multi-criteria, multi-actor, and multi-value complexities holistically. In this thesis an attempt is made to systematically separate the process of 3D layout configuration as a series of steps with local goals and with a certain set of assumptions and simplifications to achieve the global spatial planning goals. In the following section each level of problem from the biggest problem (Urban level massing) to the smallest problem of configuring a floor plan will be elaborated further with the stakeholders involvement in them. The detailed mathematical formulation and the method used will be elaborated further in the report.

4.1 Massing problem:

The highest nesting level of problem is the massing problem.As a first step toward solving the problem a volumetric representation of the building needs to be generated. In architectural practise it is common to sketch out various design options for the project on a conceptual level using representative 3d blocks having a low resolution of details. In this stage of the design process Architects, Developers, Governing authorities like Municipal corporations are the main stakeholders involved. The goals for the Architects is develop their concepts on the brief provided by the Developers. The goal of Developers is to maximise the profit or the developable area on the site and the goal of the

Governing bodies is to assess the impact of the development on the immediate surroundings and the city and to safeguard the rights of the citizens of the block.In this thesis the massing problem is solved by means of a combination of manual design and digital validation. The process taken in the massing problem is a series of systematic steps and decisions to generate and validate the design options.Generation of the massing blocks at each stage is done manually in a 3D modelling environment of Rhinoceros and the validation part is done partly on Grasshopper for simulations and python for processing the simulation data and creating a validation framework. The main reason for this approach is not to restrict the freedom of design involved in the initial phase of the design project but to generate feedback on the designs for a better development of the overall proposal.

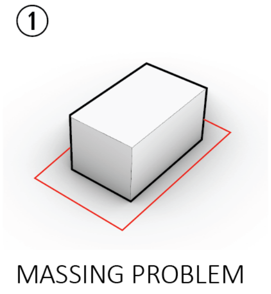


Figure 4.1: The massing problem diagram

4.2 Zoning problem:

In the selected massing option locating the appropriate zones is the main goal of the zoning problem. Based on the space program zones are defined and their requirements are also determined. The zoning requirements have two main characteristics. The first one being the closeness relation with other zones and the second one being the appropriateness of the location of the zone according to the stakeholder's decision making. The stakeholders and their goals for the zoning problem are as follows:

The architect will make a framework for all the rest of the stakeholders to participate. The framework the decisions available to each stakeholder will be mentioned and the feedback for of the stakeholders will also be taken for any additional requirements. Once the preferences and decisions of other stakeholders are recorded the architects evaluates the preferences of all the stakeholders and makes a combined decision model which will guide the zoning goals of the project. The performance of the zoning will be assessed on how close the zones are placed to the ideal location for the zone. Several methods were explored to solve the zoning problem among them the main one is a multi-agent system to generate the zoning clusters.

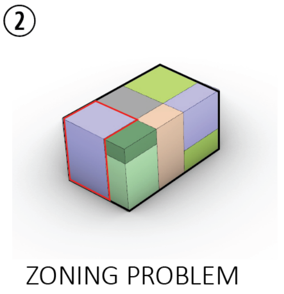


Figure 4.2: The Zoning problem diagram

4.3 Unit assignment problem:

In the selected zoning model of the building assigning the units is the main goal of this problem. Units can be defined as the higher level function or cluster of rooms defining a function. For example in a residential building an apartment is a unit of the space program. The first step before the assignment process is to generate the circulation shafts in the zoning model. The vertical circulation shaft containing stairwells elevators and building services ducting is first located by analysing the zoning model and the geometrical properties of the massing. The horizontal circulation elements can be added after this step or later after the unit assignment is done depending on the quality of circulation that the user picks. If for example the circulation is through the spaces then it can be assigned after the units are placed but if the spaces are along the circulation then the horizontal shafts have to be assigned after the vertical ones.

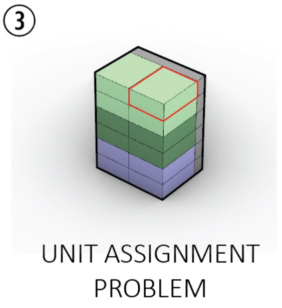


Figure 4.3: The Unit assignment problem diagram

The Architect and the Developer are the main stakeholders in the unit assignment goals. The main goal of the assigning is to generate good clean floor plans with optimal use of the floor area and a good access to daylight, while respecting the area bounds given by the developer for each unit.

4.4 Unit layout problem:

In the unit layout problem the main goal is to generate the internal room layout for the unit. The stakeholders involved in this step is the end user and the architect. The end user first records the preferences with respect to the number of rooms and their placement and the architect feeds it into the tool. The tool will generate multiple layout which fits the goals of the end user and the end user can approve the layout or change the preferences for an improved version. After a final layout is made the detailing aspect of the project is recorded. This problem of generating layouts according to user preferences without compromising the layouts of other users is quite tedious and time consuming as seen in the various case studies before in the literature review[3.2].An automated process for this which develops floor plan options based on the user preferences and complying with the rules set by the architect would be beneficial.

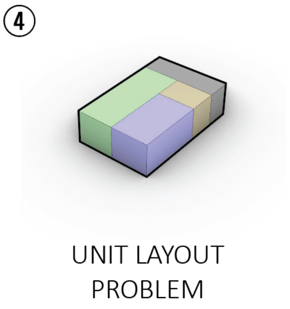


Figure 4.4: The Unit Layout problem diagram

4.5 Unit Detailing problem:

The Unit detailing problem is the next step after the Unit layout problem. The main goal of this step is to specify the infill details for the rooms which are defined in the unit layout stage. The end user along with the architect will decide on the materiality of the infill elements where the end user can pick elements from the catalogue defined by the architect. The tool at this stage will give output by the means of a dashboard with various indicators for the end-user to understand the financial, sustainability and spatial aspects of the choices made.looking at the output of the Dashboard the user can keep on modifying the choices till an desired outcome is generated. The catalogue for the infill layer will be developed by the Architect in collaboration with the Building Product manufacturers and Building contractors for maintaining accuracy in the details of it.

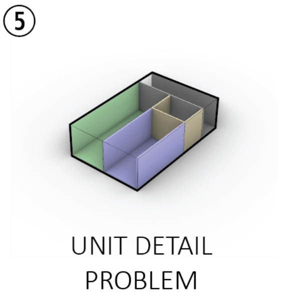
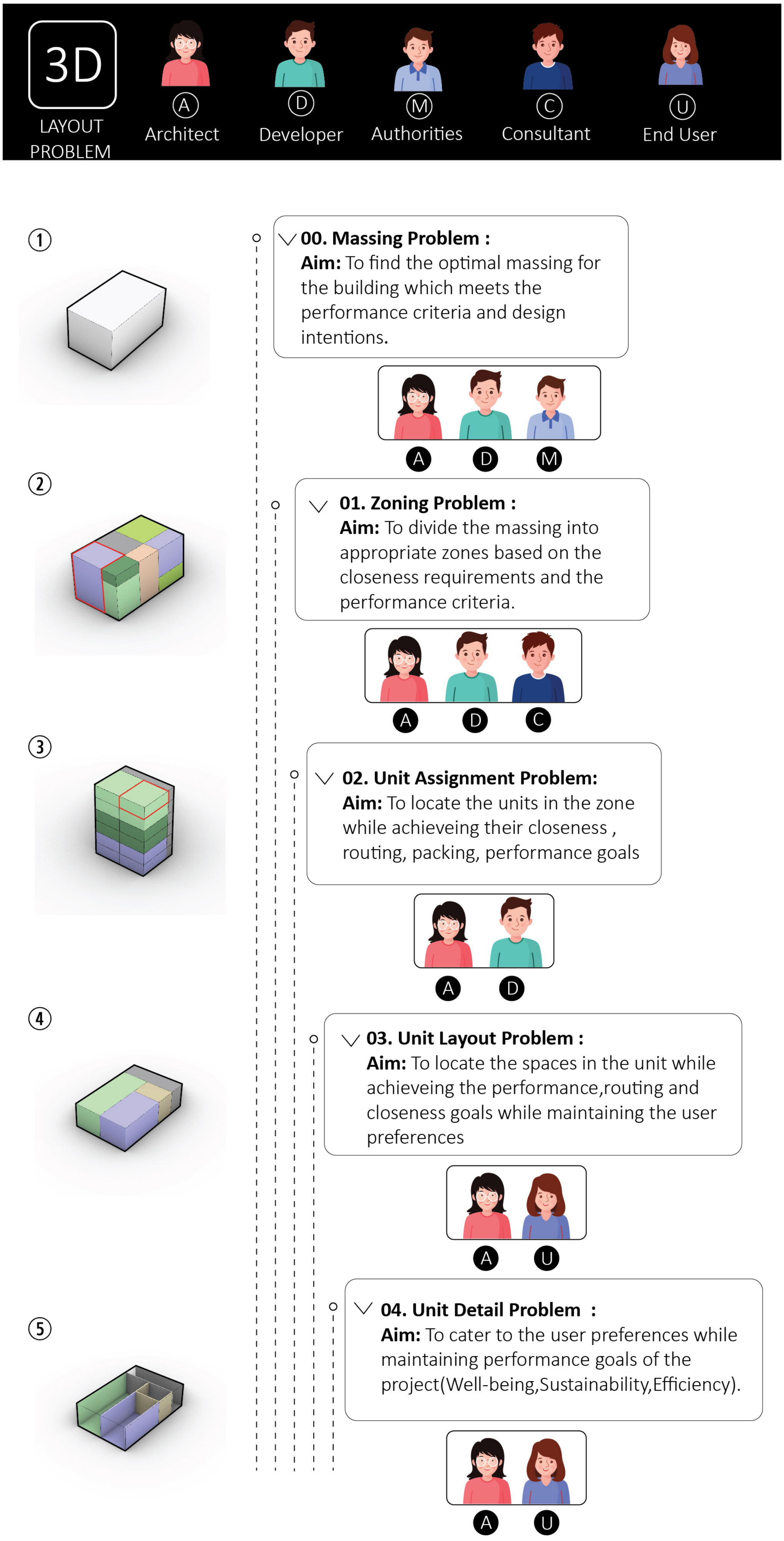


Figure 4.5: The Unit Detailing problem diagram

4.6 Limitations and Conclusions:

Solving the five problems as stated in the problem statement in their full complexity is a difficult and time-consuming task. Considering the time available for the graduation thesis only some problems are solved in detail and a framework or potential ways of solving the rest are elaborated. There are a few common elements involved in all the levels of the problem starting with a system that allows for the actors of the problem to make decisions and register their spatial choices. This is followed by a multi-criteria decision analysis system which generates the results about the desirability of a design option or a location based on the recorded results.

Finally there is a system which can act on the results of the decision making and grow a configuration based on it , resulting in collaborative configuration solutions. A best examples of integration of all the three elements can be seen in the resolution of the zoning problem further in the thesis.



00. Massing Problem :
Aim: To find the optimal massing for the building which meets the performance criteria and design intentions.

01. Zoning Problem :
Aim: To divide the massing into appropriate zones based on the closeness requirements and the performance criteria.

02. Unit Assignment Problem:
Aim: To locate the units in the zone while achieveing their closeness , routing, packing, performance goals

03. Unit Layout Problem :
Aim: To locate the spaces in the unit while achieveing the performance,routing and closeness goals while maintaining the user preferences

04. Unit Detail Problem :
Aim: To cater to the user preferences while maintaining performance goals of the project(Well-being,Sustainability,Efficiency).

Figure 4.6: Nesting levels of Architectural configuration problem

5 Case Development:

5.1 Case Introduction:

In order to test all the modules developed in the thesis a test case becomes essential. The test case for the project should be mixed used multi functional building on a site that has strong characteristics that will be influential for the design. The limits of the tool will truly be tested when there are a lot of spaces that need to be configured and they have contradictory requirements.

5.2 Selected Neighbourhood:

The site selection process started with looking at Urban centres in the Netherlands. Dense Cities provide a good context and a challenge for the testing process and choosing a Dutch city is ideal because of the good quality and the extent of the data which is available for them.
After doing an extensive search a site in the northern part of Amsterdam was selected. Buiksloterham is a neighbourhood in Amsterdam north which has an industrial past and is located very close to the old centre of Amsterdam just across the IJ river. Buiksloterham has a lot of empty plots and no monumental industries which provides a blank slate for experimentation of a new methodology of designing buildings as proposed in this thesis.

The municipality of Amsterdam has a clear plan focused on sustainability and circularity for the development of this post-industrial neighbourhood. In the action plan for a circular Buiksloterham the neighbourhood is envisioned as the test bed for the future of Amsterdam. The polluted lands of the neighbourhood aim to be the source of clean technologies and hub for closure of urban material cycles. The activities needed to close these local material flows can be used as a driver for local industry and the strengthening of local social networks. IT-based interventions can smartly connect local residents with one another and boost the efficiency of resource flows. Urban biodiversity and climate adaptation measures are conceived as a core strategy to bring long-term local resilience to the area. The development projects aim to be the beacon of change for the development of post-industrial areas around the world.

5.3 Vision for Circular Buiksloterham:

In the action plan to achieve the goals of a sustainable, smart and circular Buikesloterham the following things are on the agenda:
1.Designate Buiksloterham as an official experimental zone or Living Lab. A Living Lab status is necessary for establishing the overall character

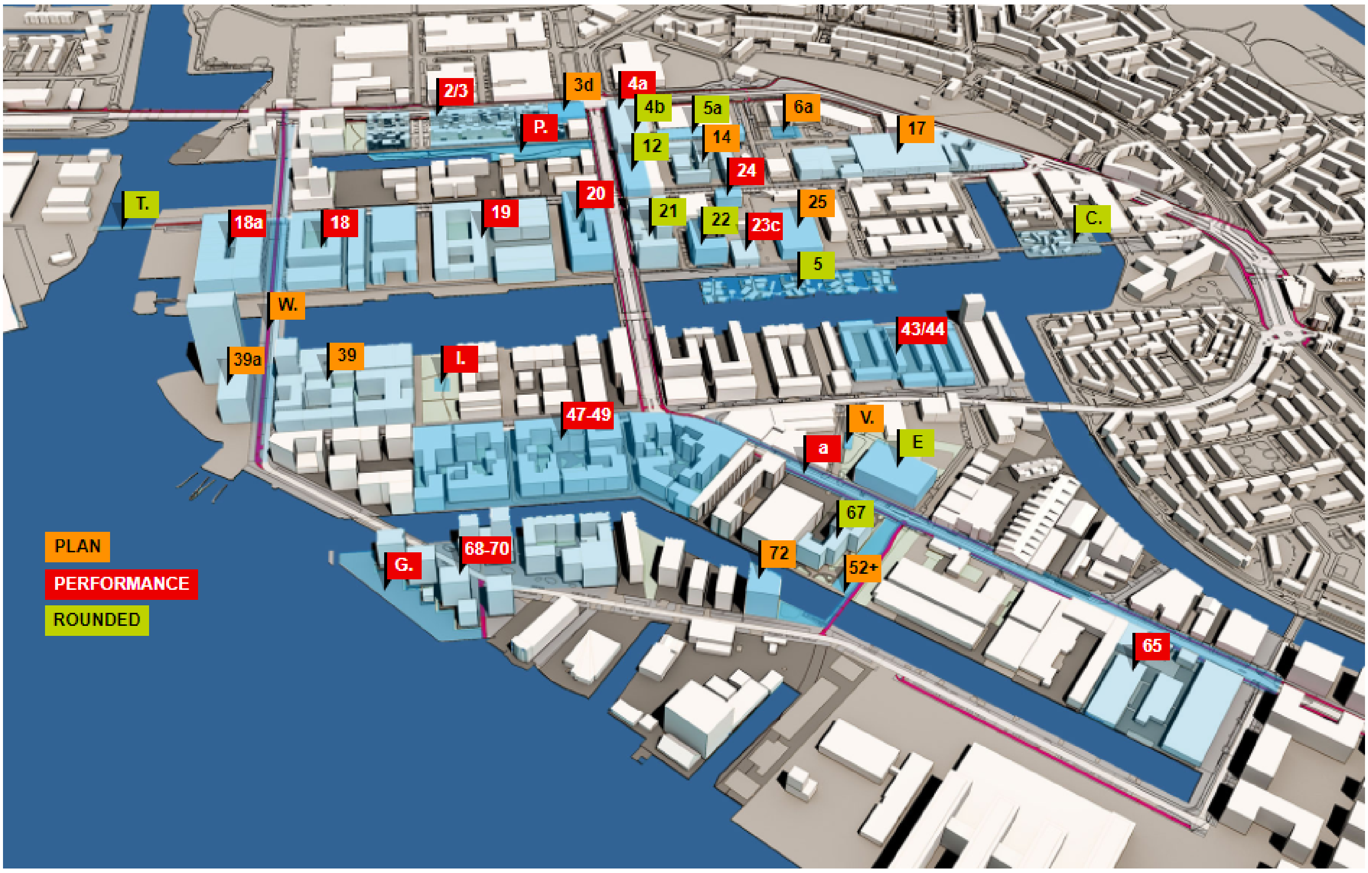


Figure 4.7: 3D layout of Buiksloterham development status [(*https://www.amsterdam.nl/projecten/buiksloterham/*, n.d.)]

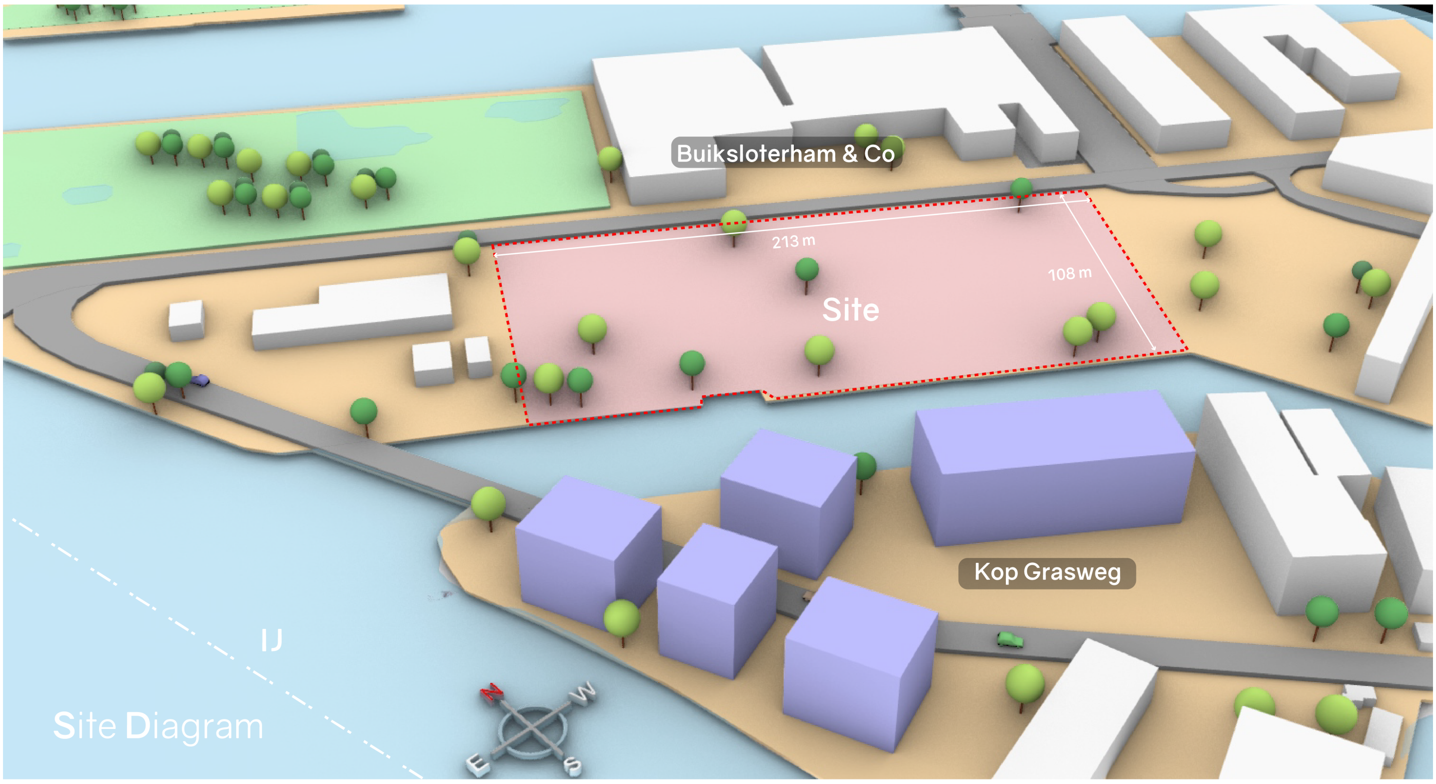


Figure 5.1: Nesting levels of Architectural configuration problem

of the neighbourhood as a place where new technologies and management approaches can be applied and learned from. The future Buiksloterham is envisioned as a neighbourhood which is run on the makers economy.

2.Develop an inclusive governance and management structure for Buiksloterham which involves more participation opportunities for the residents in managing the neighbourhood.

3.Creation of new incentive structures and financial vehicles. The transition plan to a Circular Buiksloterham requires sufficient capital for investment and appropriate incentive structures (such as tax or credit schemes) that will provide guidance and enforcement of key directives.

4.Build capacity for urban sensing and open data. Urban sensing and open data infra-structure are critical for monitoring progress towards the goals, enforcing key directives, and for purposes of research and communication.

5.Energy efficiency and Circularity of all new construction must be strictly implemented and checked.

6.Renewable energy and rainwater must be harvested to its full capacity. The vision which the municipality is envisioning for the neighbourhood makes it a perfect candidate for the testing of the platform.

5.4 Selection of specific Site in the neighbourhood:

After inspecting the neighbourhood development plan for a mixed used proposal the site at Buiksloterham and Co. was selected due to its interesting location in the neighbourhood and due to its regular shape. A mixed use living and working buildings are planned in this part of the neighbourhood. The location faces the river banks of the IJ.According to the current plan by the municipal corporation Buiksloterham and Co. will have a total of around 520 homes. The development type is varied from self-build plots and market plots (CPO) to owner-occupied apartments and rental homes. There is an attempt to integrate Living and working. Office and work spaces will be integrated with the residential spaces in the buildings. There is also a possibility of self development of houses by the owners themselves.

Buiksloterham and Co. is planned to be the first circular district in Amsterdam. In this district, new ways of living and working together are sought, with sustainability and reuse as the starting point. The reuse of materials, energy and water is paramount and car sharing is also encouraged.

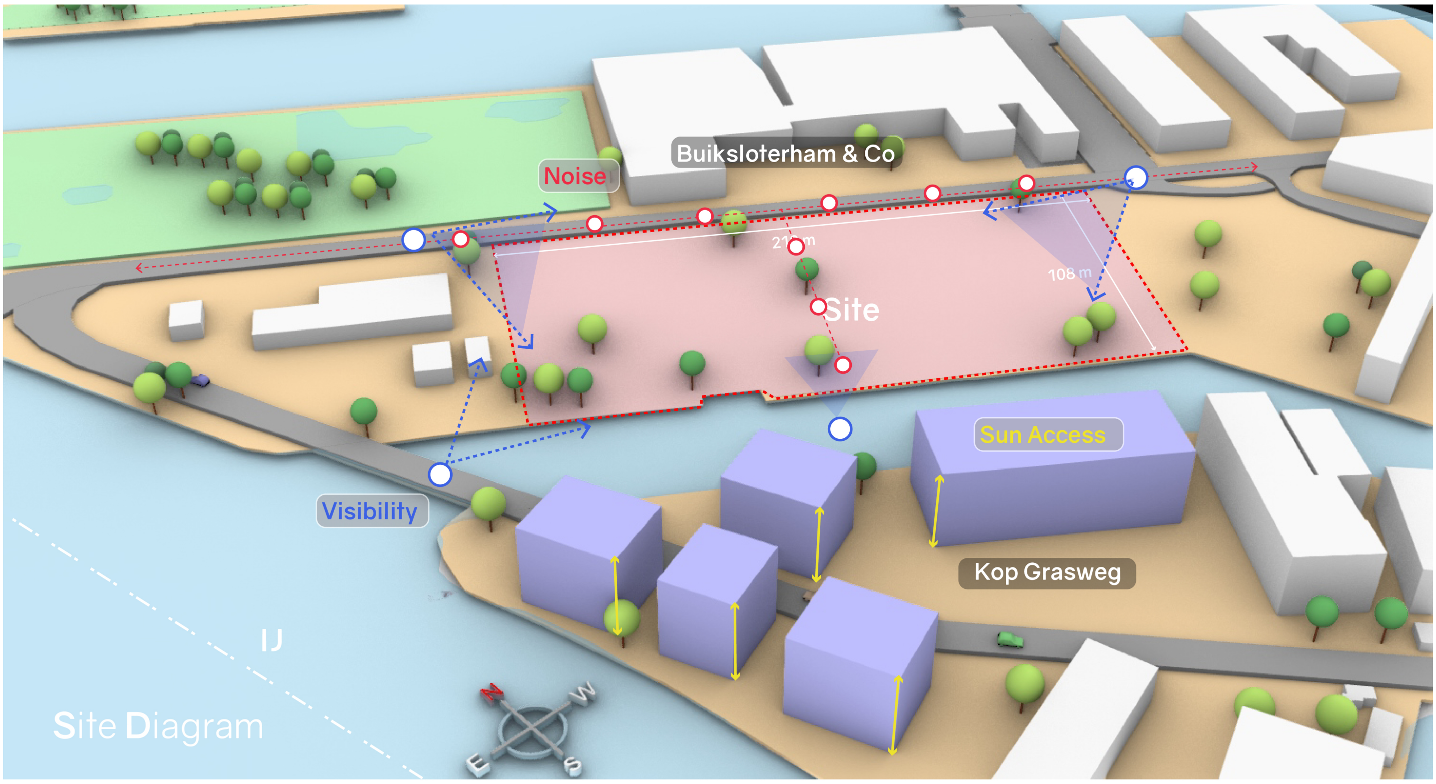
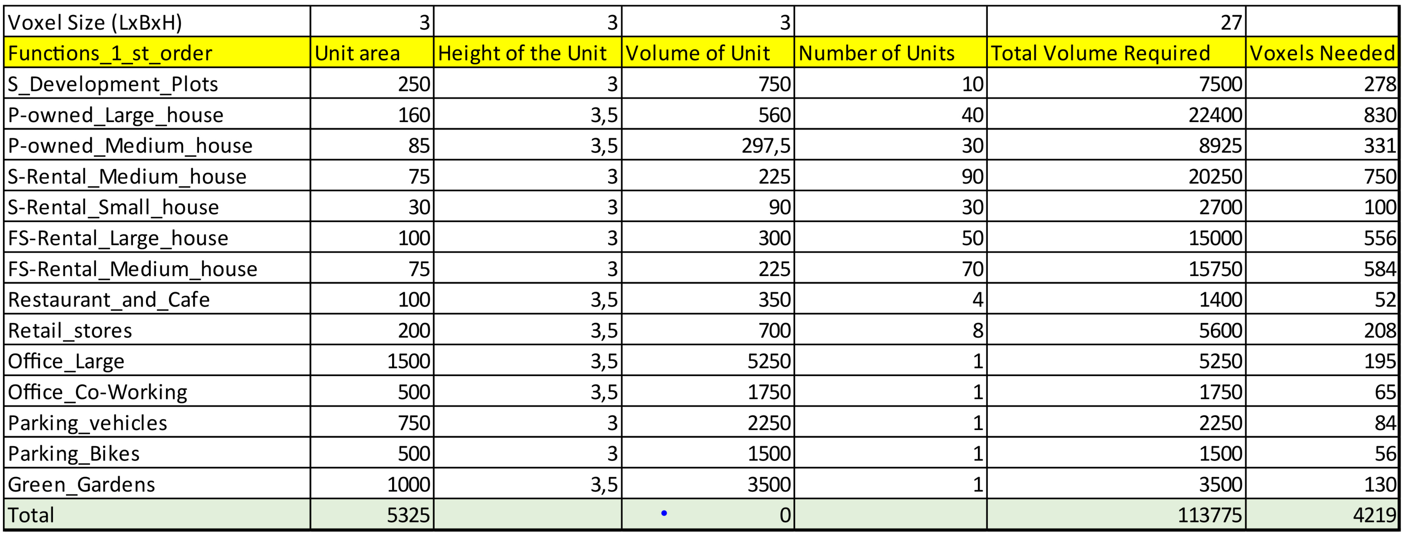


Figure 5.2: Site analysis

5.5 Site analysis:

The following factors can be considered while generating the design parameters for the project on the site. There is a road which divides the plot into two parts and is the main access road for the plot. The vehicular noise would be considerable from this part A pedestrian bridge is envisioned on the canal located on the north side connecting the Kop Grasweg a mixed used neighbourhood and the site. The location of commercial space could be driven by this connectivity. The development of Kop Grasweg should also be considered since it can block the sunlight from the South side of the plot. A large boundary of the site has visibility to the bank of IJ and could act as an attractor point for the apartments and offices. The vision for the neighbourhood is a participatory one so there is a strong chance of a healthy participation from all the stakeholders.

The space program is defined based on the preliminary calculations based on the site area and the literature studies about the existing proposal. The total buildable area will be dependant on the solution proposed and the space program will adapt to the solution maintaining the ratios of distribution of areas into the various functions defined in the space program. The functions for the space program are mixed use residential and commercial ones.



Voxel Size (LxBxH)	3	3	3		27	
Functions_1_st_order	Unit area	Height of the Unit	Volume of Unit	Number of Units	Total Volume Required	Voxels Needed
S_Development_Plots	250	3	750	10	7500	278
P-owned_Large_house	160	3,5	560	40	22400	830
P-owned_Medium_house	85	3,5	297,5	30	8925	331
S-Rental_Medium_house	75	3	225	90	20250	750
S-Rental_Small_house	30	3	90	30	2700	100
FS-Rental_Large_house	100	3	300	50	15000	556
FS-Rental_Medium_house	75	3	225	70	15750	584
Restaurant_and_Cafe	100	3,5	350	4	1400	52
Retail_stores	200	3,5	700	8	5600	208
Office_Large	1500	3,5	5250	1	5250	195
Office_Co-Working	500	3,5	1750	1	1750	65
Parking_vehicles	750	3	2250	1	2250	84
Parking_Bikes	500	3	1500	1	1500	56
Green_Gardens	1000	3,5	3500	1	3500	130
Total	5325		0		113775	4219

Figure 5.3: Space program

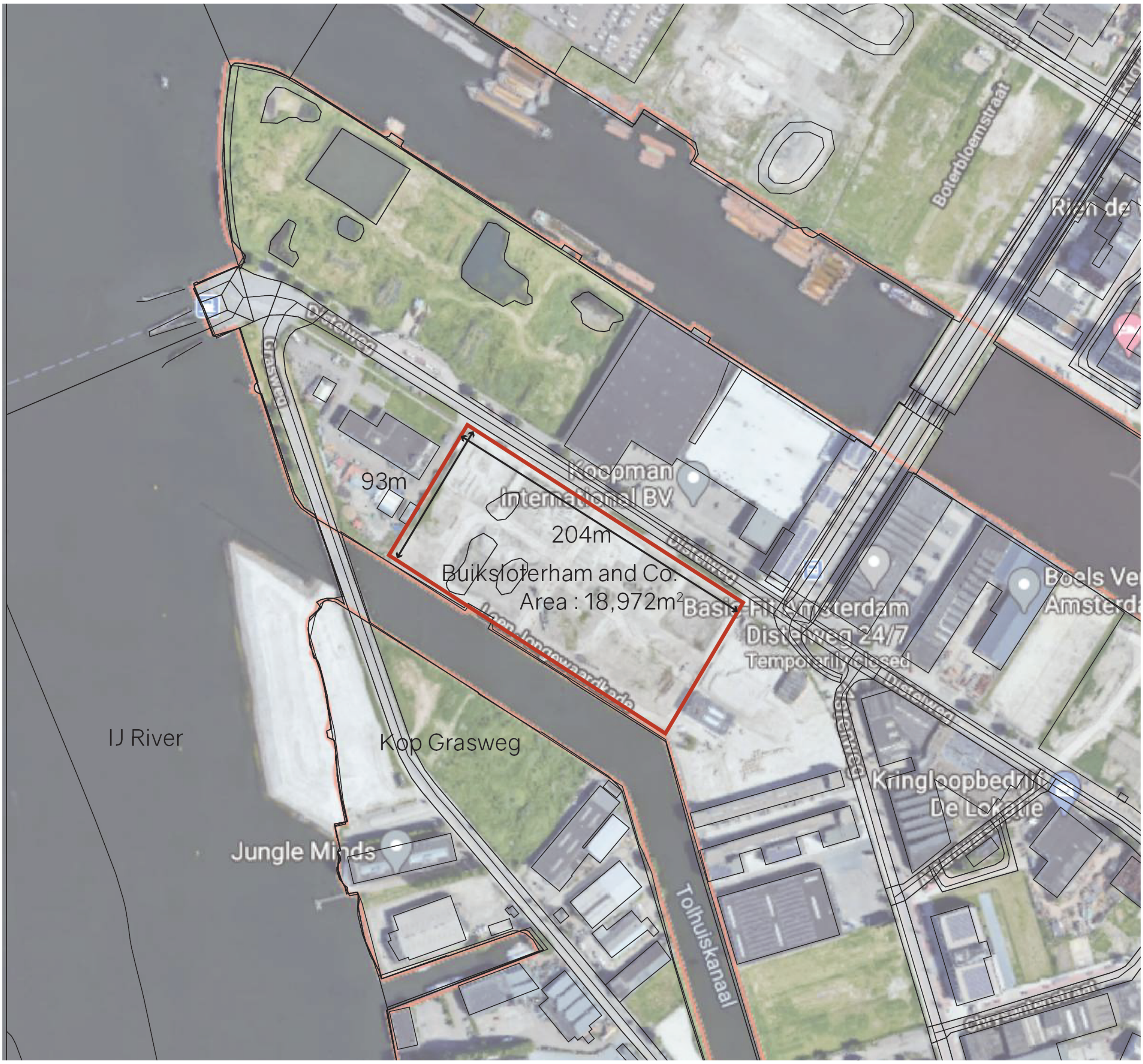
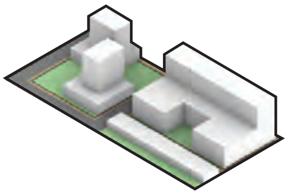
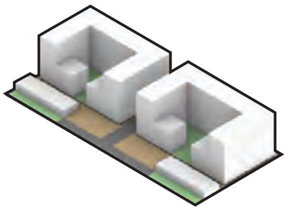
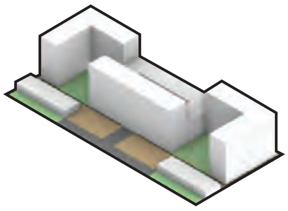


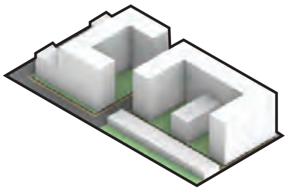
Figure 5.4: Satellite imagery of the Site

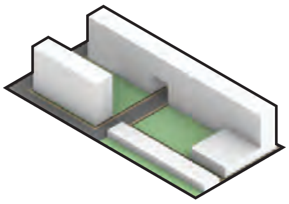
MASSING PROBLEM

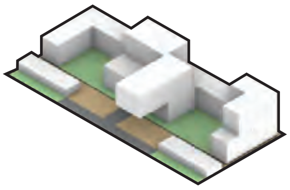


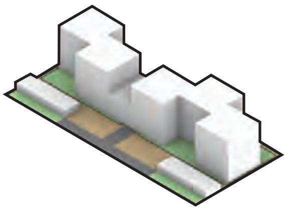


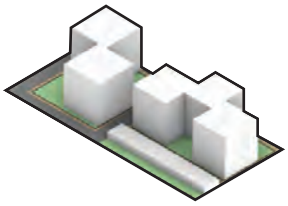


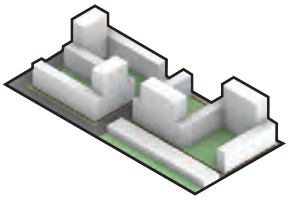


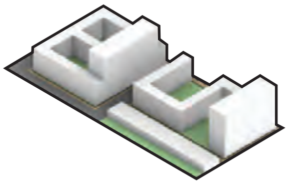


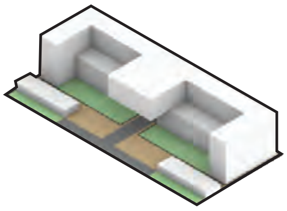


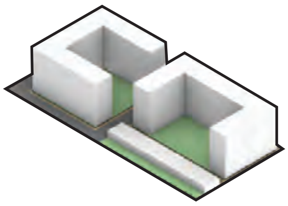


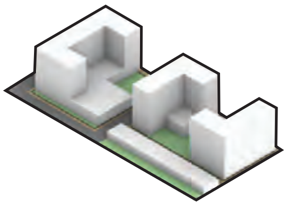


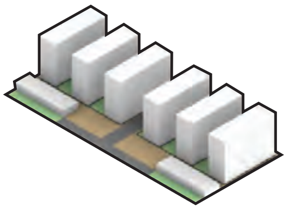


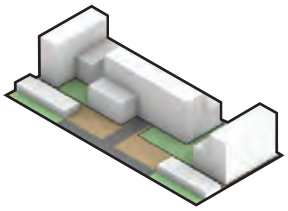


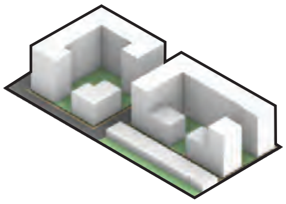












GEN-ARCH

6 Massing problem

The problem of finding the appropriate volumetric design of the building is the highest level of the space layout problem.In this stage the impact of placing volumes representing the built mass on the site is studied with respect to its effect on the surrounding buildings as well as its own performance in achieving the urban level design goals. Developing generative massing models is a huge problem and can be a project in itself so in this thesis a co-design process is taken for developing the massing options.

6.0.1 Aim:

To generate various massing variants for the site and select a variant which will maximise the site utilization without compromising on the quality of the spaces as defined by the architect.

6.0.2 Process:

The massing process described further is particular to the case at hand at Buiksloterham and Co. as described in [5].The process taken in this step is a combination of manual design and computational analysis and validation. The design ideas are sketched manually and the performance indicators for the sketches are programmed in a computational framework where simulations and calculations are done to quantify the indicators. Then weights are given to each criteria based on the designers validation preferences and a Multi-criteria decision analysis is done to rank and select the design variant from the sketches.This process also takes place in a methodical and ordered way in several steps as described below.

Looking at the space program [5.3] two distinctions can be made regarding the massing requirements of the program.The mass encompassing the volume of Self development plots and the mass encompassing rest of the space program.
The first step of the process involves creating all the logical variants of the basic distribution of the two masses along with the connecting road network and running the simulations for Sunlight hours and Visibility on them.*The basic intention of running these simulations is to pick an option having maximum visibility and sunlight hours on the building forms and the minimum shading of the self development plots by the buildings*.The variants where self development plots were placed in the center of the site were eliminated due to the possibility of shading due to the taller buildings surrounding them. In the visibility simulation points of interest on the IJ river [6.1] Front access road and the canal where a pedestrian bridge is proposed is considered with equal weightage for visibility from all points of interest.The modelling and simulation process is done on Rhino and Grasshopper with the plugin Ladybug for environmental simulations.

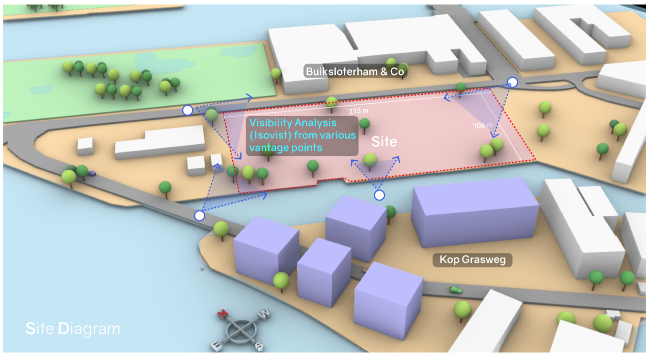


Figure 6.1: Visibility analysis points of interest

The results from the simulation can be seen in [6.4].The resulting simulation values for each design option is used as an input for the MCDA process as described in [6.0.3] for selecting the massing option. Based on the results the massing variant 3 and 5 were picked for further processing.

The Second step of the process involves designing of built and unbuilt spaces (Green spaces,Open public spaces) inside the massing reserved for accommodating the space program.A maximum height of 60m was considered looking at the new developments in the site surroundings. Various design variants were developed and certain set of simulations were selected for defining the performance of the variant which was followed by comparison and eventual selection of the variant. *The simulations used for this stage include the Total Daylight hours simulation,Visibility simulation, Access to Sky from the Unbuilt spaces, Total built space/Sunlight hours,Total floor area (considering an arbitrary floor height of 3m), Total Unbuilt Space, Total built space / Total unbuilt space* . The simulation values are recorded in an excel file and then the MCDA process is performed to rank and select the best options from the generated variants.[6.4],[6.5]. The reason for certain calculations like the ratio of the built/green spaces or the ratio of the floor area with the sunlight hours was considered to avoid designing bigger volumes for a better performance.

In the **The Third step** involves refinement of the massing developed in stage two. The design is analysed in terms of its suitability for development of floor plans for the space program based on the derived dimensions and the best combination

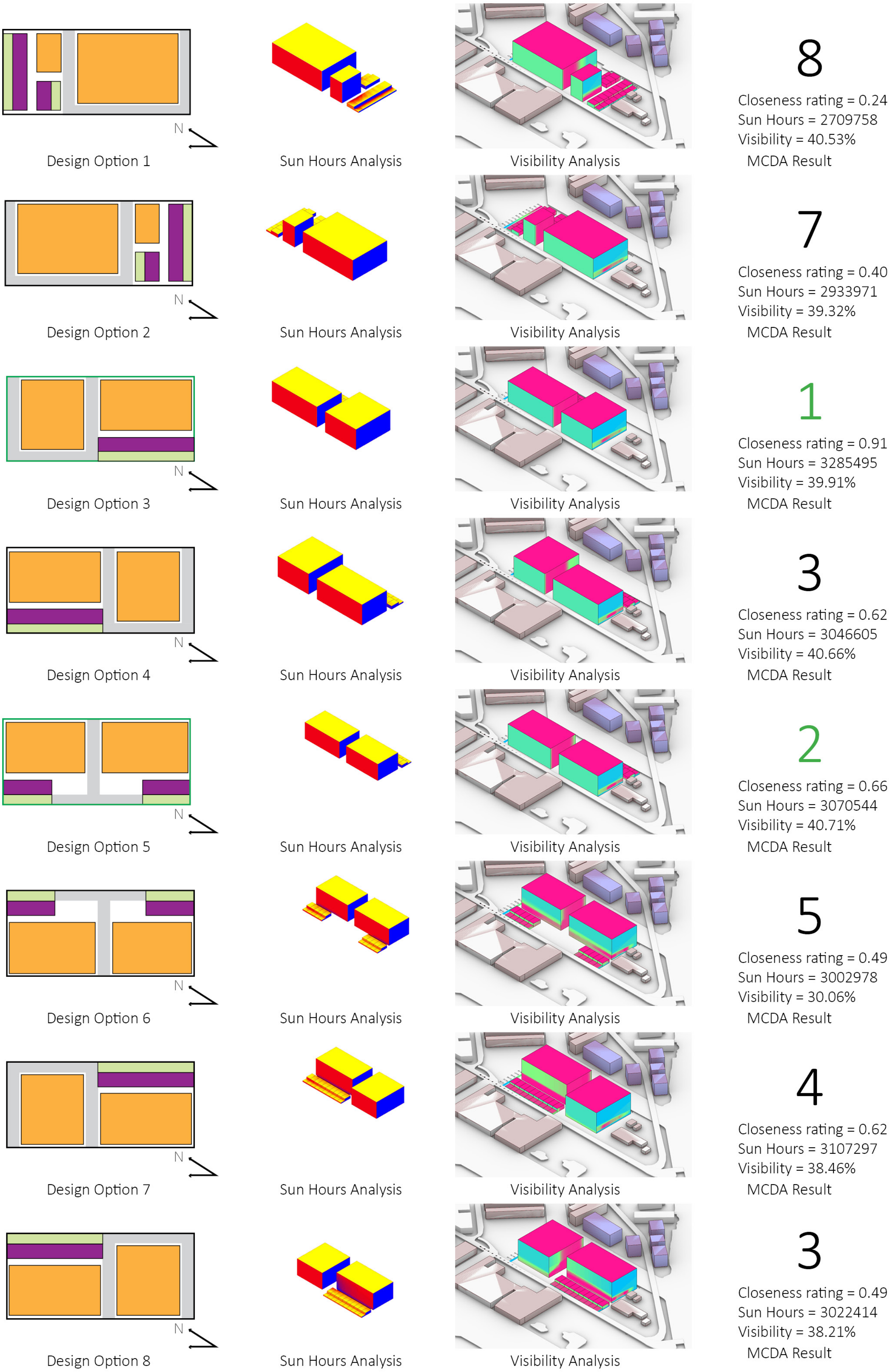


Figure 6.2: First step of massing generation and selection

of the base shape and height of the blocks are explored by converting the top ranked massing options from stage two into parametric models.The parametric models are then linked to the simulation runs from stage two and an evolutionary solver is used to find the right combination of the parameters. The objective function for the solver is to maximise the weighted product of the defined simulation parameters. Constraints are given to the base and height of the blocks according to the design problem formulation.[6.6]

6.0.3 MCDA process used in the Massing problem:

The problem of generating optimal Architectural configuration is a complex one where many decisions have to be taken simultaneously to achieve the configuration goals by multiple stakeholders.The process implemented in the thesis is a methodical one where the multi-dimensional, multi-criteria, multi-actor complexity of decision making in a large problem is divided into smaller ordered logical steps and at each step a set of decision variables is defined and the process of MCDA is applied.

The process of MCDA is capable of solving a decision making problem which has conflicting criteria and weights. The selection of appropriate method for MCDA is crucial for generation of the desired results for the decision making problem. (Wątróbski, Jankowski, Ziemba, Karczmarczyk, & Zioło, 2019) in his paper has described a method to build a formal guideline for MCDA method selection, which is independent of the problem domain and has published a free to use tool for the public [`www.mcda.it`] to choose an appropriate MCDA method.The tool was used to select a MCDA method by defining the abilities of the method.The following details were added to find the MCDA method [has weights, Weights type(quantity),Scale(quantity),Has uncertainty(no uncertainty),Topic(ranking and choice)].When the problem of configuration was analysed in the program results from the tool 6.3 indicated that

TOPSIS could be a method which can be used to solve this MCDA problem. TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) is a multi-criteria decision making technique used to rank a finite set of alternatives based on the minimization of distance from an ideal point and the maximization of distance from an anti-ideal point (Bilbao-Terol, Arenas-Parra, Cañal-Fernández, & Antomil-Ibias, 2014). The normalization step in the method can help in comparing values which have different range and scale and units which is especially beneficial when comparing different environmental simulation values.

This method was also selected as the preferred method since it could solve the problem and was easily available as a Python library.
The Scikit criteria [(Cabral, Luczywo, & Zanazzi, 2016)] python library was used to process the performance data and the user preferences to score and rank the Design variant options.The interactive HTML widgets for jupyter notebooks were used to record the user preferences for the performance indicators.[6.8].

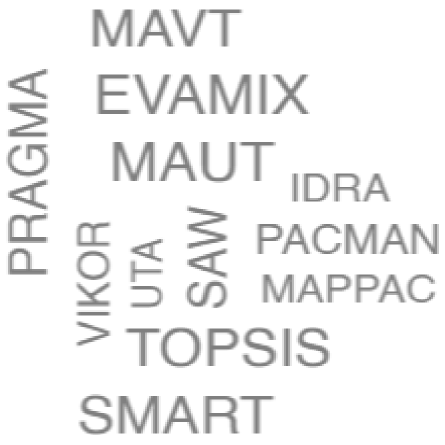
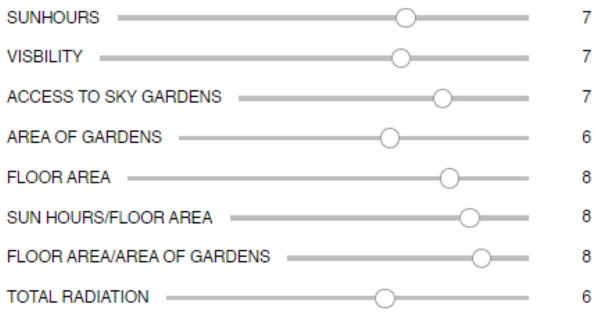


Figure 6.3: All possible MCDA methods which can be used to solve the problem according to the results of the tool



SUNHOURS	7
VISBILITY	7
ACCESS TO SKY GARDENS	7
AREA OF GARDENS	6
FLOOR AREA	8
SUN HOURS/FLOOR AREA	8
FLOOR AREA/AREA OF GARDENS	8
TOTAL RADIATION	6

Figure 6.8: User preference recording from widgets

Other types of MCDA methods available on Scikit criteria like weighted sum and weighted product method were also explored to compare the results along with TOPSIS method.

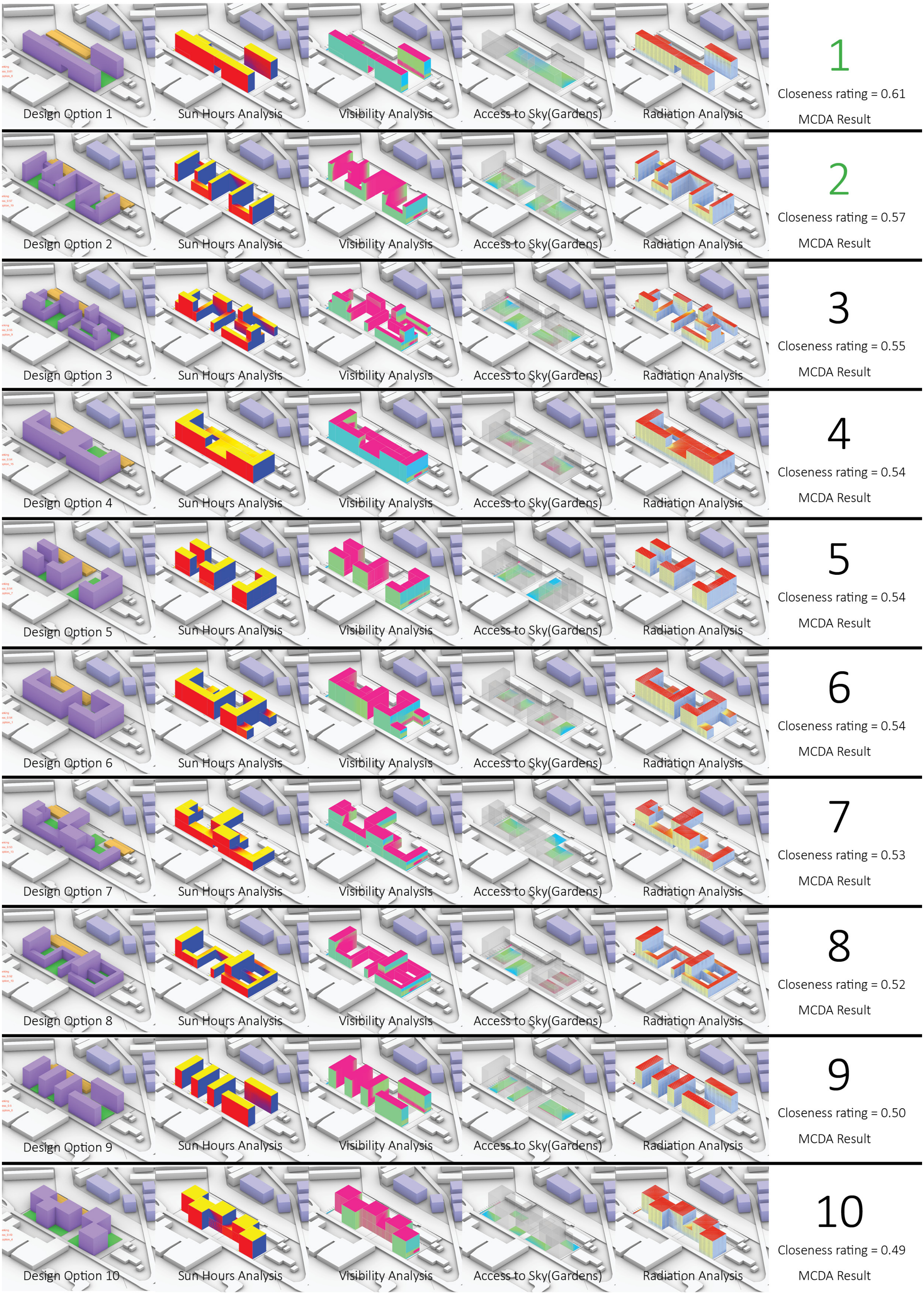


Figure 6.4: Second step of massing generation and selection

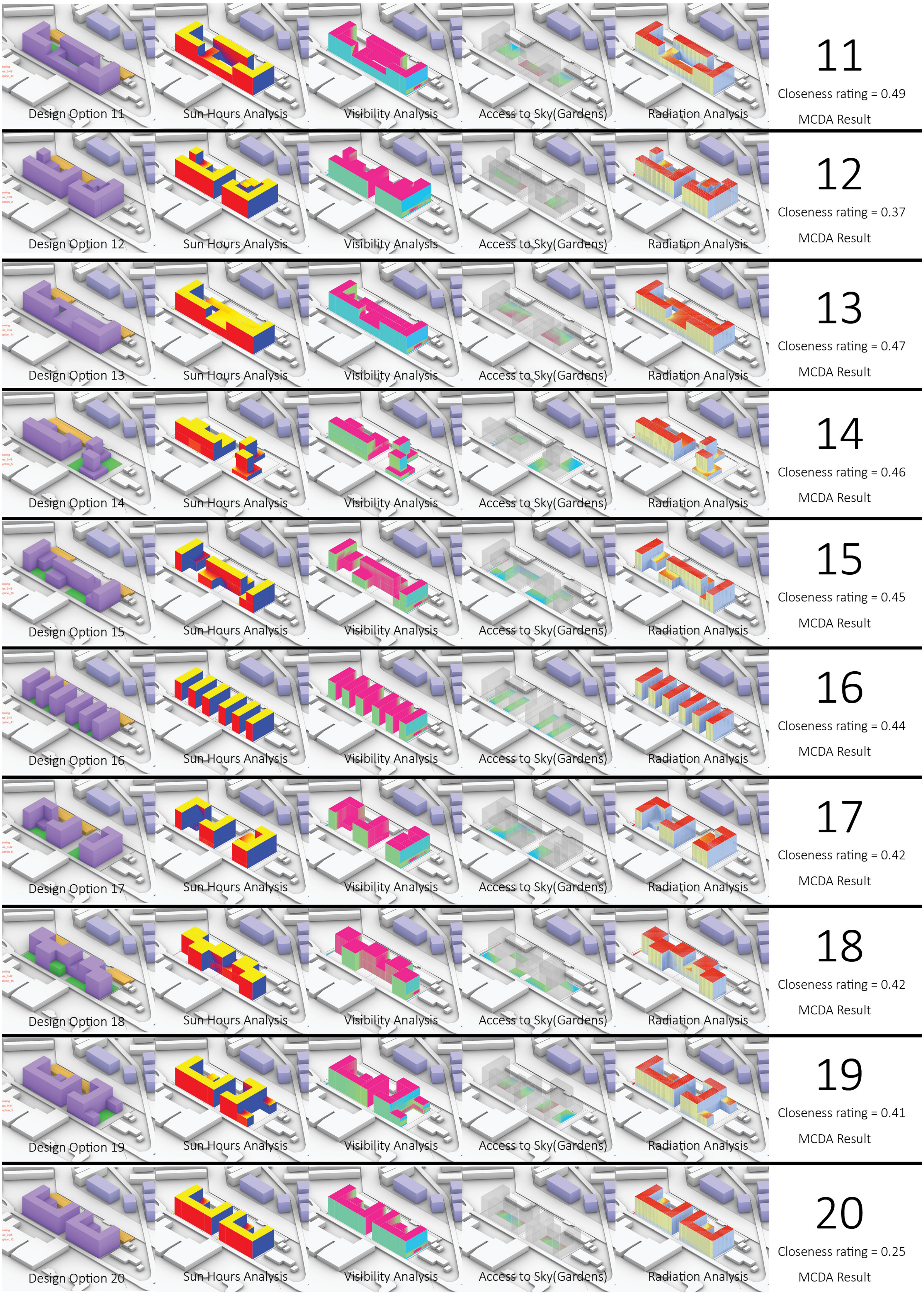
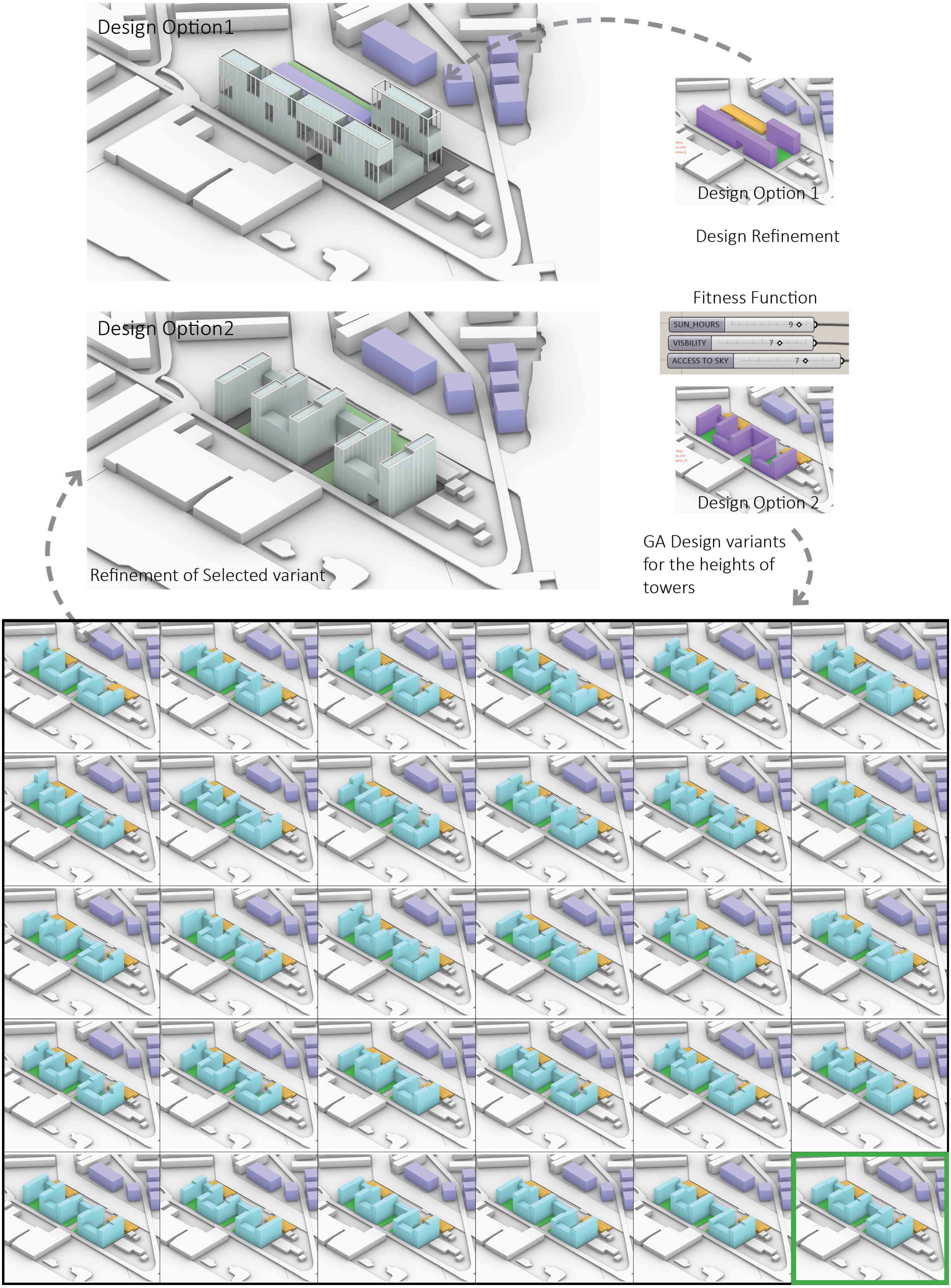


Figure 6.5: Second step of massing generation and selection

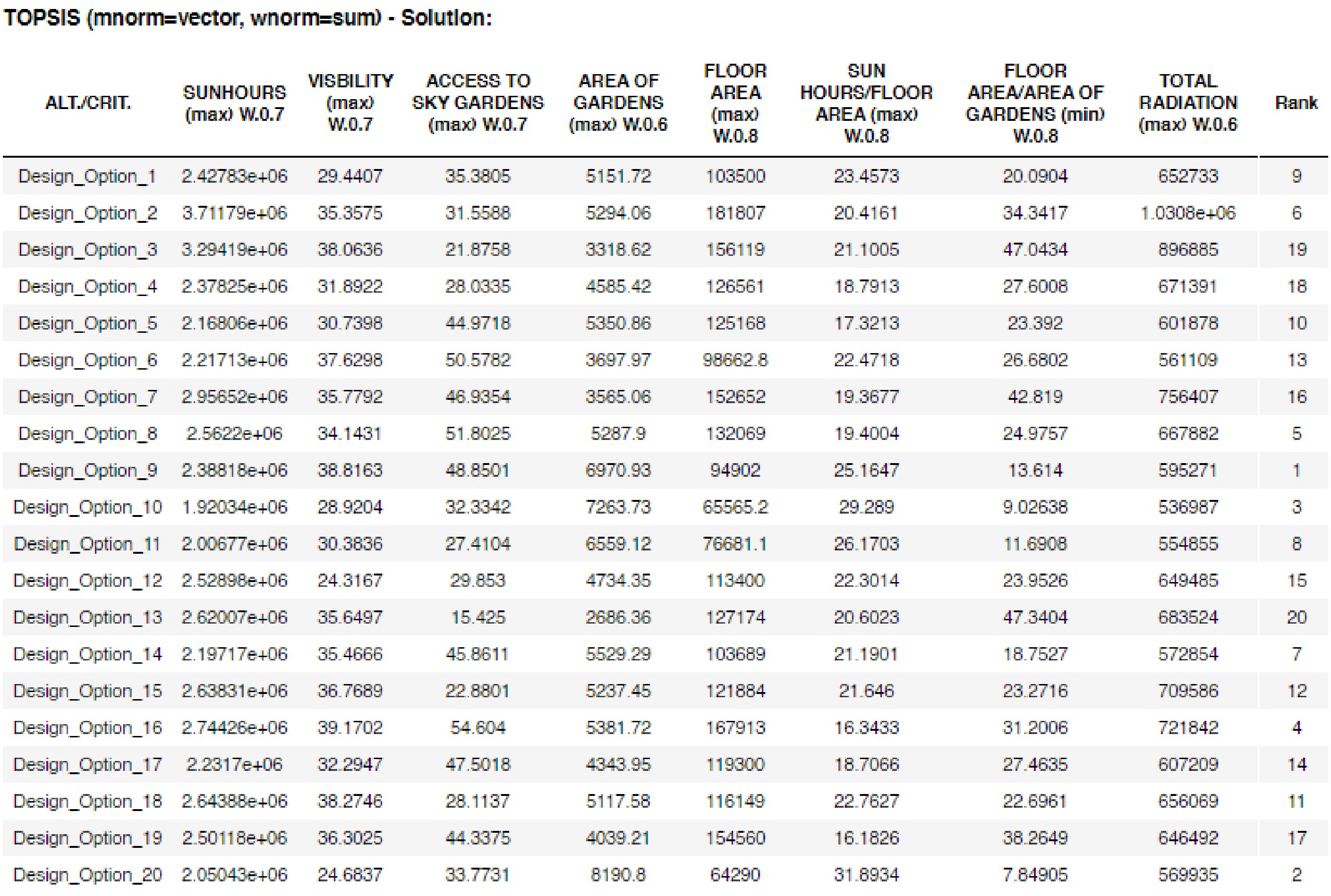


SUN_HOURS 9

VISBILITY 7

ACCESS TO SKY 7

Figure 6.6: Third step of massing generation and selection



TOPSIS (mnorm=vector, wnorm=sum) - Solution:

ALT./CRIT.	SUNHOURS (max) W.0.7	VISBILITY (max) W.0.7	ACCESS TO SKY GARDENS (max) W.0.7	AREA OF GARDENS (max) W.0.6	FLOOR AREA (max) W.0.8	SUN HOURS/FLOOR AREA (max) W.0.8	FLOOR AREA/AREA OF GARDENS (min) W.0.8	TOTAL RADIATION (max) W.0.6	Rank
Design_Option_1	2.42783e+06	29.4407	35.3805	5151.72	103500	23.4573	20.0904	652733	9
Design_Option_2	3.71179e+06	35.3575	31.5588	5294.06	181807	20.4161	34.3417	1.0308e+06	6
Design_Option_3	3.29419e+06	38.0636	21.8758	3318.62	156119	21.1005	47.0434	896885	19
Design_Option_4	2.37825e+06	31.8922	28.0335	4585.42	126561	18.7913	27.6008	671391	18
Design_Option_5	2.16806e+06	30.7398	44.9718	5350.86	125168	17.3213	23.392	601878	10
Design_Option_6	2.21713e+06	37.6298	50.5782	3697.97	98662.8	22.4718	26.6802	561109	13
Design_Option_7	2.95652e+06	35.7792	46.9354	3565.06	152652	19.3677	42.819	756407	16
Design_Option_8	2.5622e+06	34.1431	51.8025	5287.9	132069	19.4004	24.9757	667882	5
Design_Option_9	2.38818e+06	38.8163	48.8501	6970.93	94902	25.1647	13.614	595271	1
Design_Option_10	1.92034e+06	28.9204	32.3342	7263.73	65565.2	29.289	9.02638	536987	3
Design_Option_11	2.00677e+06	30.3836	27.4104	6559.12	76681.1	26.1703	11.6908	554855	8
Design_Option_12	2.52898e+06	24.3167	29.853	4734.35	113400	22.3014	23.9526	649485	15
Design_Option_13	2.62007e+06	35.6497	15.425	2686.36	127174	20.6023	47.3404	683524	20
Design_Option_14	2.19717e+06	35.4666	45.8611	5529.29	103689	21.1901	18.7527	572854	7
Design_Option_15	2.63831e+06	36.7689	22.8801	5237.45	121884	21.646	23.2716	709586	12
Design_Option_16	2.74426e+06	39.1702	54.604	5381.72	167913	16.3433	31.2006	721842	4
Design_Option_17	2.2317e+06	32.2947	47.5018	4343.95	119300	18.7066	27.4635	607209	14
Design_Option_18	2.64388e+06	38.2746	28.1137	5117.58	116149	22.7627	22.6961	656069	11
Design_Option_19	2.50118e+06	36.3025	44.3375	4039.21	154560	16.1826	38.2649	646492	17
Design_Option_20	2.05043e+06	24.6837	33.7731	8190.8	64290	31.8934	7.84905	569935	2

Figure 6.7: Topsis Result for stage 1 of massing problem from Scikit criteria

6.0.4 Limitations:

The massing process described in the previous sections is very subjective in terms of its appropriateness towards designing a building mass.The idea behind the process was to experiment with a methodical way of breaking down the design process into smaller steps so that the decision making process in the whole problem is traceable and does not directly jump into conclusions. With an increase in the accessibility of computational tools and parametric modelling software this whole generation process can be different as long as final outcome of the process (finding the location of built mass on site) can still be used for the next problem in spatial configuration.

6.0.5 Conclusion:

The MCDA methods used in operations research could potentially be a way of making decisions in a design process if the problem is clearly defined as seen in the massing problem.The machine human collaboration process in the massing problem uses the calculation power of machines and the design intelligence of humans to generate a final massing model will represent the boundary for the zoning problem and the massing will be further discretised into smaller voxels which will further be used to generate various lattices for the multi agent system approach taken in the zoning problem. The whole problem of massing could also be viewed in a different manner where the machine can be trained to generate the massing option by determining if a certain voxel or volume should exist on the site or not, that said it has a huge scope and can be a thesis project on its own. Finally to conclude this stage of design the the massing model will be saved as an .obj file and the voxelization process will be done on Python using the meshing python library trimesh [(Dawson-Haggerty et al., n.d.)] and the python library Topogenesis [(Azadi & Nourian, 2020)].

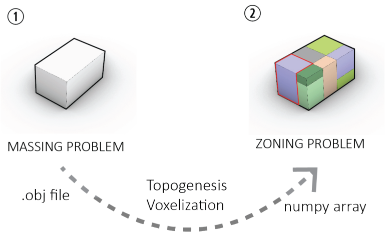
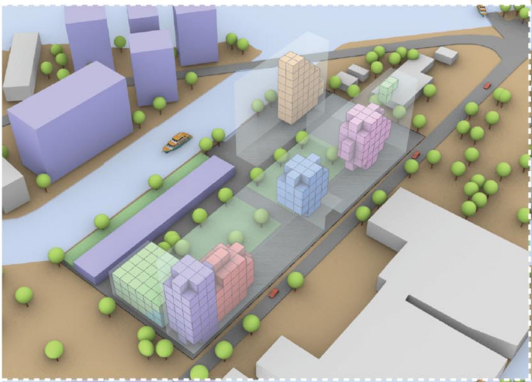
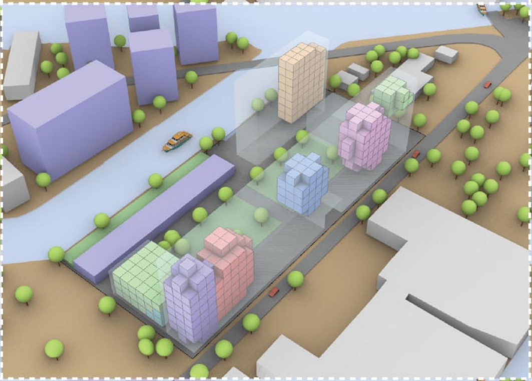
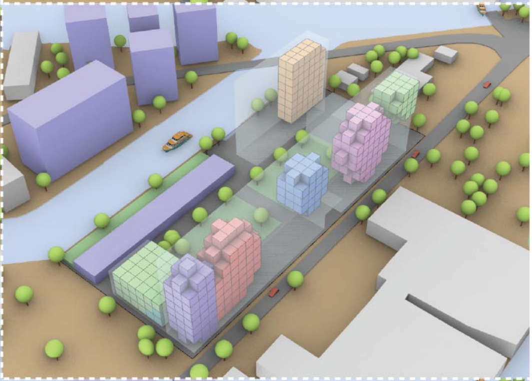
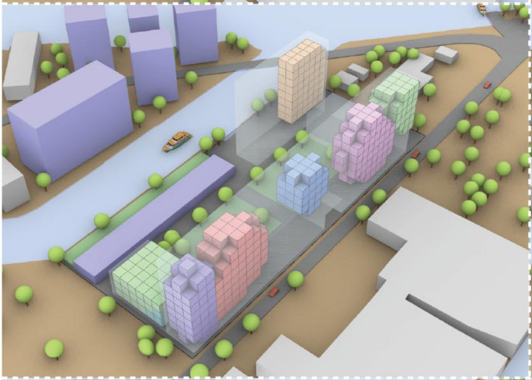
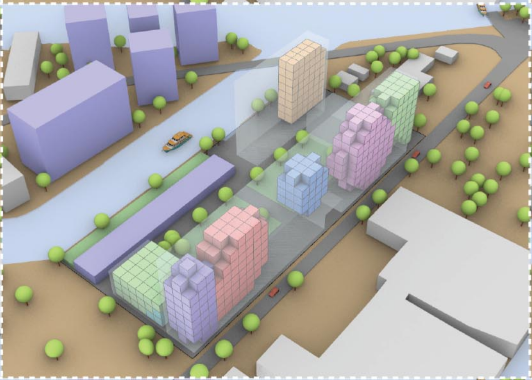
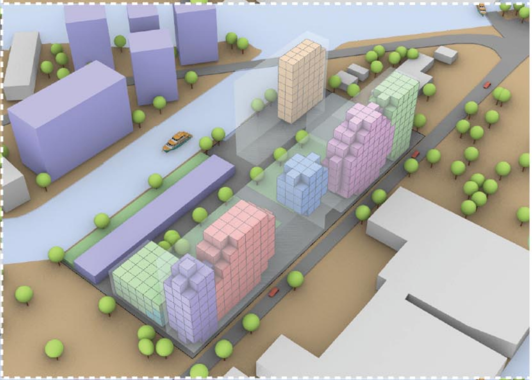
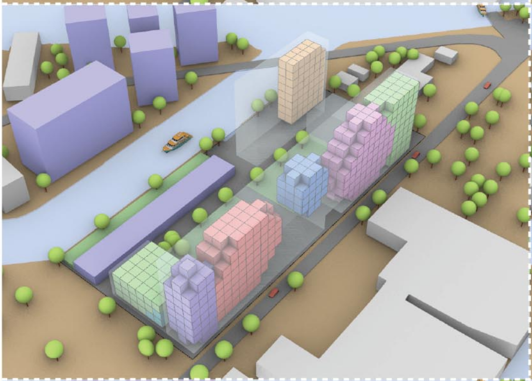
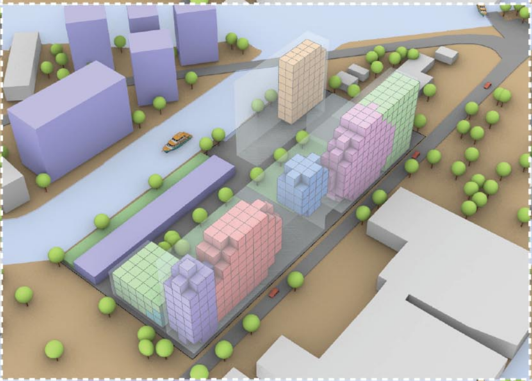
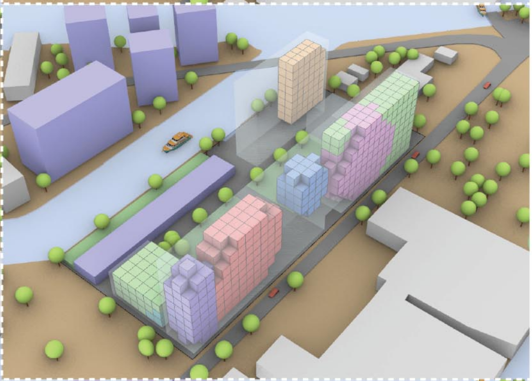
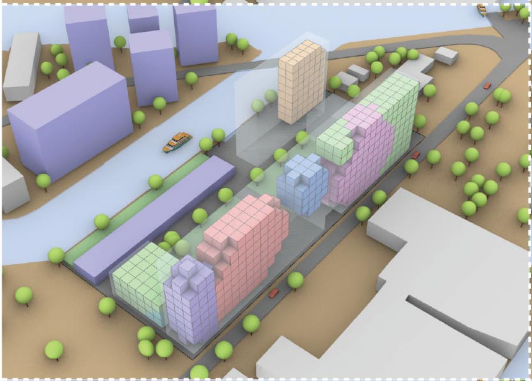
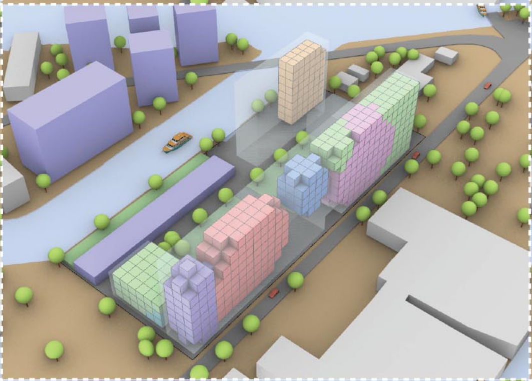
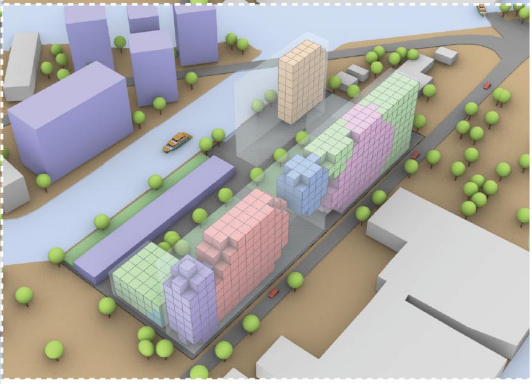


Figure 6.9: Massing problem to the Zoning problem

ZONING PROBLEM

GEN-ARCH

7 Zoning problem

The zoning problem deals with designating the appropriate locations for the various functions in the space program clubbed under particular zones. Two things are primarily important while assigning the zones the closeness requirements of the zones with respect to each other and the appropriateness of the location of the zones w.r.t to the decisions taken by the actors regarding the desired performance criteria.

*The concept of multi-agent systems is used to solve the zoning problem.The system tries to achieve the global and local goals of the zonal configuration like controllable generation of a spatial system consisting of routes pertaining to a graph with closeness weights, spatial zones fitting into locations fulfilling their requirements and fitting their performance goals, and the compactness of the entire system into a space-efficient mass by devising the rules and behaviours of the agents in a quest for emergent optimal configuration such as a termite mound.*Choosing a multi agent system over other techniques is done due to the transparency and adaptability of the method to search for valid options in the massive combinatorial search space along with didactic reasons.The agents represents the actors in the problem and try to grow the decision according to the choices of the actors.

7.0.1 Aim:

To assign Zones in a massing model which will achieve the maximum performance value and the necessary closeness requirements as specified by the stakeholders(actors).

7.0.2 Multi agent system:

In the multi-agent system designed to solve the Zoning problem there are three main elements.

1.Environment Lattices: The environment lattices are developed by performing various environmental simulations on the discretised massing model or the voxel grid generated at the last stage of the massing problem.These lattices geenerate spatial quality indicators which the actors can use to make a decision.

2.Agent Performance criteria: Performance criteria are the lattices which are generated as a result of decision making with respect to the importance and need of the selected spatial indicators for the agent by the actors.

3.Agent Behaviours: Agent behaviours are the actions which the computational agents can perform to achieve their goals on the performance criteria lattices.

7.0.3 Environment Lattices:

The simulations which were performed were based on the specific case taken as described in [5]. The idea at this stage is not to generate a highly accurate simulation models, but simulations which are good enough for decision making. The generation of highly accurate simulations is also not possible at this stage since the material properties and the geometry of the various building elements is not defined yet. he simulations each cater to a spatial quality and can be broadly divided into two types .The first one is based on finding Euclidean distances and the second one is based on Ray obstruction .

The initial step before calculations of simulation lattices was to separate the 4 masses into separate lattices for generating values locally for the individual masses. The voxelated lattice of the massing as derived from [6] is a numpy 3-dimensional array consisting of ones and zeroes. Ones represents the voxels inside the 3d object which are available for occupation and zero represents the exterior voxels which are not relevant to the simulations.The pseudo-code for voxelation is as follows.

Table 7.1: Framework of Algorithm 1: Voxelization algorithm

Input	Data Type	Input Name: Notes
Context mesh	$\mathcal{M}$	The surface mesh representing the context of the building
Output	**Data Type**	**Output Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n\times 1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Problem: generate a voxel array representing the discretised context mesh.		

Algorithm 1 Mesh Voxelization Algorithm

```
Voxelization (M):
    $B \leftarrow$ mesh bounds $m$ M $[lxbxh]$
    $u \leftarrow$ voxel size $[i \times j \times k]$
    $V \leftarrow$ Generate voxel array by B$\div u$
    foreach voxel v in V do
        $C \leftarrow$ generate voxel centroid for each voxel $c$
        if $c$ is not inside $M$ then
            eliminate voxel $v$ by changing its value to 0
            else
                $v$ inside $V$ == 1
            end
        end
    end
    return $V$
```

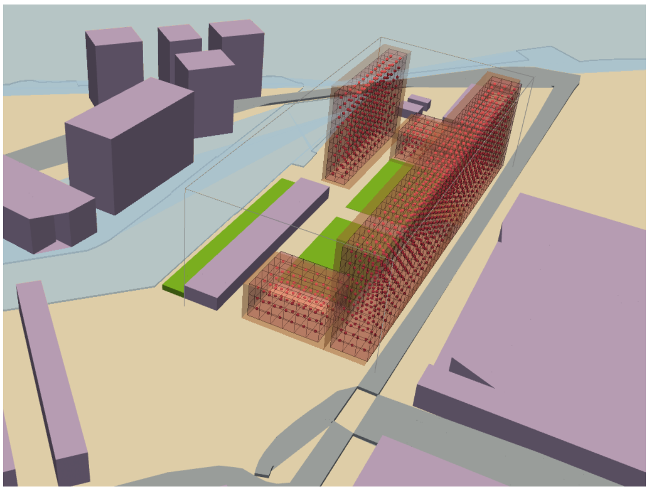


Figure 7.1: Picture of the voxelised envelope

1.Finding the Facade closeness lattices: The facade closeness lattice determines the distance of the voxels in an lattice with respect to the outermost voxels in a specified direction.

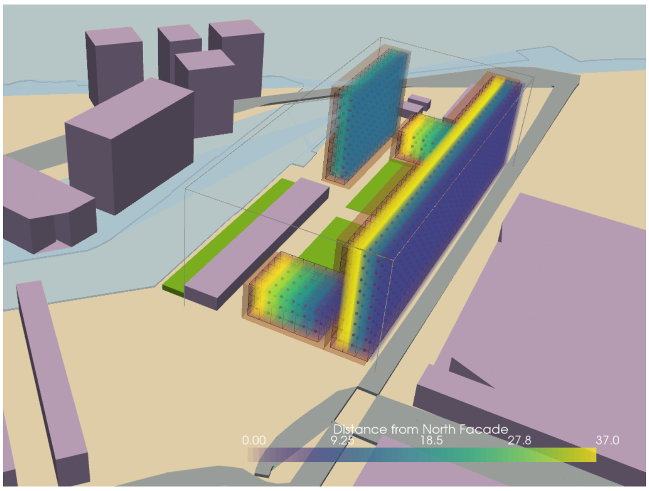


Figure 7.2: Distance from north Facade

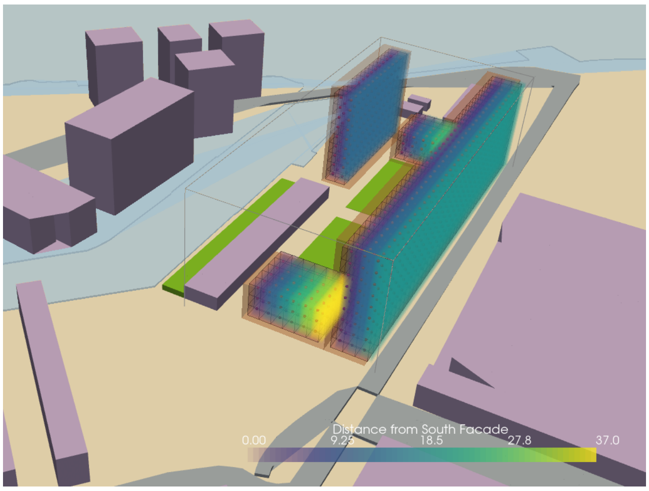


Figure 7.3: Distance from South Facade

The pseudo code for calculating the same is shown in [2]

Table 7.2: Framework of Algorithm 2: Facade Closeness algorithm

Input	Data Type	Input Name: Notes
Array of Voxels	$\mathbf{V} := [v_i]_{n\times 1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Output	**Data Type**	**Output Name: Notes**
Array of Scalar values	$\mathbf{F_c} := [F_i]_{n\times 1}$	Array of scalar values having the same structure as the voxel array
Problem: generate a scalar array indicating the distance of the voxels from the outermost voxel in the choosen direction.		

The calculations for the euclidean distances or the closeness was done for all the facades in the four cardinal directions and to the terrace and the ground for all the masses.

Algorithm 2 Facade Closeness Algorithm

```
Voxel seperation (V):
    $A \leftarrow$ generate adjacency matrix using 6 neighbourhood search stencil
    $G \leftarrow$ generate a graph from $G$

    foreach node n in G do
        $n \leftarrow$ remove disconnected nodes from $G$
        $v_i \leftarrow$ generate island id for each voxel
    end
    return $V_s \leftarrow$ voxel array with island id

Generate closeness lattice ($\mathbf{V_s}$):
    $V_i d \leftarrow$ outermost voxel for all islands $[V_i d]_n$
    foreach voxel v in $\mathbf{V_s}$ do
        $[V_i d][x] \leftarrow$ identify the outermost voxel using island id
        $d \leftarrow$ calculate centroid distance to $[V_i d][x]$
        $s \leftarrow$ replace the $v_i$ with $s$ in $V_s$ to create $F_c$
    end
    return $F_c \leftarrow$ facade closeness array
```

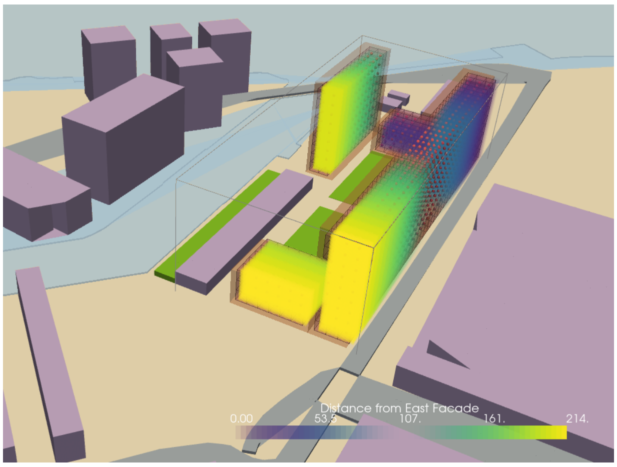


Figure 7.4: Distance from East Facade

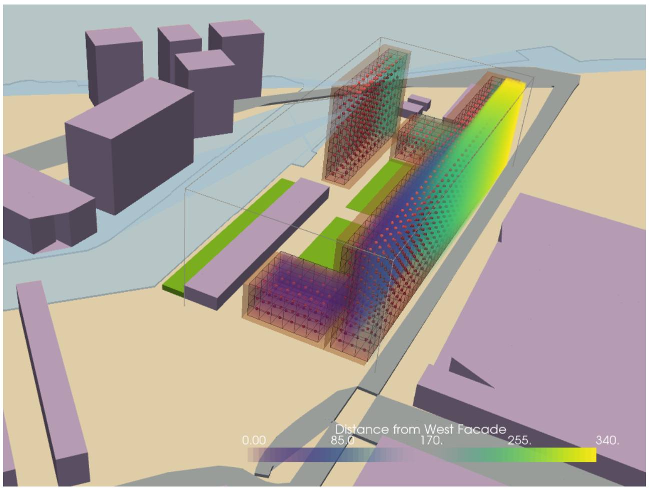


Figure 7.5: Distance from West Facade

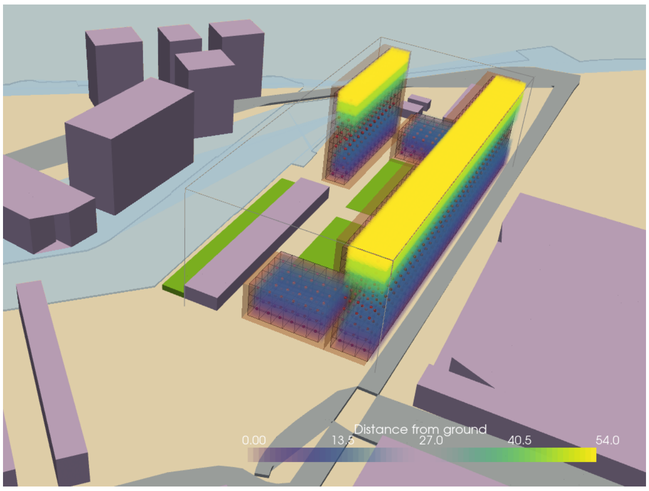


Figure 7.6: Distance from Ground

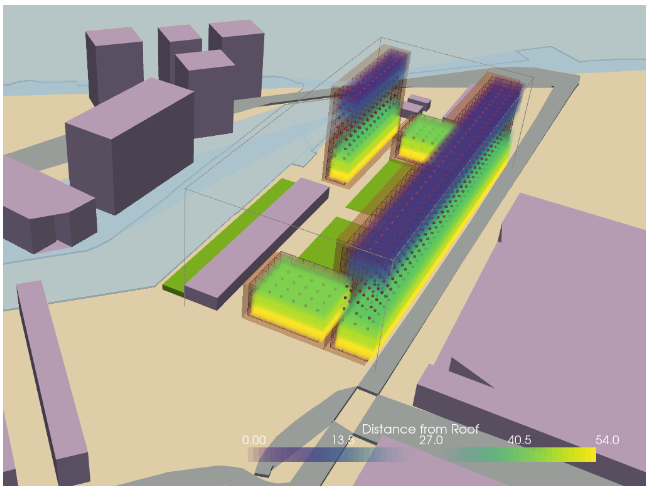


Figure 7.7: Distance from Roof

2.Finding the Quiteness Lattice:
The Quiteness Lattice is a measure of the sound pressure recieved at each of the locations in the mass or the voxelated grid.The calculations related to the sound insulation are complicated and require a lot of data with respect to the material qualities of the building the sound pressure at the source of sound etc.The general equation for airborne sound insulation of a facade is given as:

$$L_{p:inside} = L_{p:outside} - R + 10log\frac{4S\cos\theta}{A_{inside}} \tag{7.1}$$

Where:
$L_{p:inside}$ = Sound pressure level inside in Db
$L_{p:outside}$ = Sound pressure level outside in Db
θ = Angle of incidence
S = Surface area of the facade in m^2
A = Absorption inside the room in m^2

The distance from the source of the sound to the room is critical in this calculation and there can be a 6Db reduction in the sound pressure level with doubling of the distance from the source.This relation can be easily calculated, and will provide a good estimate for decision making regarding the quietness of the voxel with respect to the source of sound.The pseudo code for calculating the same is shown in [3]

Table 7.3: Framework of Algorithm 3: Quiteness Lattice generation algorithm

Input	**Data Type**	**Input Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n\times 1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
List of Co-ordinates	**S**	list of locations indicating the point sources of sound
Output	**Data Type**	**Output Name: Notes**
Array of Scalar values	$\mathbf{Q} := [Q_i]_{n\times 1}$	Array of scalar values having the same structure as the voxel array
Problem: generate a scalar array indicating the quietness of each voxel in a discretised massing model.		

The spatial distance component from Scikit criteria [(Cabral et al., 2016)] can calculate the standard eucledian distance between a group of points where the calculation done is as follows: The distance between two n-vectors u and v is:

$$\sqrt{\sum(u_i - v_i)^2/V[x_i]} \tag{7.2}$$

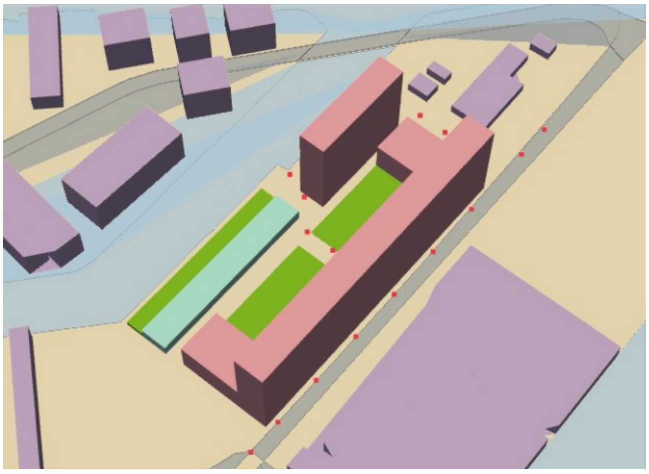


Figure 7.8: line sources like roads have been converted into point sources

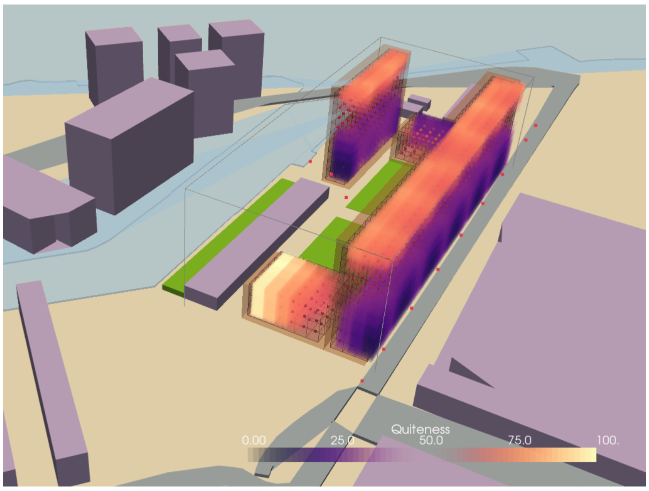


Figure 7.9: Quiteness Lattice

Algorithm 3 Quiteness lattice Algorithm

```
Spatial Distance calculation (V, S):
    $V'$ ← initiate an empty lattice with same structure as $V$
    foreach voxel v in V' do
        foreach point p in S do
            $C$ ← generate voxel centroid for each voxel $c$
            $d$ ← generate distance from $c$ to $s$ $d_{min}$ ← select the minimum distance from$d$
            append value of $d_{min}$ inside V’
        end
    end
    return $V'$ which is equal to $Q$
```

3.Generating the Sunlight hours lattice:

Access to daylight is critical for a good quality of indoor environment and has shown links to productivity and well being. Daylight analysis is generally done in a later stage of design process where the internal layout of spaces and the geometry and sizes of the openings in the spaces are already designed. The material properties of the elements used in facades and inside the spaces themselves also play a big role in determining the amount of daylight received inside a space. At an early design stage the access to sun received by the mass is also a good enough metric to calculate the daylight factor for the building. This can be calculated by calculating the Vertical sky component (VSC). Vertical sky component is the amount of sky which is visible from a point of reference considering the obstructions in the surroundings. That area of visible sky is expressed as a percentage of an unobstructed hemisphere of sky, and, therefore, represents the amount of daylight available for that particular window. It does not consider the size of the room or the number of windows it only calculates the access to the sky. [(IESVE, n.d.)]

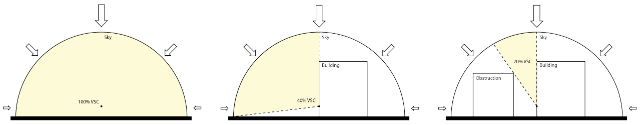


Figure 7.10: Vertical Sky Component[(*Daylight analysis in the design process*, n.d.)]

The library ladybug tools[(Roudsari & Pak, n.d.)] was used to generate the sun positions based on the location for the particular site.These locations were used to create sun vectors from which rays were shot towards the centroids of the voxels. The intersection with the context was checked using the library trimesh[(Dawson-Haggerty et al., n.d.)] and the percentage of non-intersected rays was calculated which is the vertical sky component.

Table 7.4: Framework of Algorithm 4: Sun-access Lattice generation algorithm

Input	**Data Type**	**Input Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n\times1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Sun Vectors	$\mathbf{S_p}$	Selected sun-positions for the HOY’s.
Array of Rays	$\mathbf{r} := [r_k]_{m\times1}$	Array of all the rays that are supposed to be shot from sun toward the voxel centroids
Context mesh	$\mathcal{M}_c$	The surface mesh representing the context of the building
Output	**Data Type**	**Output Name: Notes**
Array of Scalar values	$\mathbf{S_a} := [S_i]_{n\times1}$	Array of scalar values having the same structure as the voxel array indicating the sun-access in percentage
Problem: generate a scalar array indicating the access to sun for each voxel in the discretised model represented in percentage.		

Algorithm 4 Sun-access lattice Algorithm

```
Generate HOY input (W):
    $I$ ← day intervals for the HOY = 30
    foreach day in range(365) do
        if $d$ % $I$ == 0 then
            foreach h in range(24) do
                $i$ ← d * 24 + h $S_p$ = sun-positions[i]
            end
        end
    end
    return $S_p$
r= rays originating from $S_p$ towards voxel centroids Sun-access Lattice construction (r, Mc, V):
    $V'$ ← initiate an empty lattice with same structure as $V$
    foreach voxel v in V do
        foreach ray r in R do
            $I$ ← check intersection of $Mc$ and ray $(r, v)$
              // a ray with the source of $v$ and
              direction of $r$ $I$ is 1 when true and 0
              when false
            $val$ ← percentage of intersections for each voxel
              for all the rays.
              $val$ ← append the $val$ in $V'$
        end
    end
    return $V'$ which equals $S_a$
```

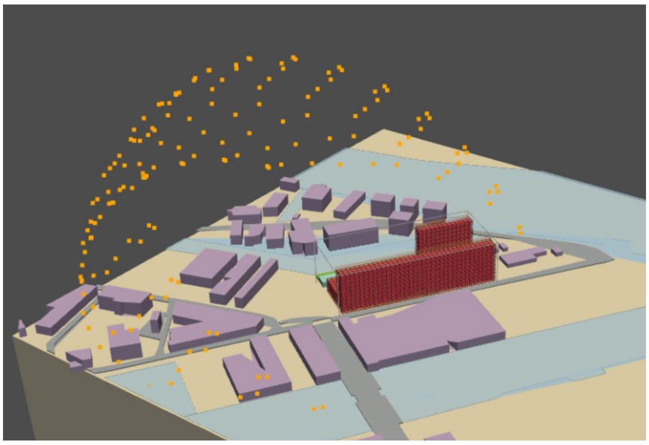


Figure 7.11: Sun Locations for the selected HOY's

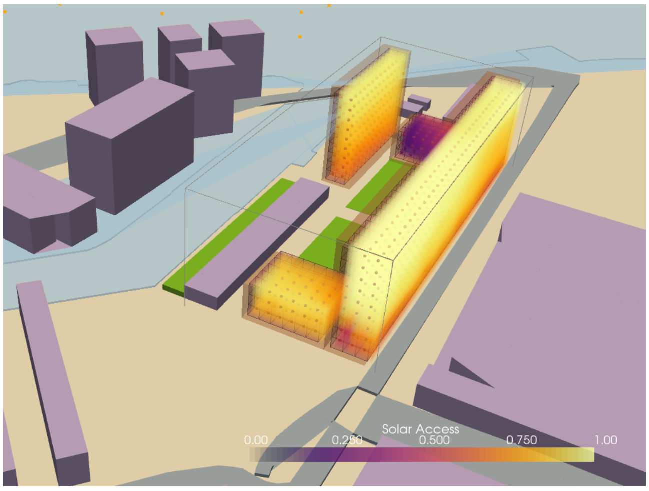


Figure 7.12: Sun Access Lattice

4.Generating the Visibility Lattices:

A visibility analysis is important to consider the compatibility of a building with its surroundings. The visibility of a building from different viewpoints like main streets public areas or other buildings becomes critical to location of certain functions which thrive on a better visibility like retail shops and restaurants. The degree of privacy can also be determined based on the direct visibility of a space from a point.

The characteristics of building locations, building height arrangements, and other circumstances need to be considered in a study of visibility analysis. Visibility studies in a two-dimensional representation have been utilized for a long time for landscape, architectural, and urban studies to calculate the view of a structure or element in the spatial environment. However, a 3D study becomes more comprehensive and accurate to determine the visibility.

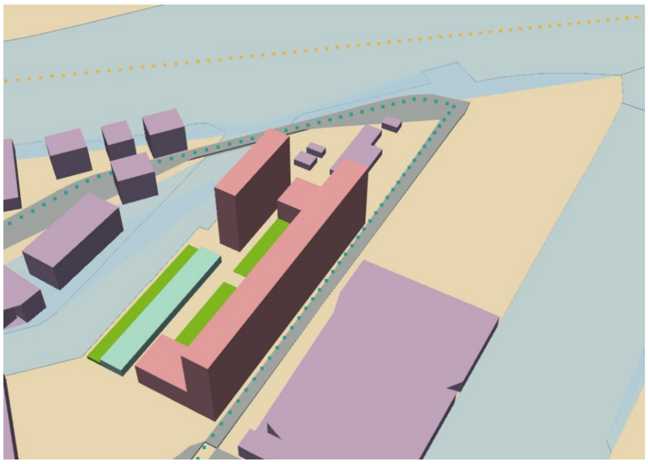


Figure 7.13: Points of interest on the street and the river IJ

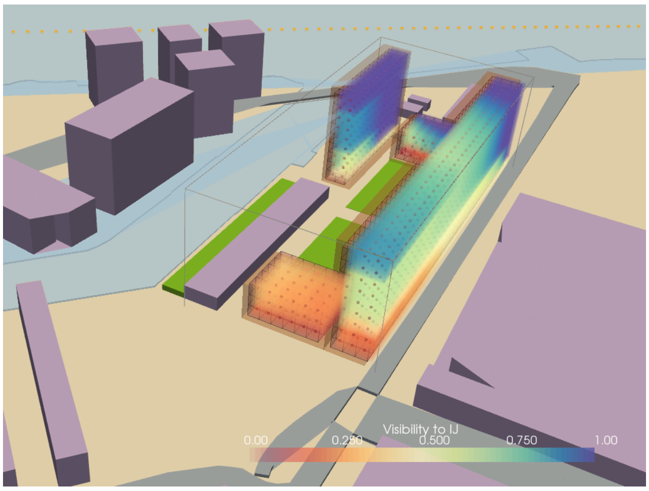


Figure 7.14: Visibility to the River IJ lattice

Table 7.5: Framework of Algorithm 5: Visibility Lattice generation algorithm

Input	**Data Type**	**Input Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n\times 1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Points of interest	$\mathbf{P_i}$	list of point locations to check visibility to/from .
Array of Rays	$\mathbf{r} := [r_k]_{m\times 1}$	Array of all the rays that are supposed to be shot from points of interest toward the voxel centroids
Context mesh	$\mathcal{M}_c$	The surface mesh representing the context of the building
Output	**Data Type**	**Output Name: Notes**
Array of Scalar values	$\mathbf{V_a} := [V_i]_{n\times 1}$	Array of scalar values having the same structure as the voxel array indicating the visibility in percentage
Problem: generate a scalar array indicating the visibility for each voxel in the discretised model towards the points of interest represented in percentage.		

The approach taken to calculate the visibility is similar to the sun access lattice where percentage of intersection is calculated to determine the visibility to a point of interest. For the particular case at Buiksloterham two visibility lattices are generated one which calculates the visibility towards the IJ and the other one calculates the visibility from the road.

Algorithm 5 Visibility lattice generation Algorithm

```
r= rays originating from $P_i$ towards voxel centroids
Visibility Lattice construction ($\mathbf{r}, \mathbf{Mc}, \mathbf{V}$):
    $V'$ ← initiate an empty lattice with same structure as $V$
    foreach voxel v in V do
        foreach ray r in R do
            $I$ ← check intersection of $Mc$ and ray $(r, v)$
              // a ray with the source of $v$ and
              direction of $r$ $I$ is 1 when true and 0
              when false
            $val$ ← percentage of intersections for each voxel
              for all the rays.
              $val$ ← append the $val$ in $V'$
        end
    end
    return $V'$ which equals $V_a$
```

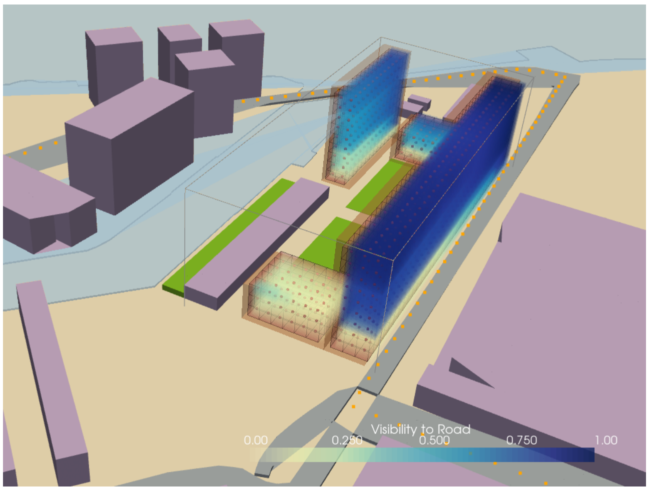


Figure 7.15: Visibility to Road lattice

7.0.4 Agent Performance criteria:

Performance criteria is defined by the stakeholder(actor) participating in the stage of the configuration problem. So in the case of the Zoning problem the Architect decides the performance criteria after having discussions and understanding the requirements of the actors involved in the project. The performance criteria can be defined as an aggregated value that quantifies the performance towards a goal based on the spatial choices of the actors involved.

The steps taken by the stakeholder for defining the criteria are as follows.Firstly the number of criteria required are figured out. It can be done on the basis of the different criteria in the space program. Then the relevant simulations for each criteria are selected. Weights are given to the simulation values and this weighted value is considered for the MCDA process which generates the final performance lattice for the selected criteria.

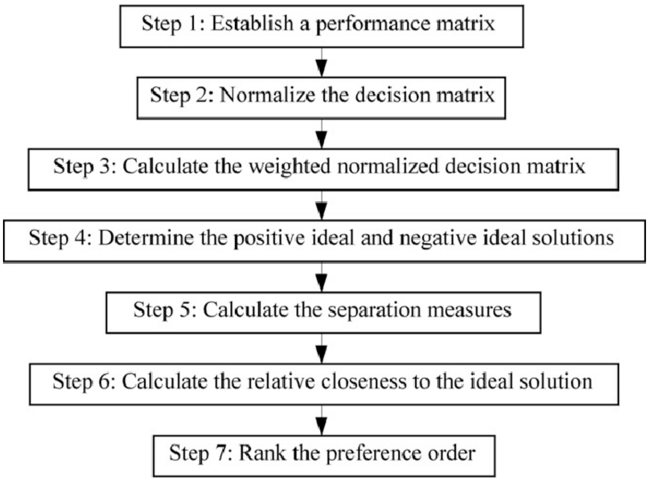
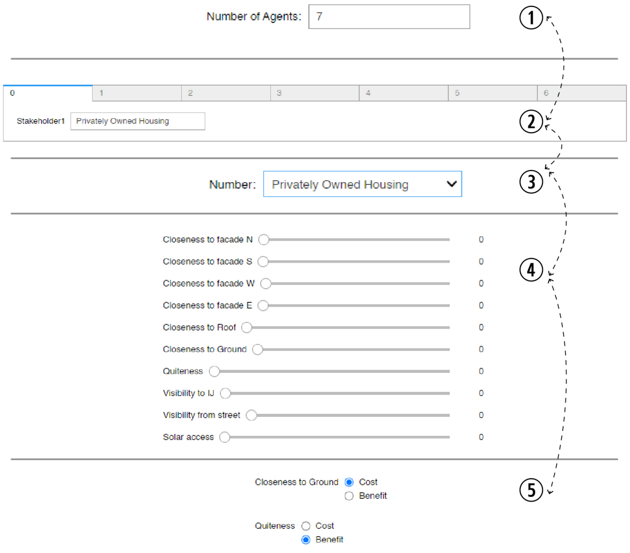

Step 1: Establish a performance matrix
Step 2: Normalize the decision matrix
Step 3: Calculate the weighted normalized decision matrix
Step 4: Determine the positive ideal and negative ideal solutions
Step 5: Calculate the separation measures
Step 6: Calculate the relative closeness to the ideal solution
Step 7: Rank the preference order

Figure 7.16: Topsis method algorithm [(García-Cascales & Sánchez-Lozano, 2013)]

In this stage also the TOPSIS method as described in [6.0.3]is used. The main reason for this is the nature of the method to normalise the values. It enables combining the various simulations with different units together to create a single value which shows the closeness of the value with respect to the ideal.The decision lattice thus created for all the different programs will have the same range of values. These lattices will act as the environment lattices for the agents responsible for those specific functions.



Number of Agents: 7

Stakeholder1 Privately Owned Housing

Number: Privately Owned Housing

Closeness to facade N 0
Closeness to facade S 0
Closeness to facade W 0
Closeness to facade E 0
Closeness to Roof 0
Closeness to Ground 0
Quiteness 0
Visibility to IJ 0
Visibility from street 0
Solar access 0

Closeness to Ground ● Cost ○ Benefit

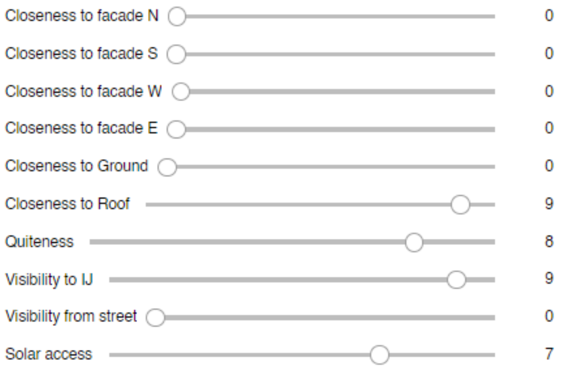
Quiteness ○ Cost ● Benefit

Figure 7.17: Images of widgets used in the Jupyter notebook

Since this step is an interactive one the implementation is done on the jupyter notebook environment of python using the library Ipywidgets(*ipywidgets*, n.d.). The role of the stakeholder is simulated by the author to generate the various lattices required for the case. In the following section the decison matrices for all the various functions in the space program for the case are illustrated with the reasoning behind it.

1.Privately owned Housing:

For the privately owned housing the main criteria was the visibility to the river and the closeness to the roof to create a lucrative opportunity for the investors and owners of the privately owned housing properties in the building. Quiteness and Solar access are also important but have a lower priority than the other criteria.



Closeness to facade N 0
Closeness to facade S 0
Closeness to facade W 0
Closeness to facade E 0
Closeness to Ground 0
Closeness to Roof 9
Quiteness 8
Visibility to IJ 9
Visibility from street 0
Solar access 7

Figure 7.18: Weights given to the various criteria for Privately owned housing

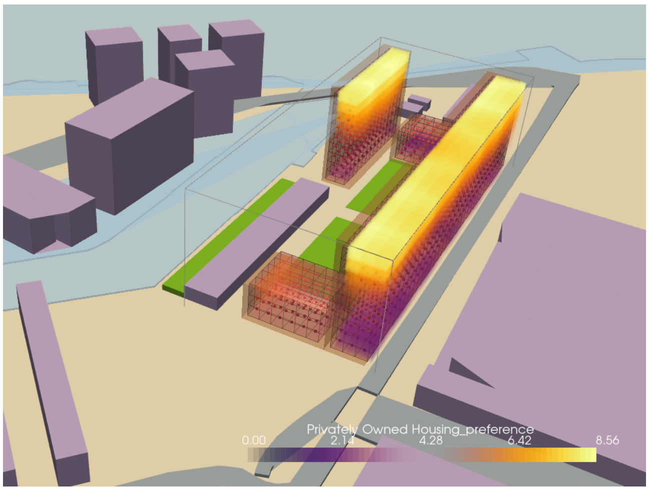
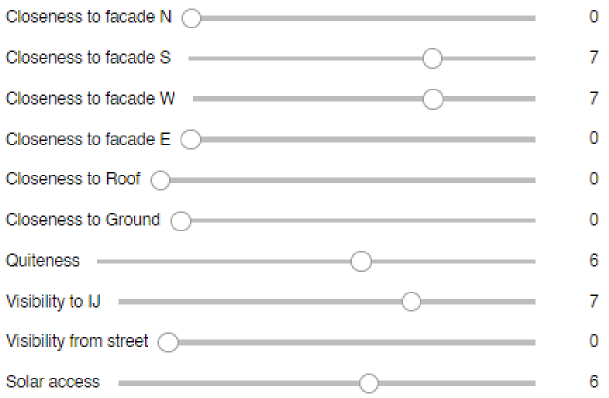


Figure 7.19: Privately owned housing output lattice

2.Free sector Rental housing:

For the Free sector Rental housing priority has been given to the view towards the river and the closeness towards the north and west facade for an even distribution and also to test out the impact of these direction based decisions on the final output. Quietness and solar access have also been considered in the decision framework.



Closeness to facade N 0
Closeness to facade S 7
Closeness to facade W 7
Closeness to facade E 0
Closeness to Roof 0
Closeness to Ground 0
Quiteness 6
Visibility to IJ 7
Visibility from street 0
Solar access 6

Figure 7.20: Weights given to the various criteria for Free sector rental housing

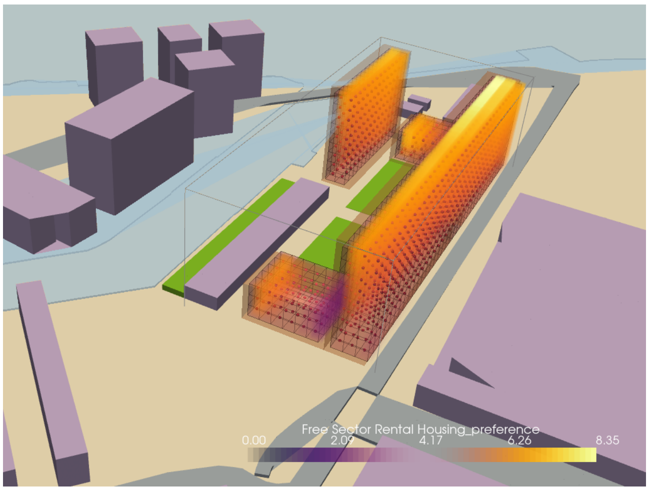
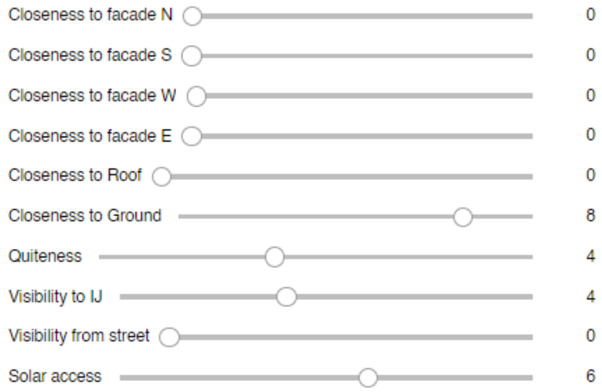


Figure 7.21: Free sector rental housing output lattice

3.Social sector Rental housing:

The hierarchy of decisions for the social sector rental housing preferences were closeness to the ground followed by solar access and finally equal weights for visibility to the river and quietness. The closeness to the ground is added for avoiding the clash with the previous two functions.



Closeness to facade N	0
Closeness to facade S	0
Closeness to facade W	0
Closeness to facade E	0
Closeness to Roof	0
Closeness to Ground	8
Quiteness	4
Visibility to IJ	4
Visibility from street	0
Solar access	6

Figure 7.22: Weights given to the various criteria for Social sector rental housing

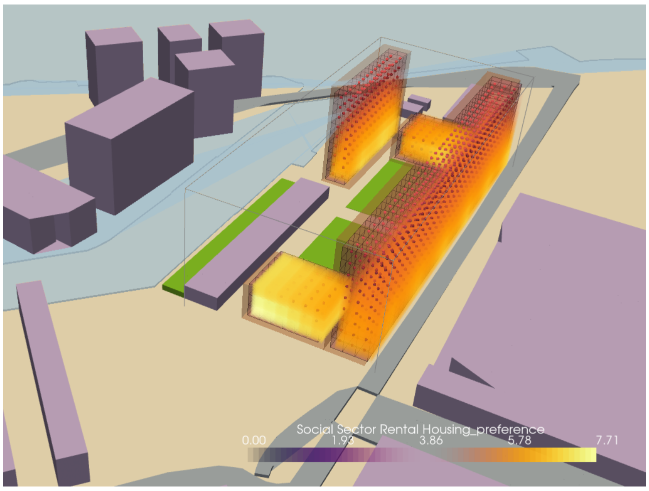
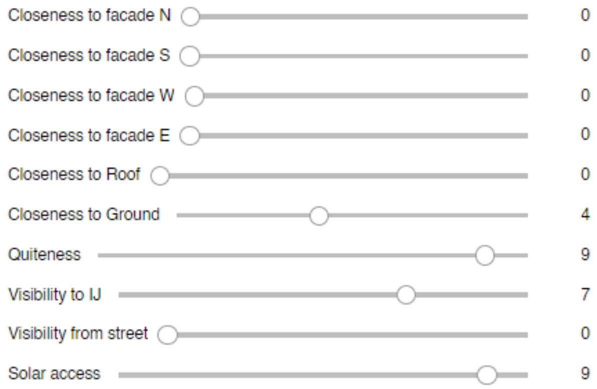


Figure 7.23: Social sector rental housing output lattice

4.Offices:

Quiteness and Daylighting are the most important decision variables for offices.Visibility to the river is also an important criteria for increasing the value of the space and make it a more lucrative investment opportunity.The criteria can be conflicting with the Private ownership housing and Free sector rental housing and the interaction betwwen the respective agents will be interesting to observe.



Closeness to facade N	0
Closeness to facade S	0
Closeness to facade W	0
Closeness to facade E	0
Closeness to Roof	0
Closeness to Ground	4
Quiteness	9
Visibility to IJ	7
Visibility from street	0
Solar access	9

Figure 7.24: Weights given to the various criteria for Offices

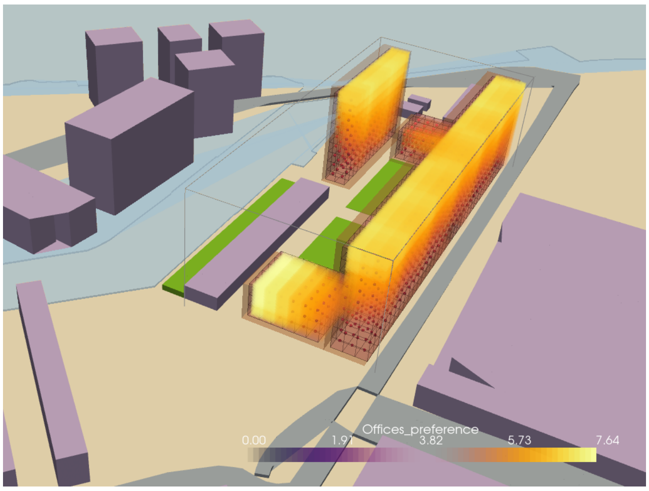
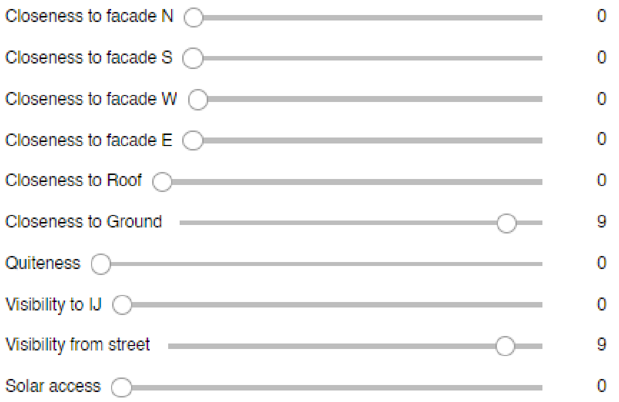


Figure 7.25: Offices output lattice

5.Retail:

The retail spaces have to be accessible from the street and need to have the best visibility from the road . These two decision variables have a bit of a contrasting nature so the combination of them is interesting to analyse.



Closeness to facade N	0
Closeness to facade S	0
Closeness to facade W	0
Closeness to facade E	0
Closeness to Roof	0
Closeness to Ground	9
Quiteness	0
Visibility to IJ	0
Visibility from street	9
Solar access	0

Figure 7.26: Weights given to the various criteria for Retail spaces

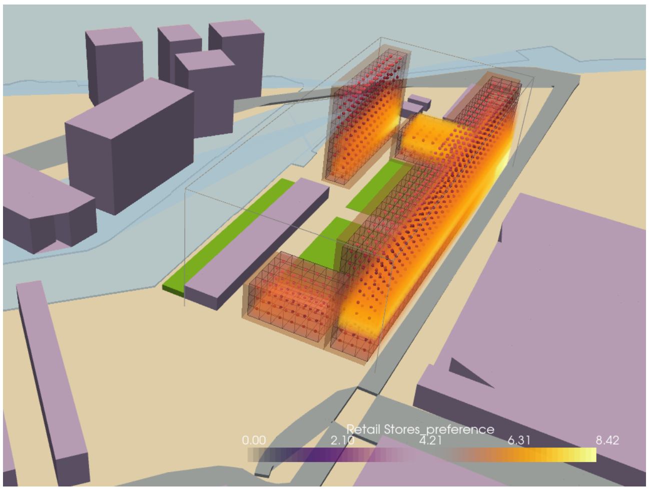
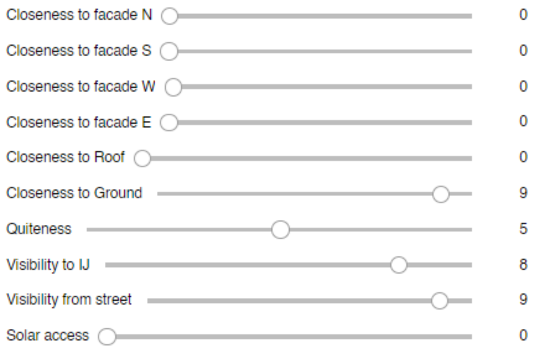


Figure 7.27: Retail output lattice

6.Restaurants and Cafes:
The decision criteria for Restaurants and Cafes are very similar to the retail stores with the added criterias of visibilty to the river and quiteness. So it would be interesting to check the interaction between the agents for both.



Closeness to facade N	0
Closeness to facade S	0
Closeness to facade W	0
Closeness to facade E	0
Closeness to Roof	0
Closeness to Ground	9
Quiteness	5
Visibility to IJ	8
Visibility from street	9
Solar access	0

Figure 7.28: Weights given to the various criteria for Restaurants and Cafes

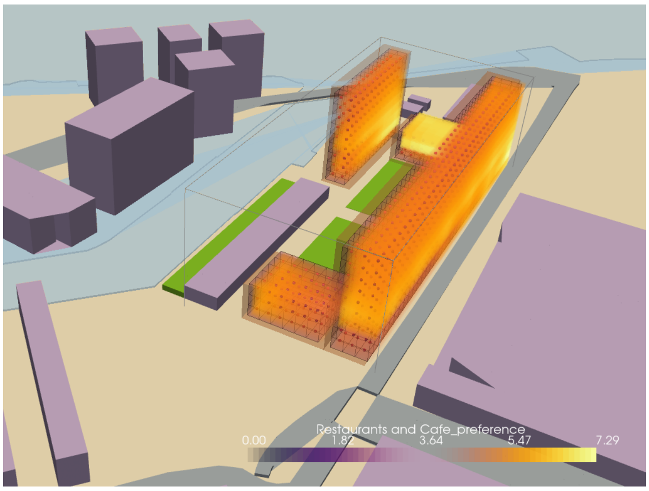


Figure 7.29: Restaurants and Cafes output lattice

7.0.5 Agent Behaviours:

The third part of the multi agent system are the computational agents representing the actors. Before starting with description of the various agent behaviours developed in the project certain terms mentioned in [Terminology] are mentioned again.

1.Agent: An agent is anything that can be considered able to perceive its environment through sensors and act on this environment through actuators.
2.Multi-Agent system: An agent-based system also can exist with several agents having the characteristics as described in the previous paragraph. Such a system is called as a multi-agent system.
3.Agent Behaviors: The actions which the agent can perform to achieve its goals.
4.Stencil: The neighborhood definition in a discrete 3d space.

In the project several agent behaviours were developed. They can be classified into three main criterias : *Agent Origin, Occupy/Unoccupy and Attraction/Repulsion.*

7.0.6 Stencils:

In order to create a boundary for scanning the neighbourhood around a voxel several stencils were defined.The function stencil is available in the library Topogenesis [(Azadi & Nourian, 2020)] where by default there can be two types of neighbourhoods. The first one is the von neumann neighbourhood containing 6 neighbouring voxels and the second is the moore neighbourhood containing 26 neighbouring voxels. The library also contains function to make the center as unoccupied for the stencil.

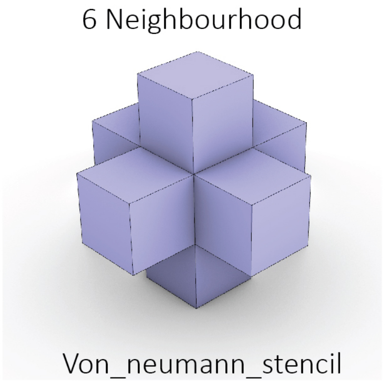


Figure 7.30: Von neumann neighbourhood stencil representation

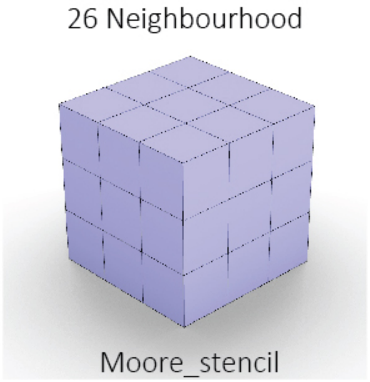


Figure 7.31: Moore neighbourhood stencil representation

The other stencils developed are derived from these two main stencils. The four neighbourhood and the eight neighbourhood stencil are especially programmed for two dimensional agent behaviours where the available neighbours on only a particular axis are considered.

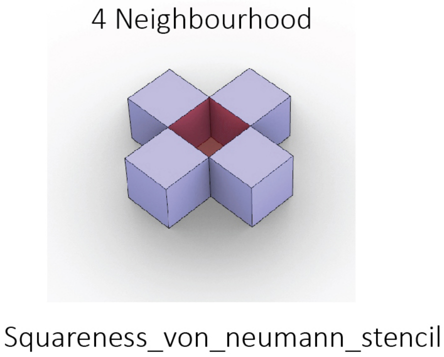


Figure 7.32: Four neighbourhood stencil

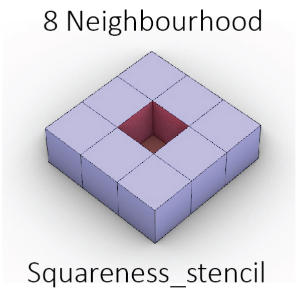


Figure 7.33: Eight neighbourhood stencil

The full floor lattice and the full lattice are considered for searching the whole search space in two dimensions or three dimensions.The full floor lattice considers the maximum extents of the horizontal or vertical slice and the full lattice considers the complete 3d lattice.

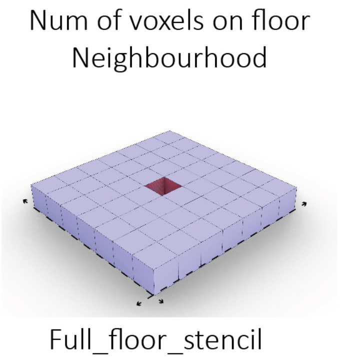


Figure 7.34: Full floor stencil

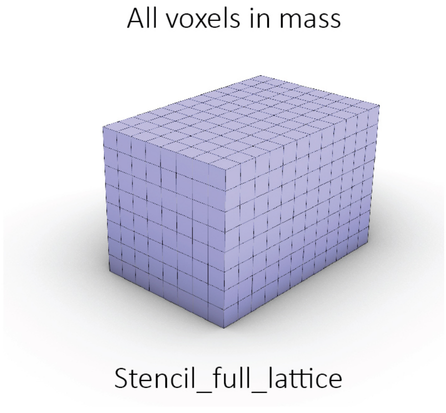


Figure 7.35: Full Lattice stencil

7.0.7 Occupy/Unoccupy Behaviour:

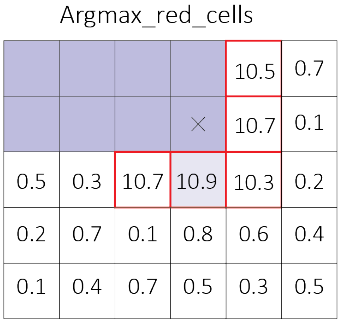
The Occupy behaviour is a class of behaviours which the agent can perform to occupy certain voxels based on their values and neighbourhood condition in a voxelated lattice. The occupy behaviour can be further classified into two dimensional and three dimensional behaviours depending on the stencil which determines their neighbour search space. The occupy behaviour works in the following manner. The first step involves identifying the neighbours based on a selected stencil from the stencils described in the previous section.The desirability value for each voxel can be derived form the performance lattice as described in [7.0.4] which becomes the environment lattice for the agent to perform the behaviour. The values in the environment lattice for the identified neighbours in the occupancy lattice are extracted and the best or the worst value is picked by the agent and the corresponding voxel is occupied.This process goes on till the number of voxels occupied by the agent is satisfied. The generalised pseudocode for the same is shown in [6]

Table 7.6: Framework of Algorithm 6 and 7 : Agent Occupy behaviour algorithm

Input	Data Type	Input Name: Notes
Array of Voxels	$\mathbf{V} := [v_i]_{n\times1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Array of Enviornment values	$\mathbf{E} := [E_i]_{n\times1}$	Array of scalar values having the same shape and size as the voxel array representing a results for decision making about the spatial quality of the zone.
Target growth size	$\mathbf{T}$	List of numbers representing the voxel count which each zone has to grow into.
Search Stencil	$\mathbf{s}_{st} := [S_st]_{n\times1}$	Array defining the neighbourhood definition of the search space.
Output	**Data Type**	**Output Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n\times1}$	Array of coloured voxels or occupied voxels indicating the location of the zones grown in the voxelated array
Problem: grow a decision using computational agents to form zones of same colour in the voxelated array.		

7.0.8 2D Rectangular growth behaviour:

In this behaviour the primary stencil used to find the neighbours is the 4 neighbourhood one [7.32].After the values of the neighbours are retrieved from the environment lattice the maximum valued neighbour is picked for the next iteration. The values for neighbours of neighbours get a boost in this type of behaviour hence the occupancy pattern for the agent is in a rectangular manner. If the behaviour reaches a point where no neighbours are available for occupancy then the search space is gradually increased to 8 neighbourhood area [7.33] and finally to the full floor lattice [7.34].


Argmax_red_cells

				10.5	0.7
			×	10.7	0.1
0.5	0.3	10.7	10.9	10.3	0.2
0.2	0.7	0.1	0.8	0.6	0.4
0.1	0.4	0.7	0.5	0.3	0.5

Figure 7.36: Neighbours of neighbours get a boost in their value

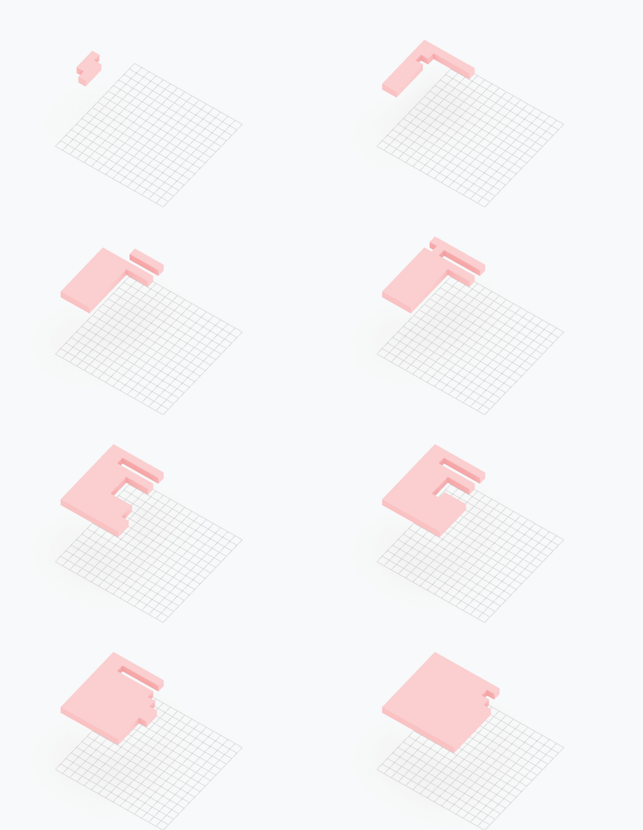


Figure 7.37: 2D Rectangular growth behaviour

The generalised pseudocode is modified in terms of primary stencils , secondary and tertiary stencils and voxel selection criteria for the different agent behaviours. The secondary and tertiary stencils are used as a method of changing agent position if there are no available neighbours based on the primary stencil.

Algorithm 6 Agent occupy generalized Algorithm

```
Define Agent Class (Origin, Stencil, id):
    $O$ ← Agent origin which is a voxel from $V'$
    $id$ ← Agent id which is a number for identification
    $S_{st}$ ← Neighbourhood definition for searching
    $N$ ← Retrieved neighbours for agent at location O according to $S_{st}$
    Define Agent Agent behaviour (self):
        foreach voxel v' in N do
            $A$ ← check availability of voxel for occupancy
                // A value above or below 1 indicates voxel is occupied
            $val$ ← retrieve value from $E$ corresponding to the neighbour id's
            $N_o$ ← The argmax of $val$ is the new agent origin
            update the value in $V'$ with id at the location of $N_o$
        end
        return $V'$ with agent at $N_o$ and voxel at $O$ coloured with the agents colour.
Define Environment Class (E, V', A, B, T):
    $E$ ← Input Environment lattices
    $V'$ ← Copy of input Voxel array
    $A$ ← List of Agents
    $B$ ← List of Agent Behaviours
    return $E_c$ Enviornment
Run Simulation ($E_c$):
    foreach env v' in $E_c$ do
        foreach agent v in env do
            $B$ ← perform the behaviour $B$ from $E$ till the target $T$ is met
            update the value in $V'$ with id's of the agents in the environments $E_c$ of the nase lattice
        end
        return $V'$ coloured voxelated array indicating zones inside the voxelated discretised massing model.
    end
```

7.0.9 2D Circular growth behaviour:

In this behaviour the primary stencil used to find the neighbours is the 8 neighbourhood one [7.33].The rest of the nature of the code is similar to the rectangular one. The main reason for this circular growth is due to the combination of selection of the 8 neighbourhood search space from the first instance and the boosting of values for the voxels whose neighbours have been occupied.This type of growth patterns where the neighbours of neighbours are given a preference ensures the topological nature of the spaces where characteristics like having a single island and no holes in the structure are maintained.

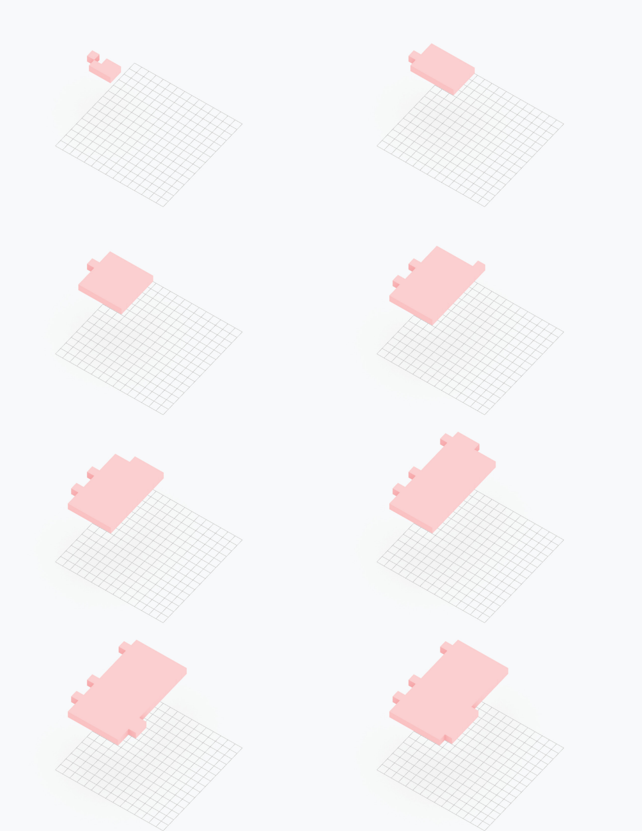


Figure 7.38: 2D Circular growth behaviour

7.0.10 2D Random growth behaviour:

In this behaviour the primary stencil used to find the neighbours can be 4 or 8 neighbourhood one [7.32],[7.33].

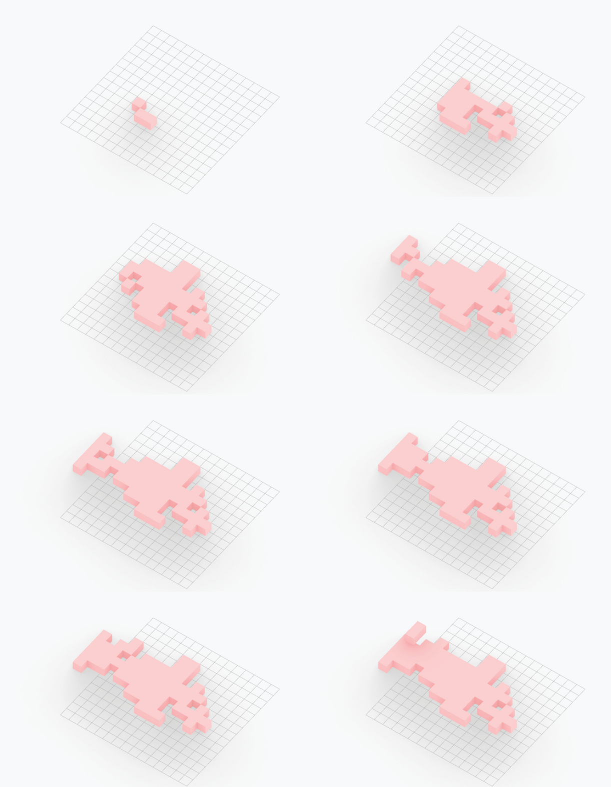


Figure 7.39: 2D Random growth behaviour

The maximum valued neighbour is picked and no boost is given in the values for neighbours of neighbours. This leads the agent to purely follow the values of the environment lattice as the basis for occupancy. This behavior has its advantages when it comes to achieving the maximum valued voxels for a function due to its singular nature in selection but it can also lead to a haphazard growth with many islands in its graph structure and with holes in between.

7.0.11 3D Spherical growth behaviour:

In this behaviour the primary stencil used to find the neighbours is the 26 neighbourhood one [7.31].This allows for selection in three dimensions. The neighbours for each agent position are stored in a list which is referred in every iteration to check for neighbours of neighbours. In this behaviour a boost is given based on the number of adjacent neighbours to a voxel under consideration. This boost creates a behaviour which forms a spherical pattern around the agent origin picking the best possible values in the environment field with the added boost based on neighbourhood occupancy condition.

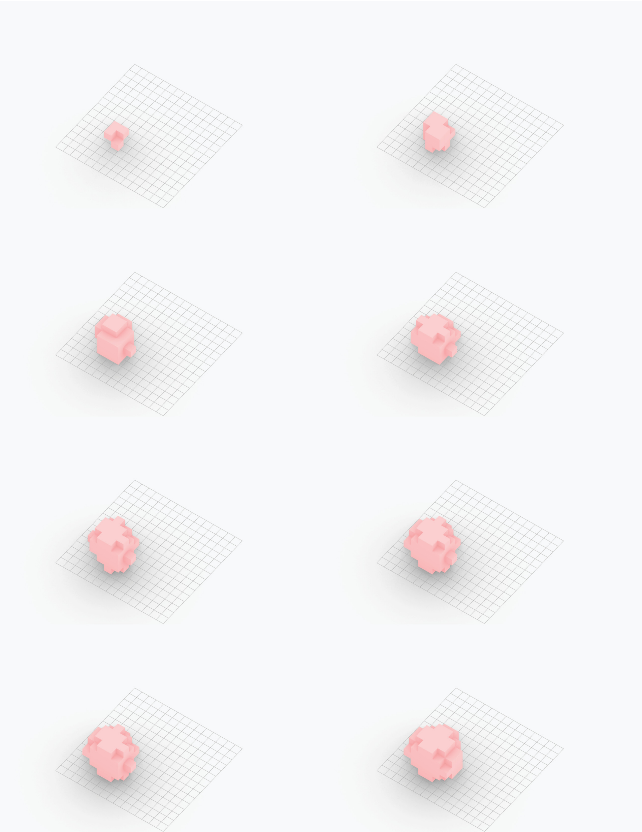


Figure 7.40: 3D Spherical growth behaviour

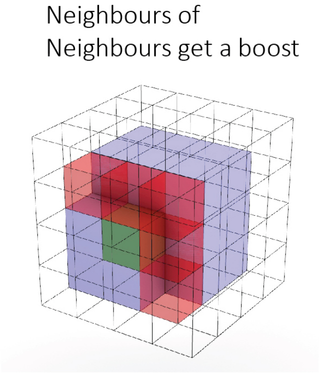


Figure 7.41: Neighbours of neighbours get a boost in their value

7.0.12 3D Cuboidal growth behaviour:

In this behaviour the primary stencil used to find the neighbours is the 6 neighbourhood one [7.30]. Similar to the Spherical behaviour the neighbours for each agent position are stored in a list which is referred in every iteration to check for neighbours of neighbours also a boost is given based on the number of adjacent neighbours to a voxel under consideration.

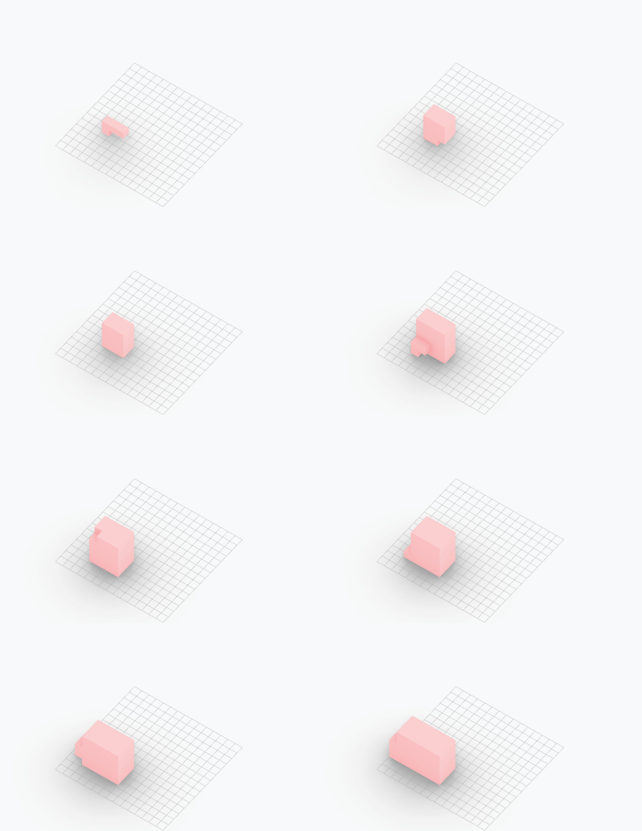


Figure 7.42: 3D Cuboidal growth behaviour

The chances of this behaviour running into conditions where there are zero neighbours is higher since it only considers vertically and horizontally connected neighbours and not the diagonal ones, hence the secondary stencil in this case is the Moore neighbourhood one and finally the tertiary one being the full lattice one so that the agent has the chance to move away from the no neighbour position and restart at a different location.This boost along with the stencils creates a behaviour which forms a cuboidal pattern around the agent origin picking the best possible values in the environment field with the added boost based on neighbourhood occupancy condition.

7.0.13 3D Random growth behaviour:

The 3d random growth behavior is similar to the 2d random growth behavior one but with a different stencil sets. In this behaviour the primary stencil can be the 6 neighbour Von-neumann stencil or the 26 neighbour Moore neighbourhood thus enabling a three dimensional behaviour. The selection of voxels is based on the argmax of the values from the available neighbours and there is no preference for neighbours of neighbours.This leads to a better output in terms of capturing the higher performing voxels for the agent but the topological requirements of single island and no hole polynomio are not satisfied. Ideally these behaviours can be used to unoccupy poorly formed shapes in the zoning model or occupying non connected discrete spaces.

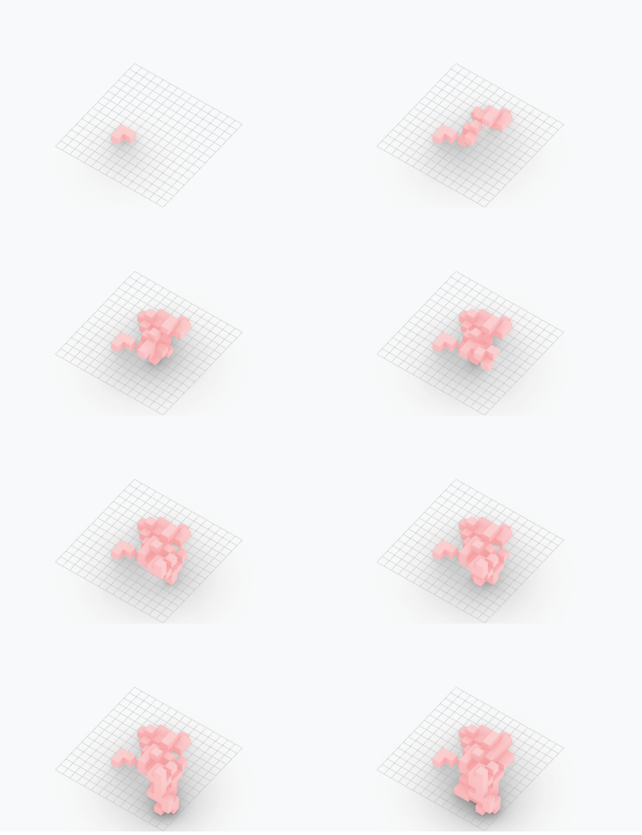


Figure 7.43: 3D Random growth behaviour

7.0.14 Attraction/Repulsion Behaviour:

The 3d growth behaviour of agents towards each other or away from each other can be classified as attraction/repulsion behaviour.The behaviour can be used for either growing (occupying) voxels towards each other or just for finding locations in the graph (generated out of voxelated grid) satisfying certain closeness requirements. The steps for generating the agent behaviours are as follows: First from a voxelated grid using the Von-neumann stencil as a connectivity relation a graph is created.The generation of the graph is done using the library network x[(Hagberg, Schult, & Swart, 2008)]. The minimum distance from all the voxels to the rest of the voxels are generated using the Floyd Warshall algorithm for finding the minimum distances.Then the points of interest or voxels of interest are chosen from the grid and the distance matrix for them are generated using the output from the Floyd Warshall algorithm. Distance matrix is nothing but the graphical distance or Manhattan distance from each voxel to the voxel of interest. Once the distance matrices are generated for both the points of interest they are used as environment lattices for the agent based simulation for the agents to grow or move towards each other depending on the use case.The pseudo code for the same is shown in [7]

Algorithm 7 Attraction/Repulsion agent behaviour Algorithm

```
Generate the Environment lattice (V):
    A ← generate adjacency matrix using 6 neighbourhood search stencil
    G ← generate a graph from G
    D ← calculate shortest distance from each voxel to all the voxels in V using Floyd Warshall Algorithm
    $S_{d1}$ ← extract shortest distance from a selected voxel to all the voxels from D
    $S_{d2}$ ← extract shortest distance from a selected voxel to all the voxels from D
Define Agent Class (Origin, Stencil, id):
    Same procedure as seen in generalised occupy behaviour
Define Environment Class (E, V', A, B, T):
    Same procedure as seen in generalised occupy behaviour
    create two environments with $S_{d1}$ and $S_{d2}$ as environment lattices
Run Simulation ($E_c$):
    Same procedure as seen in generalised occupy behaviour
    return V' coloured voxelated array indicating zones inside the voxelated discretised massing model.
```

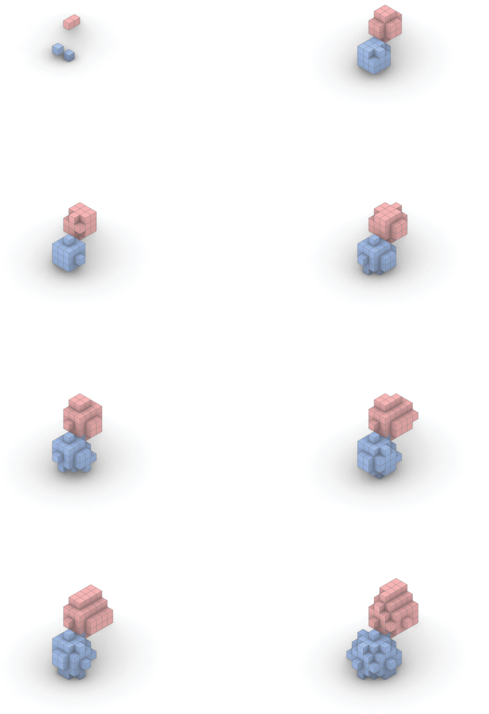


Figure 7.44: Attraction/repulsion behaviour

7.0.15 Agent Origin Behaviour:

The agent origin behaviour is set of behaviours which gives an feedback to the desirability of all location in the voxelated grid for the agent to seed.

Table 7.7: Framework of Algorithm 8 : Agent origin assignment algorithm

Input	**Data Type**	**Input Name: Notes**
Array of Voxels	$\mathbf{V} := [v_i]_{n \times 1}$	Array of all the discrete volumetric elements of the mesh representing the building mass
Array of Environment values	$\mathbf{E} := [E_i]_{n \times 1}$	Array of scalar values having the same shape and size as the voxel array representing a results for decision making about the spatial quality of the zone.
Target growth size	$\mathbf{T}$	List of numbers representing the voxel count which each zone has to grow into.
Search Stencil	$\mathbf{s}_{st} := [S_s t]_{n \times 1}$	Array defining the neighbourhood definition of the search space.
Output	**Data Type**	**Output Name: Notes**
List of 1D Voxel indices	$\mathbf{R_v}$	Voxel origin positions ranked according to the benefit provided.
Problem: Determine which is the most appropriate voxel for the agent to start growing from.		

This is done by first collecting all the neighbours for all the voxels in a base lattice according to the chosen stencil.In this case the stencil are the variations of the 26 neighbourhood Moore stencil.Once the neighbours of all the voxels are collected their corresponding values from the value lattices are extracted. The summation of all the values of the neighbours are the desirability value of the voxel for seeding. Finally the output is sorted according to the calculated desirability.

Algorithm 8 Agent origin assignment Algorithm

```
Generate the Benefit matrix ($\mathbf{V}, \mathbf{E}, \mathbf{T}, \mathbf{S} - \mathbf{st}$):
    $A \leftarrow$ Agent to be deployed to calculate benefit at all the location in the voxelated array
    $B \leftarrow$ Agent behaviour for generating $B_m$ Benefit matrix
    foreach voxel v in V do
        $O \leftarrow$ Set agent origin as voxel $v$
        $b \leftarrow$ occupy voxels in the Voxelated base equal to the number specified in T
        $val \leftarrow$ retrieve values from $E$ using the occupied voxel ids from $b$ and aggregate them
    end
    return Scalar value $val$ for each voxel corresponding to the benefit of assignment for that location
Aggregate the benefit values into $R_v$
```

In order to test the system generated in the thesis a set of Toy problems were formulated and the modules were tested. These can be seen in the next subsection of the Chapter.

7.0.16 Toy Problem formulations for the Zoning problem:

The zoning problem itself can be classified into two types one with a closeness requirement of the zones with each other.An example of this would be hospitals where the closeness requirements between the departments is critical.The other type would be the one without any closeness requirement of the zones like the example case considered in the thesis.In this case the only determining factor behind the objective of the zoning problem is to locate the zones in the best possible locations according the performance criteria defined for it.Several approaches were considered in the thesis for solving both the types of the zoning problem.The first approach which was considered was the Linear assignment approach inspired from the operations research filed in industrial engineering.The second approach was the Total closeness relation approach based again on the research done in solving the facility layout problem in industrial engineering.The third approach was inspired more from the traditional way of practising architecture. It is the Sequential assignment approach which replicates a few of the manual design processes.For each of these approaches toy problems were formulated to test their validity on a simpler and smaller scale. The conclusions derived from these Toy problems were further used to replicate and improve the initial formulation for solving the zoning problem for the case at Buiksloterham [5]

7.0.17 1.Linear assignment approach:

The first approach towards this problem is to deal with the zoning problem as an linear assignment problem researched extensively in operations research. The problem was first described by (Koopman et. al) as a relatively simple problem in the allocation of indivisible resources is that of matching two sets of an equal number n of objects, by making up pairs of objects consisting of one object from each set. Objects belonging to the same set are similar in kind but not identical. For each of the n^2 possible pairs a score or value is given. The problem is to find a matching (or assignment to each other) of objects for which the sum of the scores of pairs matched is as high as possible.

Considering the problem of zoning the objective of this same problem can be reinterpreted as maximising the summation of all the values of voxels in a cluster (function) for all clusters.

$$\begin{array}{cc} & \text{Values} \\ \text{Agents}\ \begin{array}{c} a_1 \\ a_2 \\ \vdots \\ a_N \end{array} & \begin{bmatrix} v_1 & v_2 & \cdots & v_M \\ v_{21} & v_{22} & \cdots & v_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ v_{N1} & v_{N2} & \cdots & v_{NM} \end{bmatrix} \end{array}$$

To illustrate the approach we can consider a problem where there are 4 agents and the benefit which will be gained by placing them in 4 locations is given by the benefit matrix. The probabilities of the n^2 possible agent-location pairs can be set out in the form of a square matrix, the typical element a$_{\text{ki}}$ representing the profit expected from the agent k in location$_{\text{i}}$. A possible benefit matrix of the order n = 4 would be

$$\begin{array}{cc} & \text{Locations} \\ & (1\ 2\ 3\ 4) \\ \text{Agents}\ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \end{array} & \begin{bmatrix} 6 & 9 & 5 & 4 \\ 3 & 2 & 7 & 4 \\ 6 & 5 & 5 & 2 \\ 3 & 9 & 4 & 9 \end{bmatrix} = a_{ki} = A \end{array}$$

A permutation matrix can represent the solution to the assignment problem. This is a matrix P = [P$_{\text{ki}}$] in which there is 1 present only at a single location in each row and column.The 1 represents the assignment of the agent to that location. For the example presented before the permutation matrix would be represented as :

$$\begin{array}{cc} & \text{Locations} \\ & (1\ 2\ 3\ 4) \\ \text{Agents}\ \begin{array}{c} a_1 \\ a_2 \\ a_3 \\ a_4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = p_{ki} = P \end{array}$$

The profitability(benefit) of the assignment can be written as :

$$\pi = \sum_{k,i=1}^{4} a_{ki} p_{ki} = 9 + 7 + 6 + 9 = 31$$

Hence the objective for zoning can be interpreted as assigning the agents for each function to a location in the voxelated grid so as the total benefit of the zoning configuration is maximised.

However the assignment for each voxel in a mass to an agent is not an appropriate way to go forward with it for two reasons. Firstly depending on the size of the voxel grid and the number of agents this calculation for finding combinatorial possibilities can become complex and too large to compute with a small increase in the number of agents. Secondly all the occupied voxels in a zone need to be topologically connected and coherent to form a zone (The zones should have a single island and not have any holes) which wont be possible in this approach.In order to overcome these problems certain simplifications are necessary.

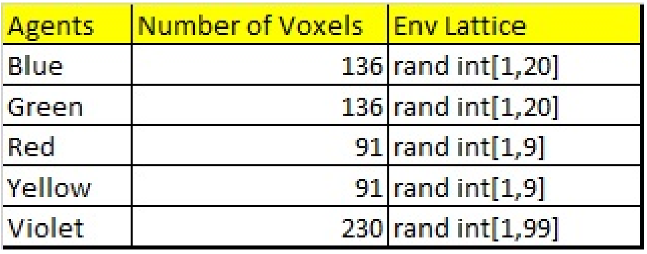
7.0.18 Assumptions and Simplifications:

Based on the agent behaviours developed in the previous section of the thesis [7.0.5] the cost of placing the agent at a certain location will be determined by the summation of all the values occupied by the agent growth starting from the origin.This can simplify the problem to just assigning the center of the growth for the agents or the origin points at appropriate locations. This will reduce the assignment problem to a much more sizable proportion, However there is still a problem which creates extra complexity in the linear assignment problem . It is the problem of finding the benefit of assigning the agent to a certain location. If all the locations (voxel positions) in the mass are eligible to be the origin point for the agent growth then determining the best combination again becomes a massive task to compute. In order to solve this problem another simplification is made .The whole voxelated base lattice is divided into smaller sections and the centers of each section is considered as a possible location for agent assignment. This reduces the number of iterations which need to be computed and does not affect the accuracy of the final result.

The simplifications mentioned above does make the combinatorial complexity a little simpler but the problem of generating single island zones with no holes still remains. In the agent behaviours developed for the multi agent system there are certain behaviors under the class occupy like the 2d_rectangular[7.0.8], 2d_circular[7.0.9], 3d_cuboidal[7.0.12],3d_spherical[7.0.11] which have the inbuilt ability to only create connected zones which have no holes. So in the simulations only these can be used which will automatically satisfy the topological constraint of the growth behaviour of the agents.

7.0.19 Toy Problem formulation for Linear assignment problem:

The following details are considered for the Toy problem.
Size of the Lattice = 19x9x4 (LxHxW)[684]
Number of Agents: 5
Agents: (Blue , Green , Yellow , Red, Violet)



Agents	Number of Voxels	Env Lattice
Blue	136	rand int[1,20]
Green	136	rand int[1,20]
Red	91	rand int[1,9]
Yellow	91	rand int[1,9]
Violet	230	rand int[1,99]

Figure 7.45: Toy Problem Details

The env lattices have been given a bigger value just to boost the importance of the largest agent in the whole simulation.

Objective: To find the best location for the origin of five agents Blue, Green ,Yellow, Red and Violet so as to maximise the total benefit which is the summation of all the values gained by the agents from their respective enviornment fields during the simulation.For the enviornment fields a random integer set having the same shape and size as the base lattice is considered.

$$Benefit = \sum_{k,i=1}^{5} a_{ki} p_{ki} = V[B] + V[G] + V[R] + V[Y] + V[V] \tag{7.3}$$

Where:
$a_{\_ki}$ = benefit matrix
$a_{\_pi}$ = permutation matrix
V = value lattice
B,G,Y,R,V = indices of occupied cells for the agents

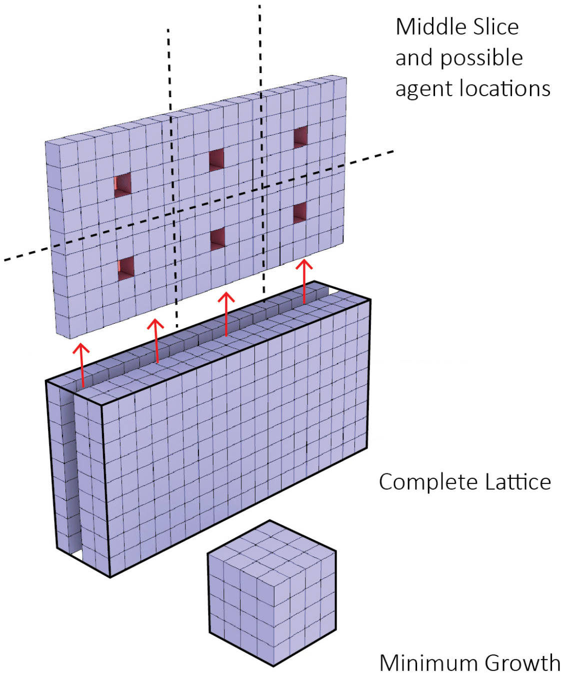


Figure 7.46: Toy Problem agent origin locations

7.0.20 Agent based Simulation procedure:

Step1: Determine the possible agent origin locations
This was done by dividing the whole lattice into sections which are atleast as small as the smallest agent or having the number which is at least equal to the number of agents. This resulted in six possible locations for the 5 origins so total permutations possible are 720 calculated using the permutation formula.

$$P(n,r) = \frac{n!}{(n-r)!} \tag{7.4}$$

Where:
P is total number of permutations
n is the total number of objects in the set
r is the number of choosing objects from the set

Step2: Determine the benefit at all locations
Simulate agent behaviour at all locations independently and calculate the benefit for each agent at each location. The benefit will be the summation of the values of all the cells from the env lattice corresponding to the occupied cells by the agent at that location.

Step3: Use the MIP solver to generate the permutation matrix
Once the cost matrix is developed then using the MIP solver in Google OR tools optimization is done to find the permutation matrix for the assignment of the agents to the appropriate location for the maximum benefit.

Step4: Validation of Results by simulating all the possible permutations.
The benefit calculated by simulating the agent based behaviour on the locations of the permutation matrix and comparing that with the simulations from all the possible 720 permutations to check if the permutation matrix from the step 3 is the best one.

7.0.21 Results:

After the simulation was completed the total benefit which was calculated from the assignment of the agent origins according to the permutation matrix was not the best solution from the 720 possible options. The ranking for the permutation matrix was really poor 678/720 for the simulation from all the aggregated values.

Algorithm 9 Procedure for Linear assignment approach

```
procedure INITIALIZATION
    Load base lattice
    Generate environment lattices
procedure GENERATE THE POSSIBLE ORIGIN POINTS
    Extract the indices of all possible points
    Create all possible permutations from it
procedure CREATE MULTI-AGENT SYSTEMS
    Create the necessary stencils
    Create the necessary Agent class
    Create the necessary Environment class
procedure GENERATE THE BENEFIT MATRIX
    Do a multi-agent simulation for all the agents at all locations
    Save the simulation output as benefit matrix
procedure LINEAR ASSIGNMENT TO GENERATE THE PERMUTATION MATRIX
    initialize the Google OR tools MIP solver using benefit matrix
    Generate the permutation matrix from the solver
    Calculate the benefit generated from the permutation matrix    ▷ This can be done via extracting the corresponding values from the env lattice
procedure VALIDATION OF THE RESULT
    Generate the total benefit for all the permutations and sort them.
    Compare the permutation matrix benefit to the best option from all the simulations to rank the permutation matrix solution.
```

7.0.22 Possible improvements:

The main two concerns in the simulation done were as follows. The environment lattice were all made up of random integers with no patterns emerging out of it. This makes the problem of assignment very complicated an unpredictable due to the random nature of the values.

Secondly the effect of one agents growth over the other was not considered at all which is not a good thing since the violet agent for example needs to grow for about 230 voxels which is more than 2.5 times the growth of Red and Yellow.Introducing the added complexity of the relative position of agents witch each other in the step where benefit for assignment is calculated will make the problem very complex to solve. A possible solution to this would be to divide the assignment problem into various numbers of equal sized agents. This

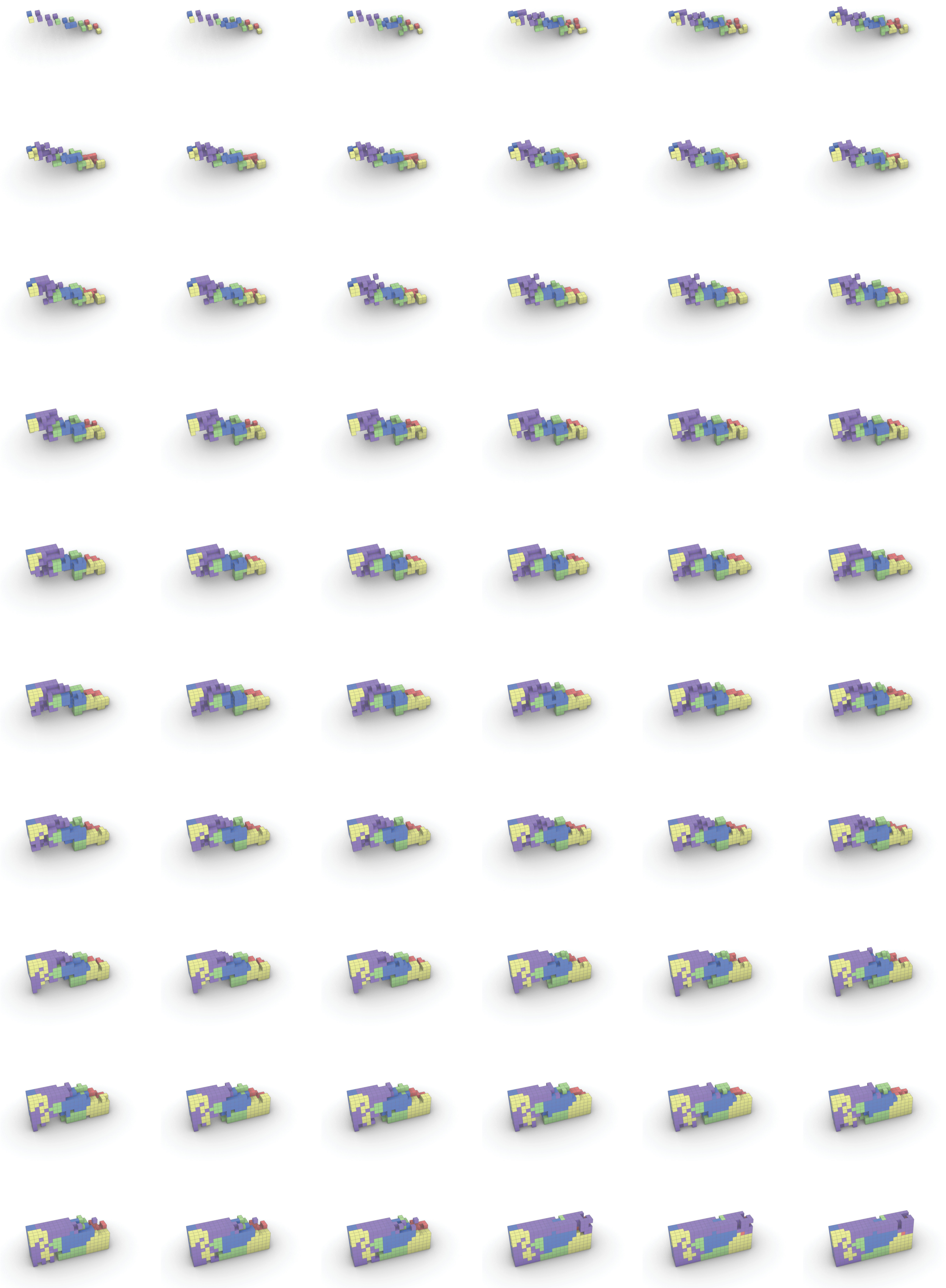


Figure 7.47: Linear Assignment Toy Problem Simulation images

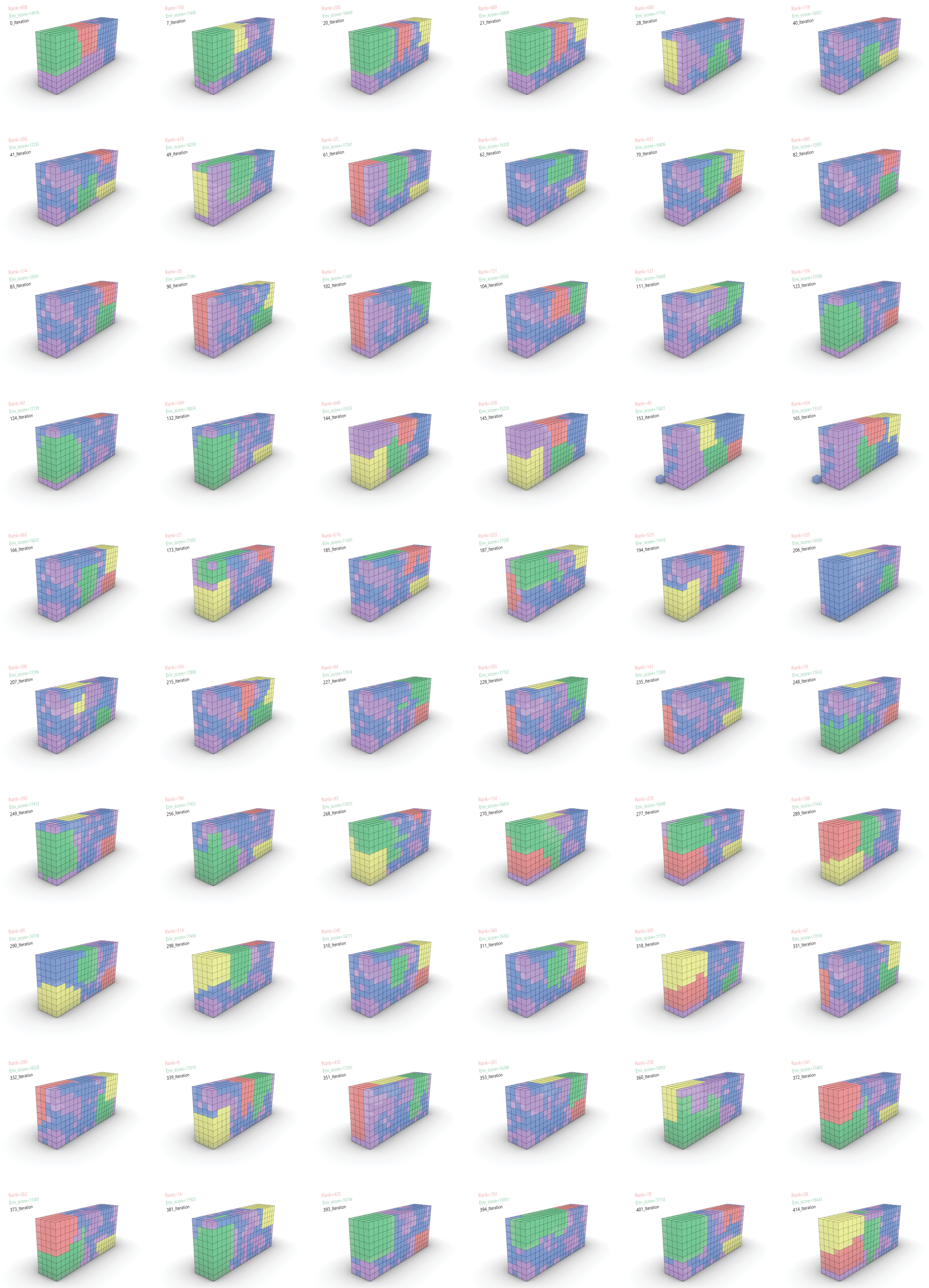
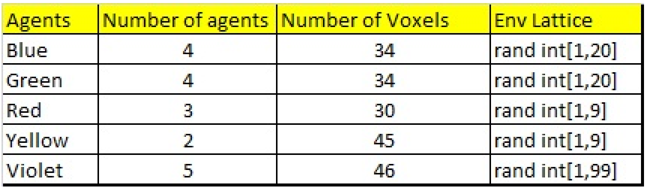


Figure 7.48: Linear Assignment all permutation performances good variants

would make the benefit more predictable and improving the validity of the approach.

7.0.23 Improved simulation:

In the improved simulation the number of agents which are to be assigned was increased to a minimum of two. So considering the smallest agent in the problem which is the Red and Yellow both having to occupy 91 voxels the minimum size comes out to be around 45 voxels. So if all the agent requirements are broken down into 45 voxels the number of locations would be around 15.2 so the next perfect divisible number to it (18) was considered.The benefit matrix was calculated for 18 positions inside the base lattice and 18 agents would be assigned in the simulation. The modified simulation table looks as follows:



Agents	Number of agents	Number of Voxels	Env Lattice
Blue	4	34	rand int[1,20]
Green	4	34	rand int[1,20]
Red	3	30	rand int[1,9]
Yellow	2	45	rand int[1,9]
Violet	5	46	rand int[1,99]

Figure 7.49: Modified Toy problem

After running the simulation for this modified set of problem the output or the total benefit from the simulation came out to be 17360 which comes close to the 210th rank out of 720 ranks in the previous simulation. This was a huge improvement from the 678th rank in the first simulation. The simulation results was further improved when the lattices were not random integers but distance fields from the closeness to the facade and the roof. The final ranking was around 155/720 which is a proof that this approach will generate a good solution for the problem if not the best one.This method can be replicated for the case study where the lattice is very large and the combinatorial complexities are immense.

7.0.24 Conclusions:

If the linear assignment approach is to be considered then the resolution of the simulation, that is the number of agents should be large enough so that the size of the agents is more or less similar. This will generate accurate results since the impact of one agents growth over the other is reduced. Secondly the environment lattices should not consist of random integers but should contain properly simulated lattices so that there is a certain relationship between the adjacent voxels.

7.0.25 Limitations:

The first limitation is that such an approach is not suitable for zoning where there is a closeness requirement between the zones.In this particular case of mixed-use typology where there wasn't any specific closeness requirement this method can be implemented to find the optimal zoning. Secondly,the reduction of the search filed space for assigning the agent origins is a heuristic applied to this specific case for reducing the search space and might not result in the best possible outcome for the optimization. That said at the stage of zoning the deviation from the optimum or the error is acceptable enough to make a decision on the outcome.Finally the output from the approach when the agent sizes are made small can at times become a bit staggered with isolated growths in the lattice. So the topological quality of the zones created might necessarily not be that great and an improvement step needs to be defined.

7.0.26 2.Sequential Assignment approach:

In the sequential assignment approach the approach is more towards replicating the manual process with the intelligence of multi criteria assignment and form evolution offered by the agent based model.In this approach the functions in the space program are assigned sequentially from the biggest function to the smallest function. This approach gives priority according to the size assuming that the impact of the biggest function is going to be the most on the output of the benefit calculations of the zoning.

The similarity of the approach to the manual process can be attributed to the fact that in the manual process multiple zones are never assigned its always a sequential process with a explicit or an assumed priority order of the various zones. The main benefit of this approach is the topological quality of the zones created in terms of their shape and island formation. Since the growth of the agents is not affected by the growth of their surrounding agents the agents can grow in the exact way that they are programmed in response to the environment lattice values which they are fed with.The interesting thing to observe in this approach would be the final score of the simulation in comparison to the Linear assignment approach. In this type of an approach the lower the function or the agent is in the priority list the more it has to compromise in the quality of voxels which it can occupy. In that sense this approach does not give equal access to all the functions.

Algorithm 10 Procedure for Sequential assignment approach

```
procedure INITIALIZATION
    Load base lattice
    Generate environment lattices
procedure GENERATE THE POSSIBLE ORIGIN POINTS
    Extract the indices of all available voxels
procedure GENERATE THE BENEFIT ORDER
    Run the ABM simulations on all origins and calculate the benefits of assignment at each location
procedure CREATE MULTI-AGENT SYSTEMS
    Create the necessary stencils
    Create the necessary Agent class
    Create the necessary Environment class
procedure SEQUENTIAL ASSIGNMENT TO GENERATE THE ZONING
    Run the ABM occupation simulation for the best location of the biggest agent
    Find the best location from the remainder voxels for the next agent according to size from the benefit list generated before.
    Run the ABM simulation on the selected origin and continue this process till voxels are occupied.
procedure CALCULATE TOTAL BENEFIT
    Extract the indexes from each agent from their respective value lattices .
    Aggregate the values from each agent and generate the total sum of benefit from all the agents.
```

7.0.27 Agent based simulation procedure:

The same problem formulation and objective is considered as described in the previous method in 7.0.19. The following steps are taken in the Agent based simulation procedure:
Step1: Determine the possible agent origin locations:
In this approach all the possible available voxels in the lattice are the agent origins. So the first step involves extracting the 2D indices of all the available voxels.

Step2: Find the benefit of simulation at all locations:
In all the possible agent origins the simulation is run for the desired number of voxels for all the agents and the benefit obtained for each origin location is calculated. The origin list is sorted according to the benefit from the highest to the lowest.

Step3: Run Sequential Agent based simulations:
According to the ascending order of the size of the agents the agent based simulations are run sequentially.For the first simulation the best agent origin location for violet is choosen and the simulation is run, then from the remaining locations which are available in the lattice the best location is choosen and the simulation for the next agent is run. The process continues till all the agents have occupied their required voxels.

Step4: Calculation of benefit for each agent and the total benefit:
The occupied voxels for each agent is traced to its value lattice in the environment and the corresponding environment values are retrieved.These retrieved values are summed up to generate the total score for the agent. The total score for all the agents is the total score for the simulation.

7.0.28 Results:

The total benefit calculated from the simulation was around 16548 which was worse in comparison to the linear assignment method. The ranking would be close to 516/720 when compared to the 720 simulations run in the validation of the linear assignment approach [7.0.21]

7.0.29 Conclusions:

The method itself has the quality of not giving access to all the agents equally so the total benefit obtained was going to be poorer in comparison to the linear assignment approach. However there are still possibilities of improving this simulation. The first approach towards achieving this would be to inspect for emergent patterns in the value lattice of the zoning model. If multiple clusters can be observed for a certain agent then the agent should be split up into several different agents which can simultaneously acquire good performing voxels in a larger search space.This will ensure that the simulations are run sequentially but can potentially achieve better results.

7.0.30 Brute Force method:

Another method which was tested on the Toy problem for zoning was the brute force method.In this method the first step included generating all possible permutations of the agent origin combinations like the 720 done in the first attempt of linear assignment approach. After the permutations are generated this method utilizes the brute force

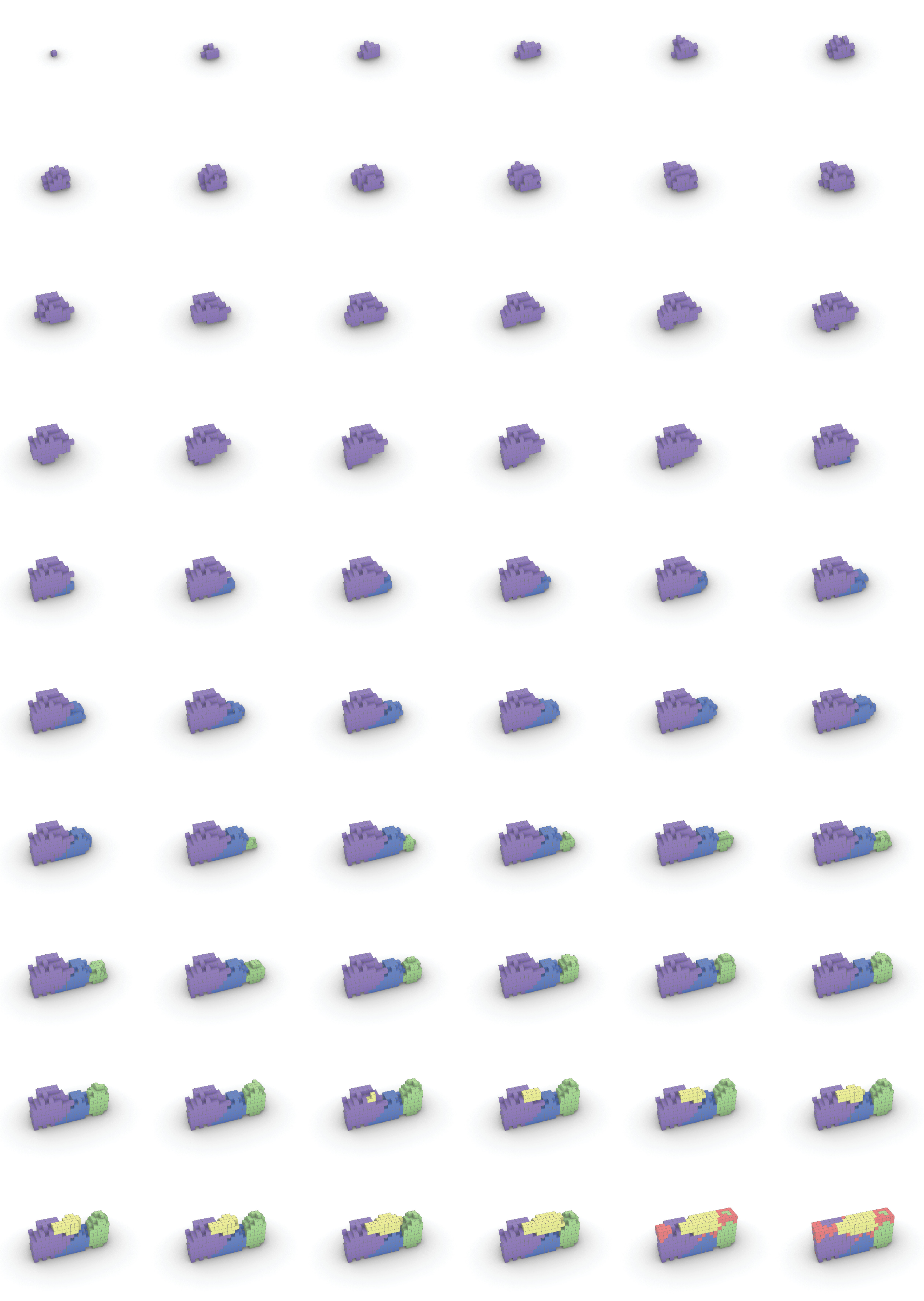


Figure 7.50: Sequential Assignment process for Toy Problem

to compute all the permutations and generate results. Finally the best among them is picked as the chosen option.The image [7.48]

This method indeed has its benefit that the best possible combination is guaranteed for the given set of points, but the obvious drawback being the computational time and power required for it. For a Toy problem with such a small size and limited combinatorial possibilities it was possible to use brute force to compute but for even a slightly larger problem with higher complexity it becomes a tedious and impossible task to calculate all the possible combinations.

This method is effective for validation of results as seen in [7.0.21] where for a limited set of combinations iterating through brute force can help in generating a large set of simulation values to compare the results and generate a validation result for the method under testing.

7.0.31 Evaluation and comparison of the methods explored:

From among the three methods explored for the zoning problem the performance of the methods needs to be evaluated. The comparison done in the results of both the Linear assignment method and the Sequential assignment method to the 720 results obtained by using brute force on a certain list of combinations is not an appropriate one. The main reason for this being the small size of the sample points considered for this evaluation. 720 permutations are generated by assuming 6 possible positions for 5 agents.The number of possibilities should be very extensive to get the best possible output.

In order to perform a complete evaluation the following methodology is used.
Step1: Extract all available indices:
The first step of the process involves extracting all possible one dimensional indices from the occupation lattice and load all the environment value lattices.

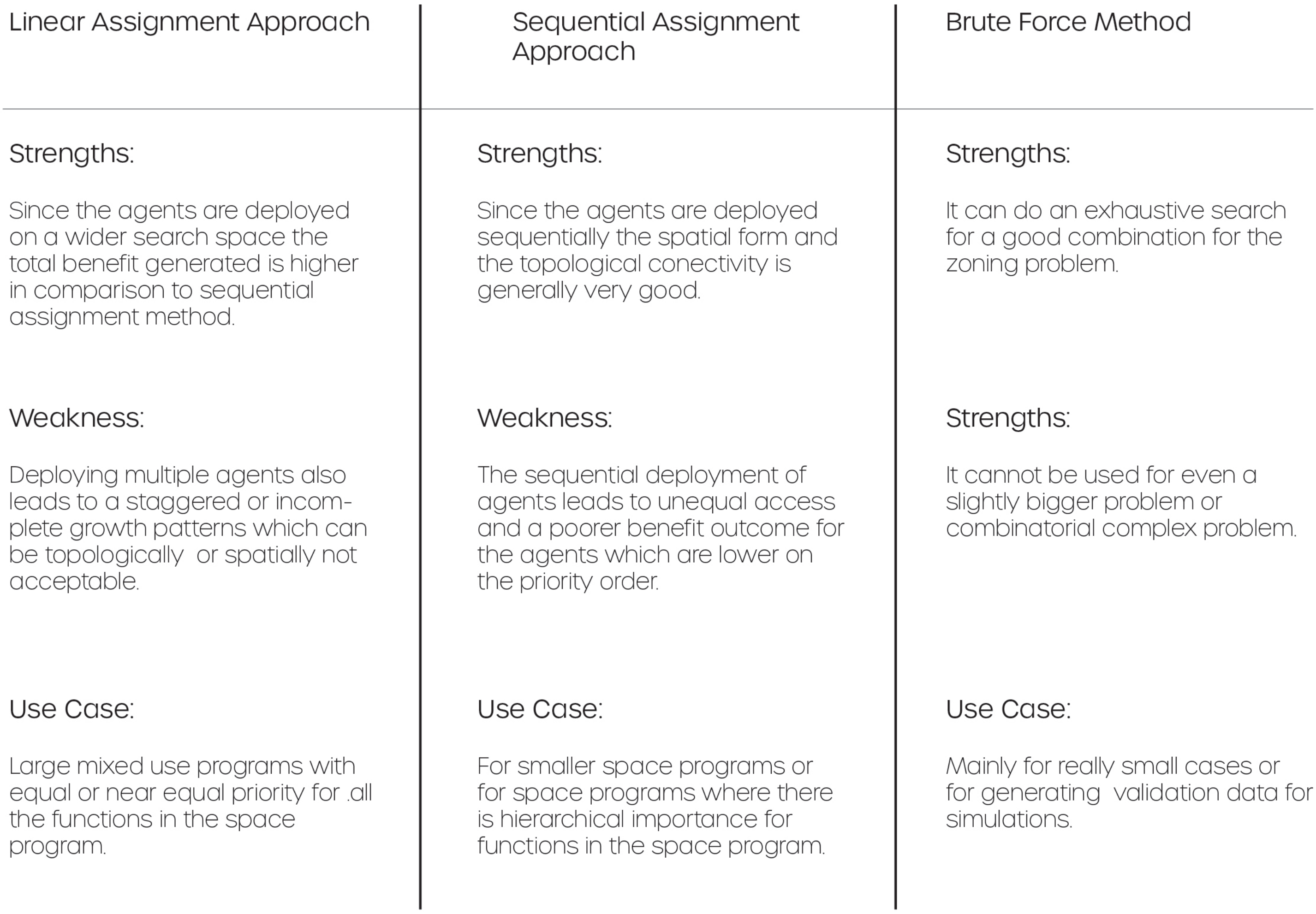
Step2: Develop the necessary ABM Simulation elements:
All the necessary elements in a Agent based simulation as done in the previous simulations need to be setup from agent class to stencils.

Step3: Generate the Benefit matrix for all positions:
For each agent run the ABM simulation using 3d circular or cuboidal behaviour[7.0.11] (assuming a 3d behaviour requirement) for all the possible origin points. Extract the values from the value lattice for all the simulations and list the aggregated values for the same. These aggregated values can be sorted to create a benefit matrix for the agent for all locations.From the benefit matrix the minimum value and the maximum value can be extracted.This same process should be continued for all the agents in the simulation.

Step4: Generate the performance index:
In the next step from each of the methods from the final output the values from the value lattices for each of the agents can be extracted based on the indices in the occupation lattice and the total aggregated value for each agent is calculated. This value is remapped to a number from 1 to 10 considering the original domain from the benefit matrix.This will give the performance indicator for the agent and the method. In order to calculate the complete performance of the method weights are considered for the individual performance indicators. The weights in this case are the percentage of voxels occupied by the agent from the total requirement. This weights when applied will generate a weighted list of performance indicators for each agent which can be aggregated to generate the total performance of the simulation.

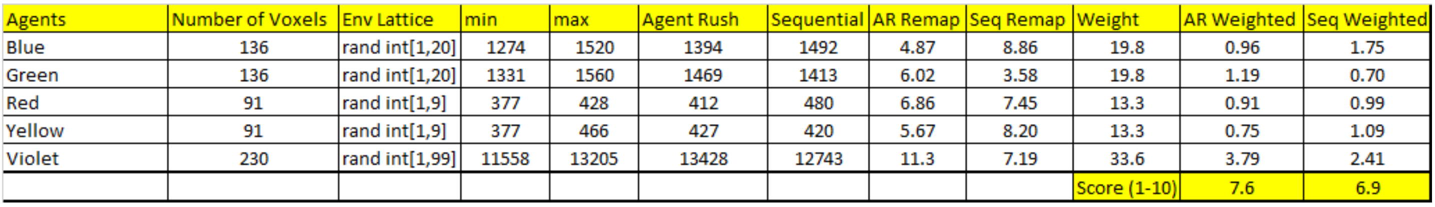
Algorithm 11 Procedure for Evaluation

```
procedure GENERATE THE POSSIBLE ORIGIN POINTS
    Extract the indices of all available voxels
procedure GENERATE THE BENEFIT ORDER
    Run the ABM simulations on all origins and calculate the benefits of assignment at each location
procedure CREATE MULTI-AGENT SYSTEMS
    Create the necessary stencils
    Create the necessary Agent class
    Create the necessary Environment class
procedure GENERATE THE PERFORMANCE INDICATORS
    Generate the min and the max from the benefit matrix
    Remap the aggregated simulation benefit calculation for an agent into a desired domain (for ex. 1-10)    ▷ This is the metric for individual agent
    Generate weights for all the agent indicators
    Fom the weighted product model generate the final performance index for the whole simulation
```



Linear Assignment Approach	Sequential Assignment Approach	Brute Force Method
Strengths: Since the agents are deployed on a wider search space the total benefit generated is higher in comparison to sequential assignment method.	Strengths: Since the agents are deployed sequentially the spatial form and the topological conectivity is generally very good.	Strengths: It can do an exhaustive search for a good combination for the zoning problem.
Weakness: Deploying multiple agents also leads to a staggered or incomplete growth patterns which can be topologically or spatially not acceptable.	Weakness: The sequential deployment of agents leads to unequal access and a poorer benefit outcome for the agents which are lower on the priority order.	Strengths: It cannot be used for even a slightly bigger problem or combinatorial complex problem.
Use Case: Large mixed use programs with equal or near equal priority for .all the functions in the space program.	Use Case: For smaller space programs or for space programs where there is hierarchical importance for functions in the space program.	Use Case: Mainly for really small cases or for generating validation data for simulations.

Figure 7.51: Comparing the methods used for Zoning problem



Agents	Number of Voxels	Env Lattice	min	max	Agent Rush	Sequential	AR Remap	Seq Remap	Weight	AR Weighted	Seq Weighted
Blue	136	rand int[1,20]	1274	1520	1394	1492	4.87	8.86	19.8	0.96	1.75
Green	136	rand int[1,20]	1331	1560	1469	1413	6.02	3.58	19.8	1.19	0.70
Red	91	rand int[1,9]	377	428	412	480	6.86	7.45	13.3	0.91	0.99
Yellow	91	rand int[1,9]	377	466	427	420	5.67	8.20	13.3	0.75	1.09
Violet	230	rand int[1,99]	11558	13205	13428	12743	11.3	7.19	33.6	3.79	2.41
									Score (1-10)	7.6	6.9

Figure 7.52: Performance indexing for the toy problem

The table [7.52] show the comparison of the two methods and the performance indicators generated using the method described in the previous paragraph. The linear assignment method works better than the sequential assignment method for the reasons stated before.

The method for evaluation is also highly dependant on the choices made for the agent performance. If a lot of agents have similar choices then the maximum performance which can be achieved can be lower. The performance indicator portrays the degree of compliance/compromise made by the agents while solving the zoning problem. This compliance/compromise is with respect to the decisions made for generating the desirability matrix for the agents. the better spread out the matrices for the various agents the better will be the compliance.

7.0.32 3. Total closeness rating method:

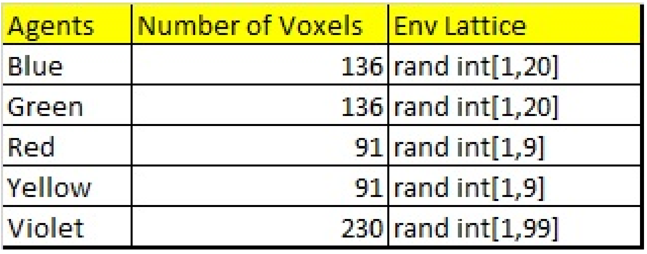
The zoning problem can be classified into two types: problems with closeness requirements and the problems without the closeness requirement, as described in [4.2]. In the case for Buiksloterham the space program is such that there is no closeness requirements of the zones with each other but for many other examples like hospitals closeness requirement for the zones becomes the critical parameter for analyzing the performance of the zoning solution.

The approach considered for the zoning problem with closeness requirements is inspired from the algorithms like the CORELEAP and the ALDEAP [3.1.9] developed to solve the facility layout problem in industrial engineering which are based on the concept of TCR or total closeness rating.

In order to define the closeness relationships a REL chart (Relationship Diagram) is made for the various zones according to the space program describing the closeness requirements of the zones with respect to each other.The closeness is classified into six categories :***A:****Absolutely Necessary,* ***E:****Especially Important,* ***I:****Important,* ***O:****Ordinary closeness,* ***U:****Unnecessary closeness,* ***X:****Avoid closeness* .

7.0.33 Toy problem formulation:

The following details are considered for the Toy problem.
Size of the Lattice = 19x9x4 (LxHxW)[684]
Number of Agents: 5
Agents: (Blue , Green , Yellow , Red, Violet)



Agents	Number of Voxels	Env Lattice
Blue	136	rand int[1,20]
Green	136	rand int[1,20]
Red	91	rand int[1,9]
Yellow	91	rand int[1,9]
Violet	230	rand int[1,99]

Figure 7.53: Toy Problem Details

The above problem is the same with an addition of closeness requirement of the zones with each other given by the REL Chart below:

$$\begin{array}{cccccc} & a_B & a_G & a_Y & a_R & a_V \\ a_B & nil & A & O & E & O \\ a_G & - & nil & O & O & X \\ a_Y & - & - & nil & O & O \\ a_R & - & - & - & nil & A \\ a_V & - & - & - & - & nil \end{array}$$

The values corresponding to the closeness described in the chart are as follows:***A:(4)****Absolutely Necessary,* ***E:(3)****Especially Important,* ***I:(2)****Important,* ***O:(0)****Ordinary closeness,* ***U:(1)****Unnecessary closeness,* ***X:(-1)****Avoid closeness* .

$$\begin{array}{ccccccc} & a_B & a_G & a_Y & a_R & a_V & TCR \\ a_B & nil & 4 & 0 & 3 & 0 & = 7 \\ a_G & 4 & nil & 0 & 0 & -1 & = 3 \\ a_Y & 0 & 0 & nil & 0 & 0 & = 0 \\ a_R & 3 & 0 & 0 & nil & 4 & = 7 \\ a_V & 0 & -1 & 0 & 4 & nil & = 3 \end{array}$$

Calculations for the total closeness rating (TCR) would be the sum of the relationships of the agents with all other agents given by the matrix above. The TCR method can have two approaches for generating a solution. The first approach is the constraint based approach where the rel-chart is converted into conditional statements for closeness of agents with respect to one another. This approach is suitable for voxelised grids where the growth of agents is not hampered by the form of the massing model and the size of the agent is predictable as seen in figure [7.54].The radius for the agents is calculated based on the number of voxels it needs to occupy and that is substituted in the rel-chart with an addition or subtraction of values to assure the condition of adjacent or disconnected for the extreme relations like avoid and absolutely necessary.

$$\begin{array}{cccccc}
 & a_B & a_G & a_Y & a_R & a_V \\
a_B & nil & 7 & X & 10 & X \\
a_G & 7 & nil & X & X & 11 \\
a_Y & 9 & 8 & nil & X & 10 \\
a_R & 8 & 8 & X & nil & 6 \\
a_V & 12 & 9 & 11 & X & nil
\end{array}$$

In the above matrix for a_Y from a_V +3 was added to the radius of Y considering the value for the avoid closeness relation thus ensuring that the agents wont be close to each other.

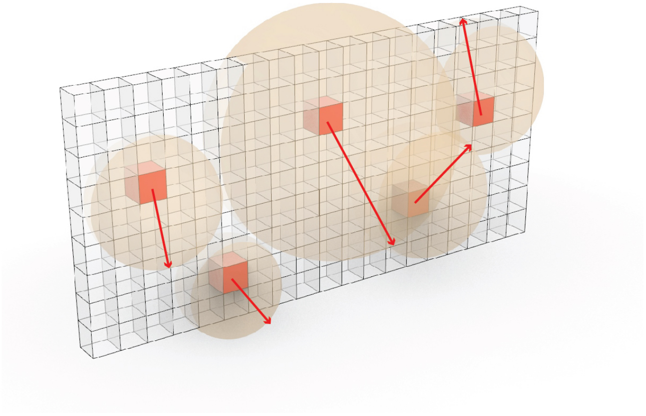


Figure 7.54: Growth radius for the agents assuming a perfect cuboidal or spherical behaviour

The Second approach is more inspired from the facility layout techniques where the total closeness requirement is calculated first. Based on the results of the calculation the agents are deployed sequentially so that the closeness constraints are met starting fro the agent having the maximum closeness rating.This approach is similar to the CORELEAP algorithm [3.9] but considered in three dimensions.The procedure for both the approaches is described in the following section.

Objective: To find the best locations which satisfy the closeness constraints for the origin of five agents Blue, Green ,Yellow, Red and Violet so as to maximise the total benefit which is the summation of all the values gained by the agents from their respective enviornment fields during the simulation.

7.0.34 Agent based Simulation procedure for the constraint based TCR approach:

Step1: Determine the possible agent origin locations
The middle section of the lattice was parsed for possible positions for the agent origins based on a set interval producing a range of points where the agents could be assigned. I the case of the Toy problem 36 points were generated in comparison to the 6 points in the previous approaches.The primary reason for parsing more points was to have a good sample size for closeness constraint checking.

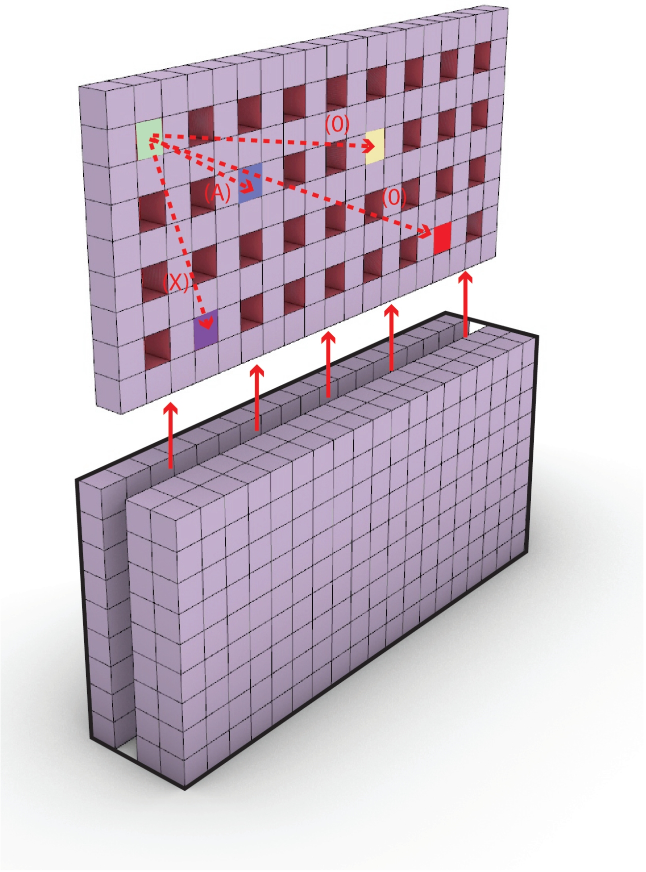


Figure 7.55: Toy Problem agent origin locations valid permutations

Step2: Determine all the valid agent origin permutations:
The relationships from the rel-chart as defined in the problem formulation are converted in to constraints and all the possible permutation of the 5 origins form the set of 36 origins are computed in this step. This ensures the validity of the zoning output at least in terms of the closeness requirements.

Step3: Generating the benefit matrix for all the permutations
The benefit matrix is computed for all agents at all the locations from the valid permutations defined in the previous step. The benefit matrix is recorded and is further used in the linear assignment process.
Step4: Linear Assignment Process
The linear assignment problem is formulated and solved using the MIP solver from the Google OR

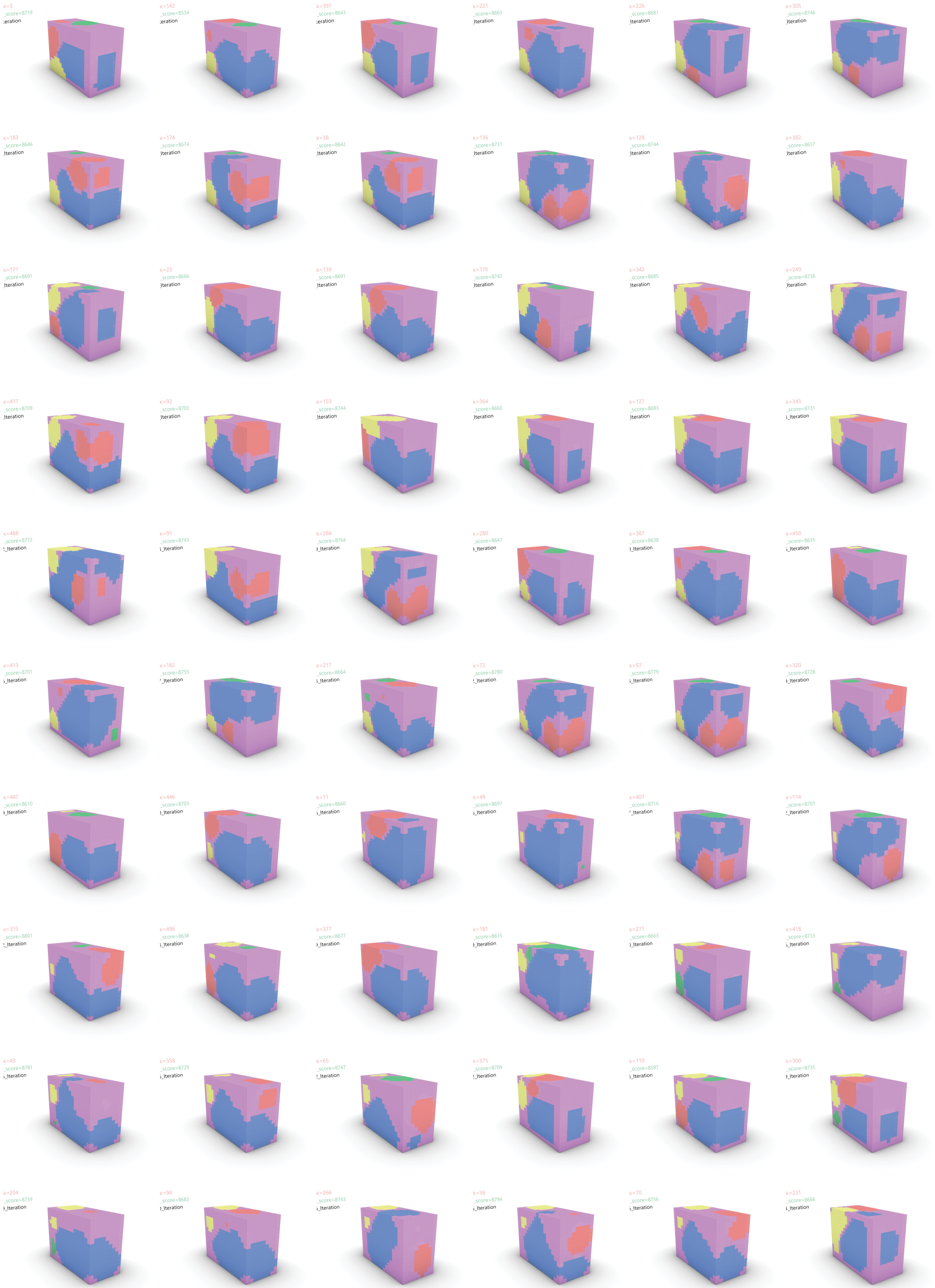

k=3 _score=8719 :eration
k=142 _score=8534 :eration
k=397 _score=8643 :eration
k=221 _score=8663 Iteration
k=226 _score=8681 Iteration
k=305 _score=8746 Iteration
k=183 _score=8646 Iteration
k=174 _score=8674 Iteration
k=38 _score=8642 Iteration
k=136 _score=8731 Iteration
k=128 _score=8744 Iteration
k=382 _score=8657 Iteration
k=171 _score=8691 Iteration
k=23 _score=8666 Iteration
k=139 _score=8691 Iteration
k=170 _score=8742 Iteration
k=342 _score=8685 Iteration
k=249 _score=8738 Iteration
k=417 _score=8709 Iteration
k=92 _score=8703 Iteration
k=153 _score=8744 Iteration
k=364 _score=8660 Iteration
k=127 _score=8683 Iteration
k=343 _score=8731 Iteration
k=488 _score=8772 Iteration
k=91 _score=8743 Iteration
k=284 _score=8764 Iteration
k=280 _score=8647 Iteration
k=387 _score=8638 Iteration
k=450 _score=8635 Iteration
k=413 _score=8701 Iteration
k=182 _score=8755 Iteration
k=217 _score=8664 Iteration
k=72 _score=8780 Iteration
k=57 _score=8779 Iteration
k=320 _score=8728 Iteration
k=447 _score=8610 Iteration
k=446 _score=8703 Iteration
k=11 _score=8660 Iteration
k=49 _score=8697 Iteration
k=407 _score=8716 Iteration
k=114 _score=8707 Iteration
k=315 _score=8801 Iteration
k=496 _score=8638 Iteration
k=377 _score=8677 Iteration
k=181 _score=8635 Iteration
k=271 _score=8663 Iteration
k=418 _score=8733 Iteration
k=43 _score=8781 Iteration
k=358 _score=8729 Iteration
k=65 _score=8747 Iteration
k=375 _score=8709 Iteration
k=110 _score=8597 Iteration
k=300 _score=8735 Iteration
k=204 _score=8739 Iteration
k=94 _score=8683 Iteration
k=266 _score=8743 Iteration
k=36 _score=8794 Iteration
k=70 _score=8756 Iteration
k=231 _score=8666 Iteration

Figure 7.56: Good performing results from the TCR process for Toy Problem

tools as described in [7.0.20]. This will generate a permutation matrix which will be used to generate the final zoning output and the total benefit from the assignment.

Step4: Validation for the TCR approach
The total benefit from the linear assignment will be compared with all the outputs from all the permutations obtained in the first step to validate the approach in terms of achieving efficient results.

7.0.35 Agent based Simulation procedure for the TCR approach sequential assignment:

Step1: Determine the possible agent origin locations
The middle section of the lattice was parsed for possible positions for the agent origins based on a set interval producing a range of points where the agents could be assigned. I the case of the Toy problem 36 points were generated in comparison to the 6 points in the previous approaches.The primary reason for parsing more points was to have a good sample size for closeness constraint checking.

Step2: Determine the TCR rating for all agents:
The relationships from the rel-chart as defined in the problem formulation are converted into Total closeness ratings as seen in [7.0.33]. The TCR rating will determine the order of assignment of the agents starting with the agent having the highest TCR rating.

Step3: Generating the benefit matrix
The benefit matrix is computed for all agents at all the locations as described from the first step and the origin positions for each agent are sorted according to the benefit.

Step4: Sequential Assignment Process
The agents are sequentially deployed in the simulation based on their TCR rankings and each time a new agent is deployed the best available position from the sorted list of origins from the previous step is considered as a starting point for the agent.
Step4: Validation for the TCR approach
The Zoning output will be checked for the validity in terms of the closeness constraints.The result of the total benefit will also be checked with the results of the simulations run by brute force method for the same problem to check the effectiveness of the method in finding valid solutions.

7.0.36 Results:

The validation of the results becomes a challenge in this method. Since the number of origin positions are increased in comparison to the previous methods the brute force method has to iterate through thousands of valid options to come up with a base for comparison.In order to solve this problem the best 500 iterations for the biggest agent in the simulation are considered for a benchmark by referencing to the list of origins from step 3. This is not the accurate way of comparing or setting a benchmark but it considers the highest impact factor for comparing.

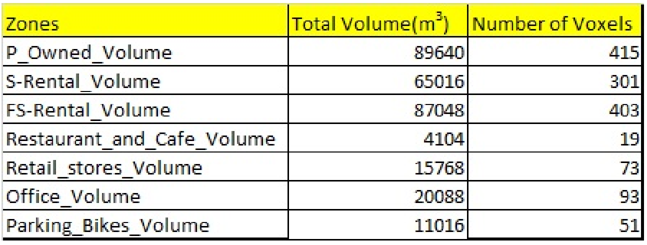
The ranking of the permutation matrix based on the total benefit calculated from the simulation was close to the 120′s range from the top 500 results from all the possible results according to the step one of the process for both the methods of simulation as described before.

7.0.37 Conclusions:

The limitations and the possible improvements for the methods used in this approach are similar to the sequential and the linear assignment approach since this method follows the same steps with added closeness constraints.Improving the Linear assignment and the sequential process will also lead to a better output from the TCR process.

7.0.38 Solving the Zoning problem for the case

The steps taken to interpret the methodology followed for the toy problem to the case is shown in the fig[7.58].As a first step the desirability matrix is generated for each of the zones in the space program as seen in section[7.0.4] by making decisions on the selection and prioritizing the performance factors for each of the zones in the space program. The simplified requirements of the zoning based on the space program as stated in [5.5] are shown in the table below:



Zones	Total Volume(m^3)	Number of Voxels
P_Owned_Volume	89640	415
S-Rental_Volume	65016	301
FS-Rental_Volume	87048	403
Restaurant_and_Cafe_Volume	4104	19
Retail_stores_Volume	15768	73
Office_Volume	20088	93
Parking_Bikes_Volume	11016	51

Figure 7.57: Simplified zoning requirements

The voxelated lattice from the massing problem acts as the occupancy lattice for the ABM simulation.The voxel size considered for the problem is

6x6x6m taking in account the scale of the building and the spaces.These lattices will form the base environment for the ABM simulations further. Certain details regarding the number of agents for each zone their behaviours their origins and ID also need to be defined before the simulations are run.

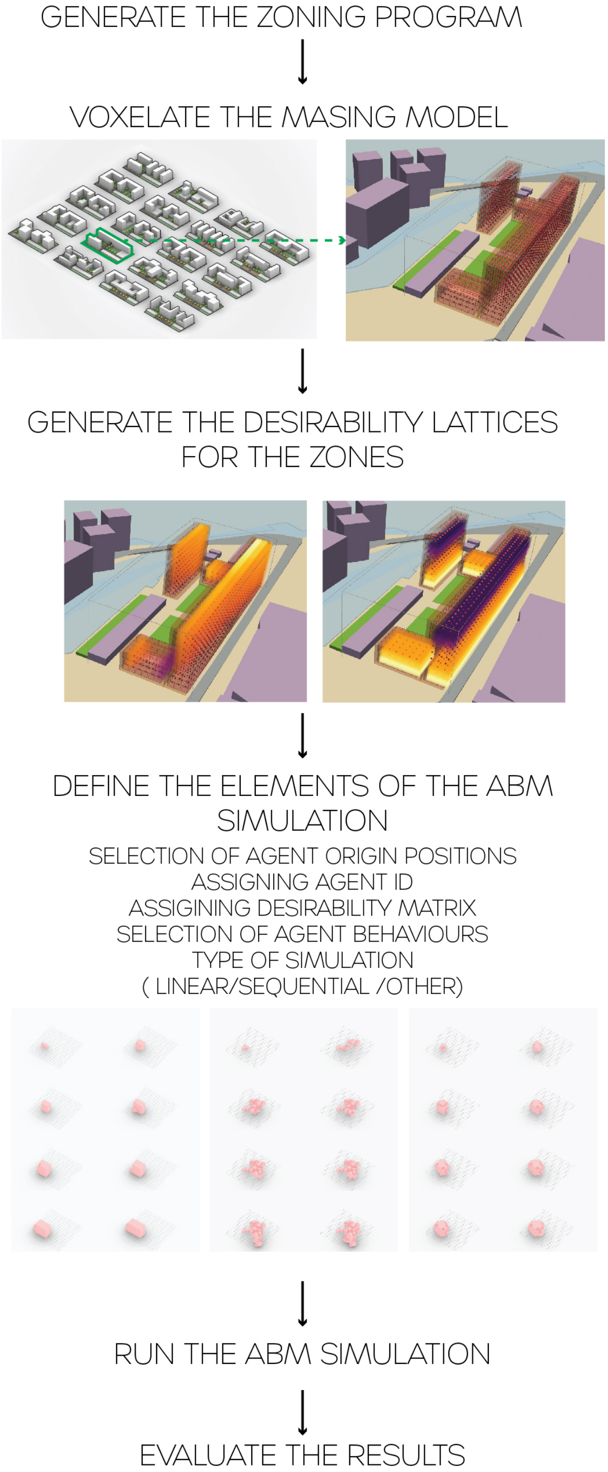

GENERATE THE ZONING PROGRAM
VOXELATE THE MASING MODEL
GENERATE THE DESIRABILITY LATTICES FOR THE ZONES
DEFINE THE ELEMENTS OF THE ABM SIMULATION
SELECTION OF AGENT ORIGIN POSITIONS
ASSIGNING AGENT ID
ASSIGINING DESIRABILITY MATRIX
SELECTION OF AGENT BEHAVIOURS
TYPE OF SIMULATION
(LINEAR/SEQUENTIAL /OTHER)
RUN THE ABM SIMULATION
EVALUATE THE RESULTS

Figure 7.58: The Zoning problem process

Once all the necessary details for the simulation are specified then the method needs to be defined.For the Case at Buiksloterham both the Linear assignment method and the Sequential assignment method will be tested to see the differences in the performance.

7.0.39 Problem formulation for the Case

Size of the Lattice: 6336
Shape of the Lattice: 36 x 16 x 11
Size of Available voxels: 1359

Objective: The main objective of the zoning problem for the case is to locate the zones in the best possible positions in the voxelated lattice based on the desirability matrix generated for each of the zones by the participant.There there are no particular constraints on the closeness of the zones since the elements of the space program are discreet with one another. The requirement for a single island is enforced by the agent behaviour defined in the simulation.

$$TotalBenefit = \sum_{k,i=1}^{7} a_{ki}p_{ki} = V[P] + V[S] + V$$
$$[FS] + V[R] + V[RE] + V[O] + V[PA]$$

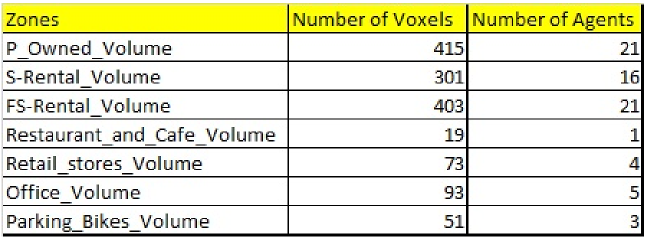
Where:
a$_{-ki}$ = benefit matrix
a$_{-pi}$ = permutation matrix
V = value lattice
P = Indices occupied for Privately Owned Housing Zone
S = Indices occupied for Social Rental Housing Zone
FS = Indices occupied for Free-sector Rental Housing Zone
R = Indices occupied for Restaurants and Cafes Zone
RE = Indices occupied for Retail Zone
O = Indices occupied for Offices Zone
PA = Indices occupied for Parking Zone

7.0.40 Linear Assignment Approach for the Case

The first approach taken to generate a solution for the case is the Linear assignment approach. The following steps were taken to generate the result as seen in [7.0.40]:

Step1: Determine the possible agent origin locations
The smallest zone in the whole program is the Restaurant and Cafe Zone consisting of 19 voxels that need to be occupied. So Considering this as

the benchmark for division of the occupancy lattice all the available cells were divided into clusters of 20 cells to find the positions for 71 possible origin points for the agents.The number of agents for each of the zone was also divided based on this smallest occupancy size of the zone as seen in the table below:



Zones	Number of Voxels	Number of Agents
P_Owned_Volume	415	21
S-Rental_Volume	301	16
FS-Rental_Volume	403	21
Restaurant_and_Cafe_Volume	19	1
Retail_stores_Volume	73	4
Office_Volume	93	5
Parking_Bikes_Volume	51	3

Figure 7.59: Number of Agents for each zone

So each agent in the table has the goal of occupying the best 20 voxels surrounding its origin position.The final output of this step is a list of possible origin locations.

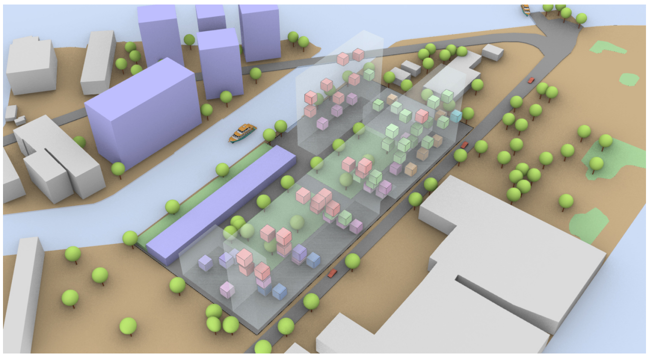


Figure 7.60: 71 possible agent origin locations in the lattice

Step2: Generating the Benefit matrix for the Agents
The ABM simulation for all the position for all the agents is run in isolation and the total benefit generated by the occupying agents at each of the positions is recorded. The environment lattice which is the desirability lattice for the agent is also considered according to its zone. This simulation in isolation will give an idea about the benefit for the particular location for the twenty voxels which each agent has to occupy in the simulation. These recorded benefits are stored in a matrix as the output of this step.

Step3: Linear Assignment Process
The Benefit matrices for the agents are used in the MIP solver with SCIP backend of the Google OR tools to formulate a linear assignment problem where 71 agents have to be assigned to 71 origin positions in the base lattice of the model. The solver runs the calculation and generates the permutation matrix for the assignment problem which acts as the output of this step.

Step4: Running the ABM Simulation according to the Permutation matrix
The permutation matrix is used to run the agent based simulation for all the agents and the total benefit of the whole simulation is calculated by aggregating the occupied values from the desirability matrices of all the agents to calculate the total benefit.

7.0.41 Sequential Assignment Approach for the Case

The second approach taken to generate a solution for the case is the Sequential assignment approach. The following steps were taken to generate the result as seen in [7.0.41]:

Step1: Determine the possible agent origin locations
The first step in the sequential assignment procedure was to parse the base lattice and generate a good sample size for the agent origins. Out of the 1359 available voxels a voxel after every 10 voxels was considered in the list for possible agent origins. This kept the resolution for the simulation higher the intention being to generate more accurate results.

Step2: Separating the lattice into seperate buildings.
The graph seperation was done by using the connected components function from the library networkX [(Hagberg et al., 2008)] to separate the available voxels into individual separate building blocks. This would help in controlling the spread of the zones over all the building blocks.

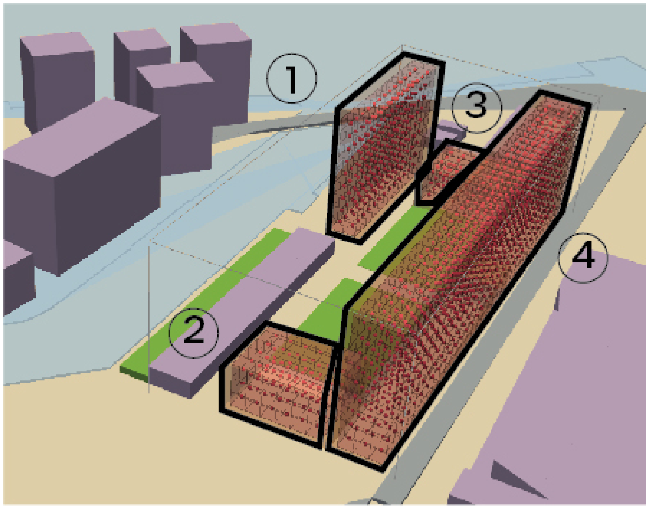


Figure 7.62: Splitting of base lattice into buildings

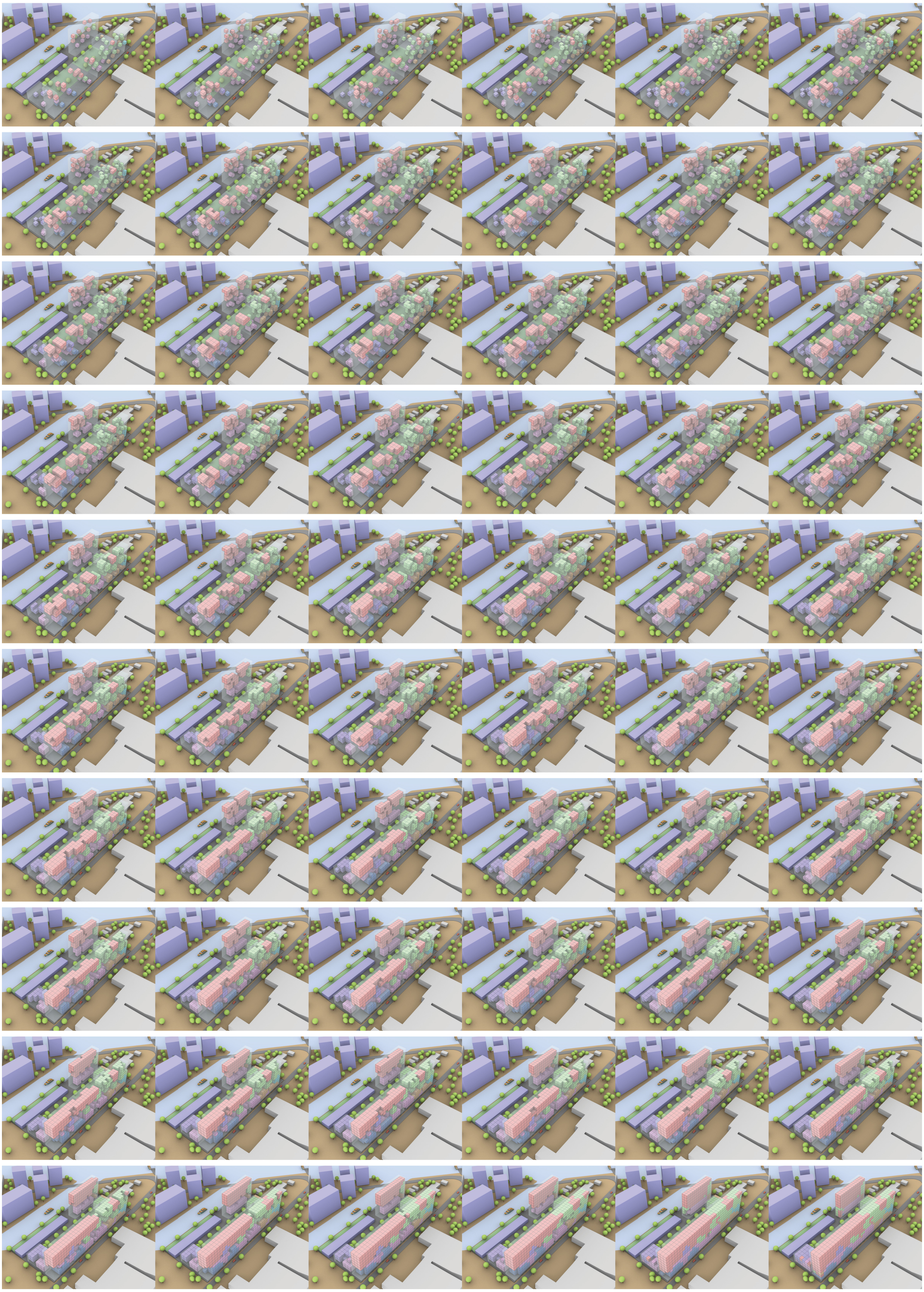


Figure 7.61: Visualizations of the Linear assignment method implementation for the Case

Step3: Generating the Benefit matrix for the Agents

This step is the same as the linear assignment approach second step where ABM simulation for all the position for all the agents is run in isolation and the total benefit generated by the occupying agents at each of the positions is recorded. The environment lattice which is the desirability lattice for the agent is also considered according to its zone.These recorded benefits are stored in a matrix as the output of this step.

Step4: Sequential Assignment Procedure.

The sequential assignment procedure is started at this step beginning with the parking and the Privately owned housing zones. The agent behaviour for parking was a 2d rectangular growth one and for the privately owned housing was 3d cuboidal one considering the nature of the spaces.The agents are initialized in Building one and Building four together to make the zone distributed over the two buildings.

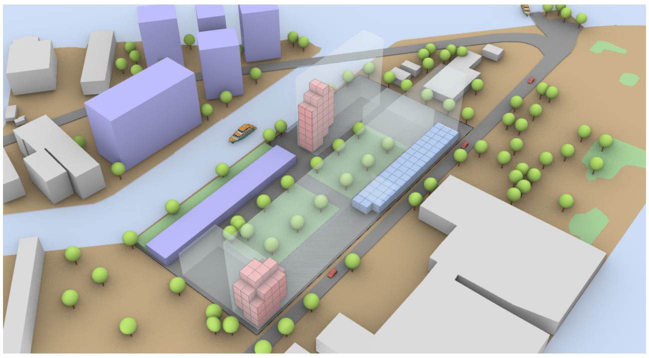


Figure 7.63: First step in assignment of Zones with parking in Blue and Privately owned housing in pink

Once all the voxels for the agents are occupied then the availability of the remaining voxels are scanned for the best possible origin location for the next agent by comparing it with the list generated from step one of the process.Once a suitable origin is found the simulation is run again . This process continues till all the agents have occupied the desired number of voxels.

Step5: Generating the Total Benefit of the simulation:

Once all the agents have occupied the desired number of voxels the desirability values for each one of them are extracted from their respective desirability lattices. The aggregated value for each of the lattice is added to generate the total benefit of the assignment process .

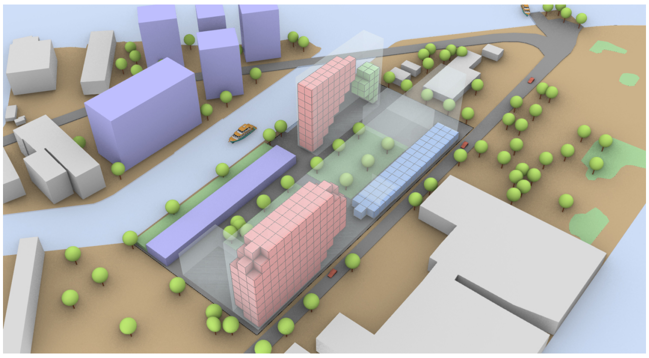


Figure 7.64: Second step in assignment of Zones with Social rental housing zone displayed in Green

7.0.42 Generation of a comparison matrix

The comparison matrix is generated for the case by simulating all the agents on all the locations which are available at the start of the simulation for the occupancy.This Brute force method will generate a comparison matrix for evaluation of the scores obtained by individual agents as well as the cumulative score of each of the methods used.The agent behaviour used for all the agents will be the cuboidal behaviour [7.0.12] and the number of voxels which each agent has to occupy after originating depend on the zone under which it belongs as seen in [7.57]. The aggregated values from the value lattice of the occupied cells for each iteration will be stored as the benefit value which the agent can obtain if it originates at that location.

Algorithm 12 Generation of Comparison matrix

```
procedure EXTRACT ORIGIN POINTS
    available_points= Extract available points from base lattice [value == 1]
procedure GENERATE THE BENEFIT MATRIX
    for point in available_point do
        Create an Agent with the point as an origin point
        Run the ABM using cuboidal occupy behaviour
        Extract the values for the occupied cells from the value lattice
        Aggregate the extracted values
    end
    Repeat this procedure for all points and all Value Lattices
procedure FIND MIN/MAX FOR ALL THE ZONES
    Find the minimum benefit and the maximum benefit from the output of the previous step for all the zones
```

This process is computationally super heavy but is necessary to evaluate the performance of the

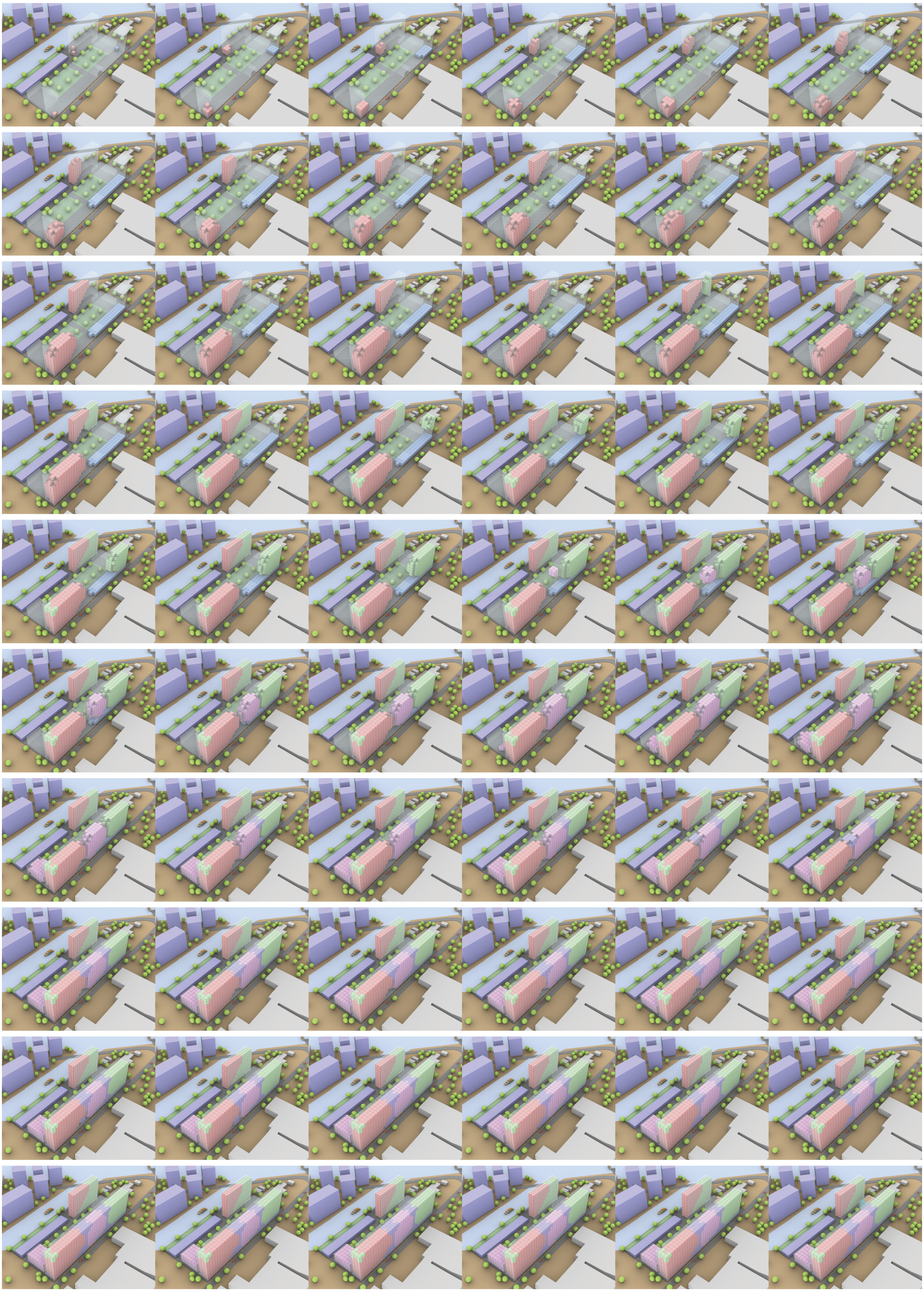


Figure 7.65: Visualizations of the Sequential assignment method implementation for the Case

zoning options which have been made using the methods explained before.

7.0.43 Results

The results for both the type of zoning approaches were calculated according to the method described in the [7.0.31] section. The Linear assignment approach outperformed the Sequential assignment one substantially. The Linear Assignment approach output obtained the score of 9,98 / 10 and the output obtained from the Sequential assignment approach scored 6,24 / 10. This can be attributed to the uniform divisions of the agent occupy goals for the linear assignment one where the influence of one agent growing over the other is reduced substantially and the MIP solver is able to predict accurate results.The sequential assignment method does not give equal access to all the agents to achieve their respective goals with preferences given to the agents assigned first. There are a few drawbacks with the linear assignment process as well.The main drawback of the linear assignment method is the formation of scattered unconnected smaller zones in the lattice. Since the only aim in this approach is to maximise the benefit the topology of the zones formed is not really considered. Isolated small clusters can be seen at a lot of places in the output which need to be consolidated.

One interesting thing which was also observed from the outputs was that at times the assignment benefit in the Linear Assignment method for certain agents at times exceeded the maximum possible value by quite a bit. The primary reason behind this is the assumption of a single island generation for the evaluation matrix and the multiple island approach for the Linear assignment. Due to the removal of restriction of the single island constraint the linear assignment method can effectively scan the search space of the desirability matrix more efficiently than the method used in the calculation of the evaluation matrix. This has its drawbacks as explained in the previous section but nevertheless this has to be considered in the evaluation process.

The possible solution implemented to solve this drawback of the evaluation method is to sort the value lattices that is the desirability lattices for all the zones based on their values. From this sorted list the required number of values can be aggregated base don the zone size to generate the maximum possible value for that zone. This value however will be theoretical since it does not consider the adjacency requirements and the topological requirements of the zones. When compared to this matrix the performance scores of the Linear assignment approach was 7,18/10 and for the Sequential one it was 4,35/10 .

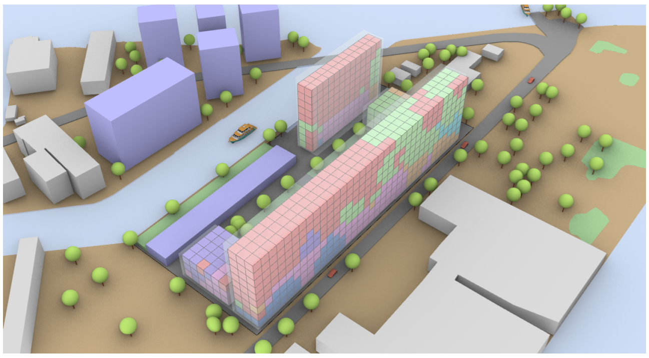


Figure 7.66: Output from the Linear assignment approach

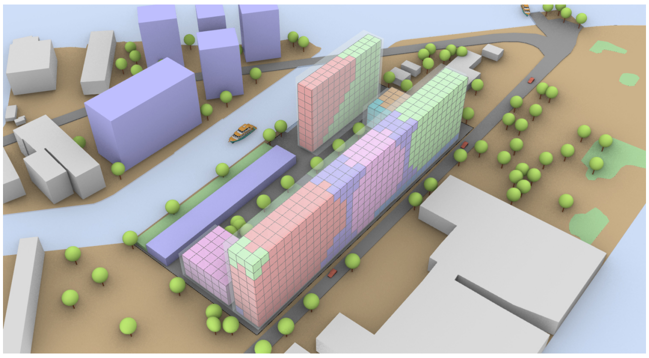


Figure 7.67: Output from the Sequential assignment approach

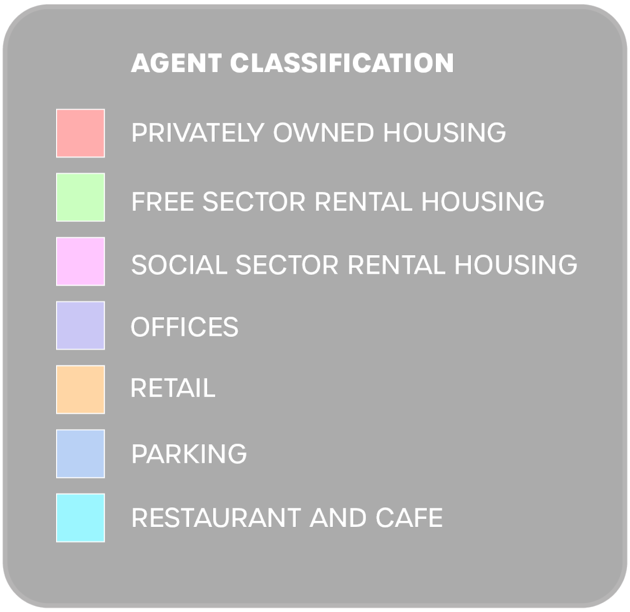


Figure 7.68: Agent Classification and colour legend

7.0.44 Need for an Improvement Algorithm:

The output from the zoning step as seen in [7.66] and [7.67] has problems in terms of Topological

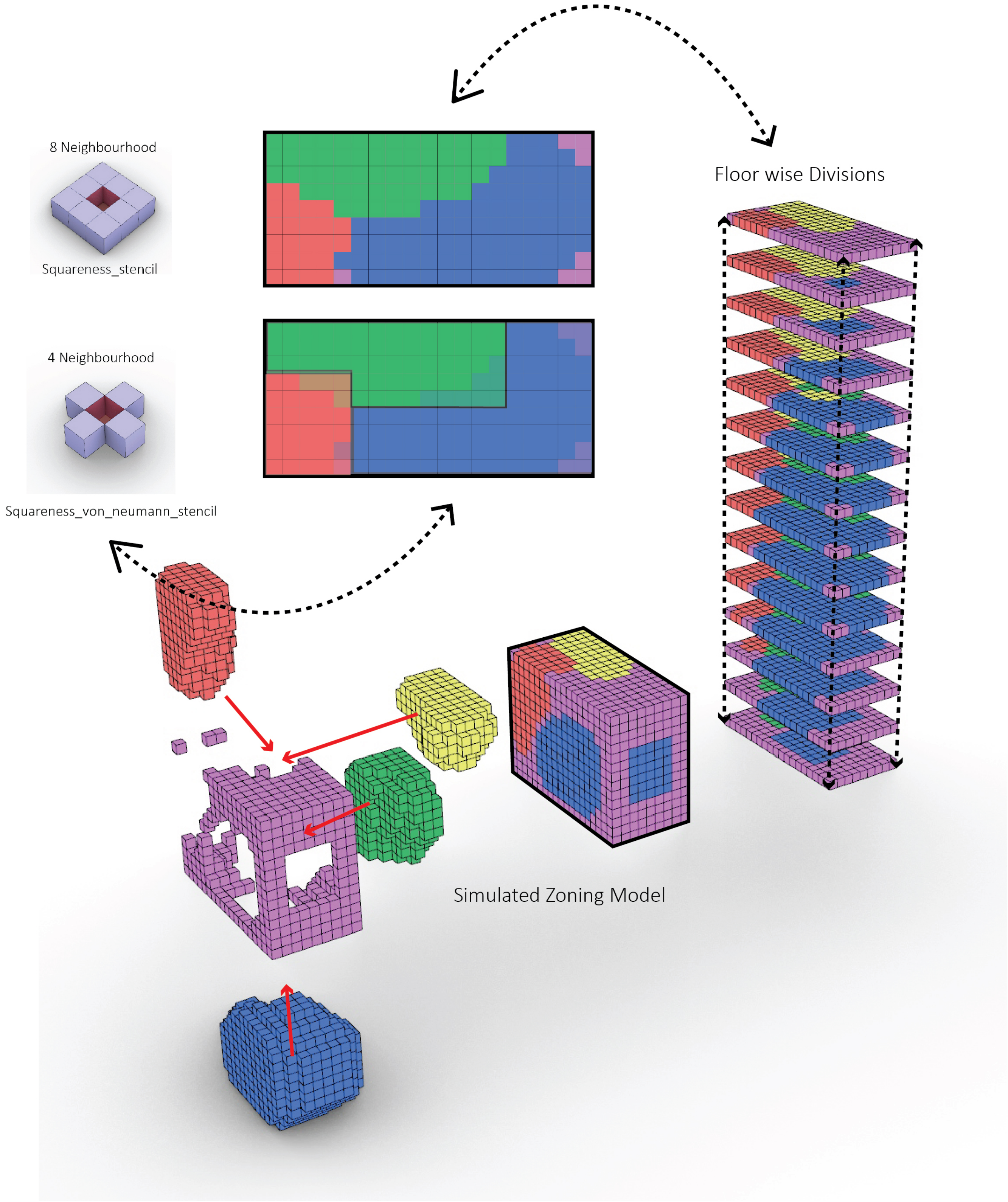


Figure 7.69: Problems of the zoning approach for the Toy problem and possible improvements

characteristics of it. The zones created do not consists of single islands or connected islands. There are isolated growths of zones resulting out of impact of the growth of other agents in the simulation. It can also be seen that the inbuilt character of the agent behaviours to occupy certain shapes are hampered because there is no interaction between the agents in the simulation. There needs to be an addition of a behaviour like negotiation inbuilt in the system where before acquiring a voxel there is a negotiation process happening between the agents where the decision is made on the assignment of the voxel to one of the multiple competing agents. Some of the problems are highlighted in the image. [7.69]

7.0.45 Improvement Algorithm:

In order to fix some of the problems highlighted above an improvement algorithm is proposed. The improvement algorithm tries to improve the output of the zoning problem in terms of its topological character.The following are the steps included in the algorithm:

Step1: Eliminating the isolated growths
The first improvement includes scanning the occupancy lattice for stray growths. This is done by analysing the neighbourhood of all the cells in the occupancy lattice . The stencil used for this is the Cuboidal stencil or the Moore stencil having 26 neighbours [7.0.12]. The threshold for the elimination can be set by the user. If the neighbourhood analysis gives feedback of the neighbour count for a certain section of the zone below the threshold value then the cells are identified as stray growths and are unoccupied in the simulation.

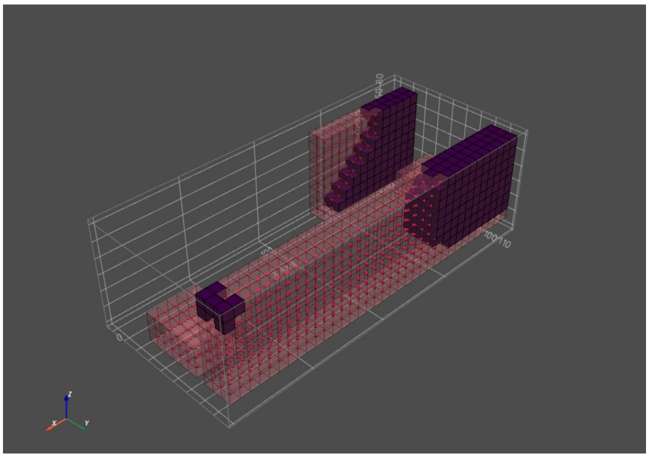


Figure 7.70: Example of Isolated growth from the sequential assignment output for Privately owned offices

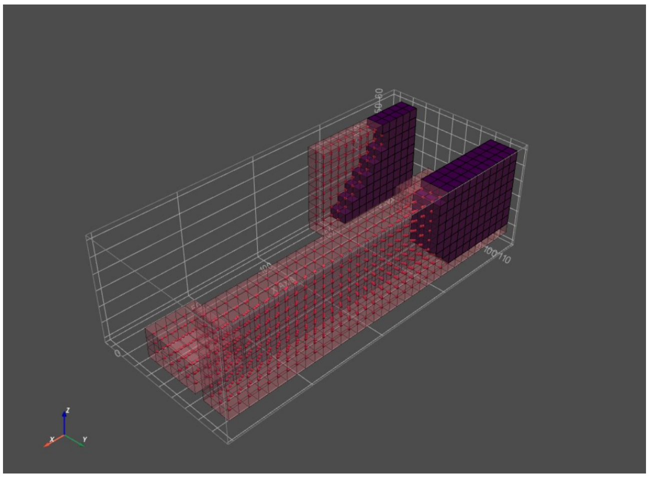


Figure 7.71: Example of improved output from the isolated growth as seen in fig [7.70]

Step2: Improving the 2D Neighbourhood
In this step of the improvement process all the zones are scanned individually in two dimensions using the eight neighbourhood stencil [7.33] on all floors.The isolated single elements in this neighbourhood are identified in the zone by checking through the four iterations of the eight neighbourhood stencil as shown in figure [7.72] where the red voxel indicates the cell under review and the pink ones indicate the half neighbourhoods of the stencil. If there is an isolated voxel with a different id in the half neighbourhood it is unoccupied. This assures of a continuous outline of the zones with no staggered edges.

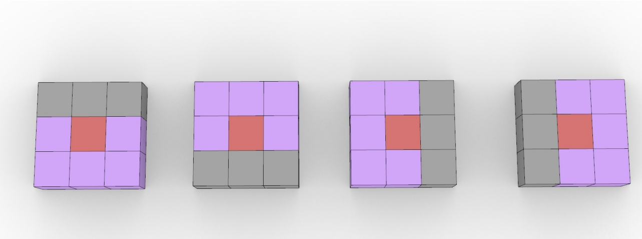


Figure 7.72: Image indicating half neighbourhood sin the eight neighbourhood stencil

Step3: Voxel Recount
Once the Unoccupy behaviour is performed for all the zones the voxel count is done again for all the zones and tallied with the requirements. After the tallying process the new requirement will be generated for the zones depending on the number of voxels which were unoccupied in the zone.

Step4: Voxel Reassignment
The voxel requirement as derived in the previous step is used to assign the empty voxels to the zones which will benefit the most in terms of its topological structure from occupying it. The stencil used in this step is again the 26 neighbourhood

one [7.0.12]. This concludes the improvement process.

There are a lot of limitations to the improvement process as described before.The unoccupy and reassignment will in evidently lead to a dip or a change in the performance of the zoning solution since there is reassignment of voxels to certain zones. This step might prove to be counter intuitive to a process like the Linear assignment one thus hampering with the results. Also the improvement of 2d neighbourhood and the elimination of isolated growth is very user subjective in terms of the type of stencil to be used and the threshold limit for elimination.

7.0.46 Conclusion and Limitations for the Zoning Problem:

The zoning problem is complex in its nature and the selection of the appropriate method is crucial in generating a solution for it. As a first step the nature of the problem needs to be identified in terms of its requirements related to the closeness between the zones.

If there is a closeness requirement then the approaches mentioned in the TCR approach need to be considered and if here isn't a closeness requirement the Sequential approach or the Linear assignment approach can be considered.The Sequential assignment approach has an advantage in generating topologically well formed zones since there is no interference of the growth of other agents but it performs poorly when it comes to achieving the performance goals for each of the zones due to unequal access. The Linear assignment approach performs excellently in achieving the performance goals since it has a finer resolution for growth and scans the environment fields more accurately. However it may perform badly in terms of topological structure since generation of a single island is not under consideration in the method.

One of the biggest drawbacks or limitations in the whole Agent based method is the lack of communication between the agents while performing the behaviour. The whole requirement of an improvement step will be eliminated if the agents would communicate with each other before performing a behaviour against a voxels having shared interests. If this negotiation would happen during the simulation then the agents could grow in the topological form that they were programmed to grow into not affected by the growth of the other agents.

One of the avenues which could be explored to generate good combinations is the use of evolutionary algorithms to search for a suitable permutation matrix for the agent origins. Since the nature of the growth of one agent affecting the other is not easily predictable.

Finally as an output of the whole zoning process the improved simulation version of the Sequential assignment procedure was considered for the next stage of the configuration problem since the improvement algorithm was run on it. The final output after improvement can be seen below :

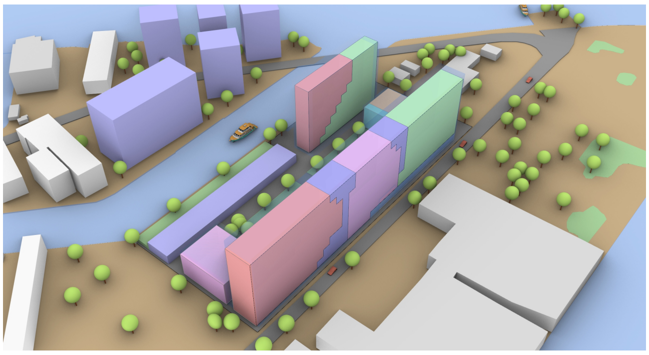


Figure 7.73: Final Result of the Zoning problem

The final output for the zoning has a voxel dimensions of 6x6x6m which is interpolated into a finer resolution of 3x3x3m for the unit assignment problem using the components from the Scipy Library.

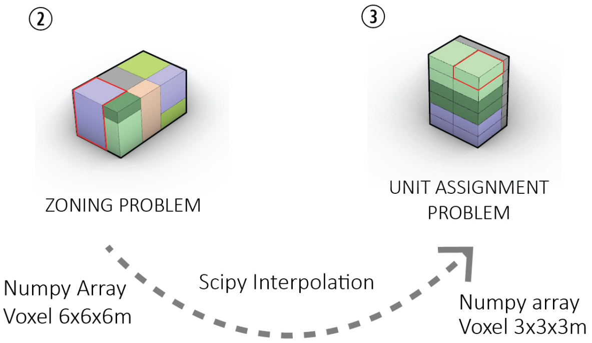
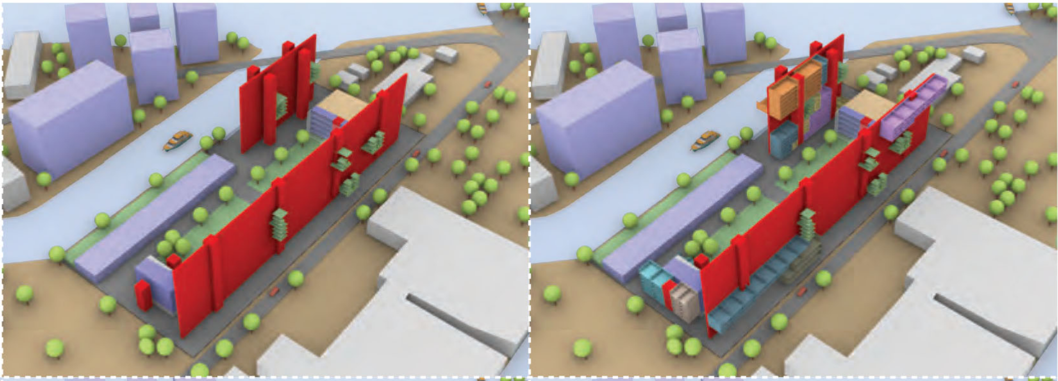
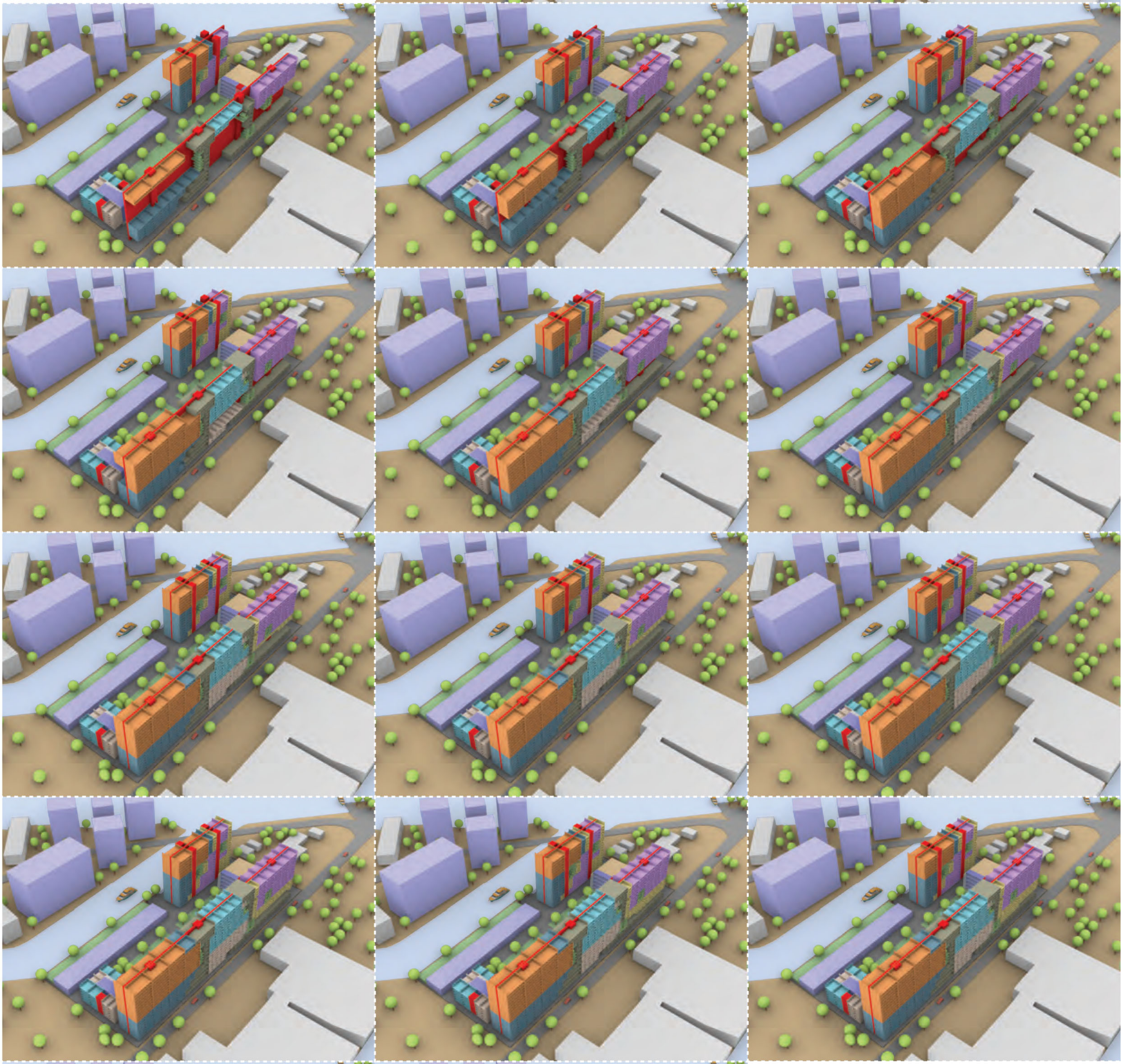
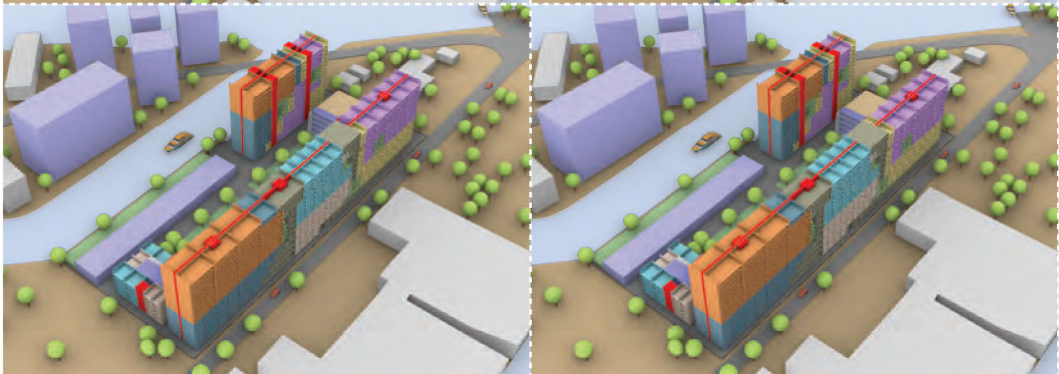


Figure 7.74: Zoning problem to Unit assignment problem

UNIT ASSIGNMENT PROBLEM





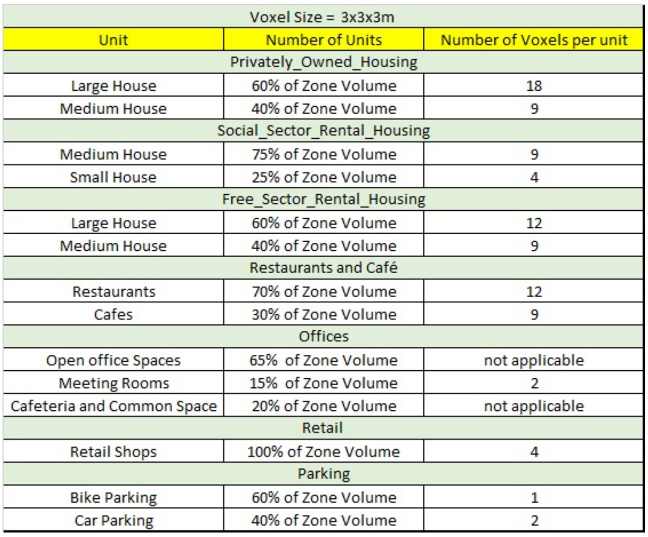


GEN-ARCH

8 Unit Assignment problem

The Unit assignment problem deals with assigning units inside the zones created in the previous layer of the configuration problem. Unit can be defined as an entity in the space program which comprises of a self contained set of rooms collectively catering to a common function. A three room apartment in a zone dedicated for an apartment complex is example of unit inside a zone. Another example could be a department inside a zone in a hospital.A Single restaurant in a zone dedicated for Restaurants and Cafe would also be considered as a unit inside a zone.

In the case for Buiksloterham the following units need to be assigned in the zoning output :



Voxel Size = 3x3x3m		
Unit	Number of Units	Number of Voxels per unit
Privately_Owned_Housing		
Large House	60% of Zone Volume	18
Medium House	40% of Zone Volume	9
Social_Sector_Rental_Housing		
Medium House	75% of Zone Volume	9
Small House	25% of Zone Volume	4
Free_Sector_Rental_Housing		
Large House	60% of Zone Volume	12
Medium House	40% of Zone Volume	9
Restaurants and Café		
Restaurants	70% of Zone Volume	12
Cafes	30% of Zone Volume	9
Offices		
Open office Spaces	65% of Zone Volume	not applicable
Meeting Rooms	15% of Zone Volume	2
Cafeteria and Common Space	20% of Zone Volume	not applicable
Retail		
Retail Shops	100% of Zone Volume	4
Parking		
Bike Parking	60% of Zone Volume	1
Car Parking	40% of Zone Volume	2

Figure 8.1: Unit Assignment Problem Details

8.0.1 Aim:

The Unit Assignment problem can be classified into two categories:
The first category is the zoning problem where there is no closeness relationship between the units of the zone and assigning the units anywhere inside the zone will have equal impact on the assignment goals. An example of this are the units or apartments which are to be assigned inside the zones like the Privately owned housing / Social sector rental housing / Free-Sector Rental Housing. The zones which have been designed for them already satisfy the desirability of the location in the building in the zoning problem itself so the aim at this stage would be to reach the target number of units inside the zone which can be achieved by simple mathematical subdivision of the grid.

The second category of unit assignment problem is similar to the zoning problem where there is a relationship for the closeness of the units to each other and each unit itself has a preference for placement in the zone. The problem in this case would be a higher resolution zoning problem itself with closeness requirements. The aim here would be similar to the zoning problem to achieve the maximum benefit of assignment based on the desirability matrices for the various units considering the constraints of closeness in the problem. The Office zone in the Case of Buiksloterham can be considered as one such problem.

8.0.2 Process:

Step1: Interpolation of the Zoning problem output:
The first step in the Unit assignment problem is to make the resolution of the voxelated grid finer from a 6x6x6m size to a 3x3x3m size. This finer resolution will make assigning the units more accurate due to the smaller division size of the area into voxels.

Step2: Assigning Vertical and Horizontal Shafts:
In the base lattice consisting of zones and a high resolution lattice with smaller voxels first the shafts would be assigned . The shafts in this case includes the vertical circulation elements like the stairwell and the elevators as well as the buildings services shafts. The horizontal shafts would include the circulation passages connecting the various spaces in the building.

Step3: Assigning Green Spaces:
The assignment of green spaces depending on the shape of the Zones and the units needs to be done. The location of Green spaces has the potential to elevate the spatial quality of the units and the zones around it.

Step4: Assigning the first category of Units:
The first category of units are the ones without any closeness requirements or relations. These include all the housing zones in the space program and the zoning output which will be assigned first.

Step5: Assigning the Second category of Units:
The Second category of the units include the ones having closeness requirements with other units in the zone. The units for the office zone the restaurant zone lie under this category. The assignment of these units will be formulated like the zoning problem.

8.0.3 Interpolation of the Zoning problem output:

The aim of interpolation step is to make a higher resolution zoning problem matrix by interpolating the values from the 6x6x6m voxel lattice to a 3x3x3m voxel lattice.The following steps are considered in the interpolation problem :

Step1: Initialization of the base lattices:
The first step in the process is to voxelise the massing model into a 3m unit lattice in a similar manner as shown in the process of voxelization of the massing model for the zoning problem [7.1].This will act as the low resolution target lattice and the out put from the zoning problem will act as the reference lattice.

Step2: Interpolating the values:
The values from the zoning lattice containing the ids of the various zones as voxel values will be interpolated using the function *RegularGridInterpolator* from the Scipy library[(Virtanen et al., 2020)]. The class of nearest neighbour interpolation is considered since the requirement is to maintain the same zonal id's of the voxels

Step3: Saving the interpolated lattice as base lattice:
The output from the interpolation will be saved as a base lattice for the unit assignment problem. The improvement step as mentioned and done on the sequential assignment method will also be done on this output for a better structure.

8.0.4 Assigning Shafts and Circulation pathways:

The next step before assigning the units in the zones is to determine the nature of the circulation spaces in the building. The circulation spaces consists of vertical shafts containing circulation elements like stairwell , elevators and building services elements like service shafts. The horizontal circulation elements include passages and corridors that bind the different spaces in the building and the vertical shafts for movement in the structure.

Step1: Assigning vertical shafts:
The main function of the vertical shaft is the movement of the occupants vertically and also out of the building. The location of the vertical shafts are mainly guided by the restrictions of effectively vacating the building in case of emergencies like a Fire. Another factor that is relevant when locating the vertical shafts is the effectiveness of the shaft in connecting multiple zones together. Shaft areas are typically non-saleable areas and the effectiveness in its design leads to a better financial model for the selling of the building.

The shape of the building also plays a critical role in the design of the vertical shafts. If the building is more rectangular with one side significantly shorter than the other, then the placement of the vertical shafts can be straightforward based on the evacuation distance. The evacuation distances are the distances from any point in a floor plate to the nearest exit point on that floor. The standards for this are generally mentioned in the development control regulations or the department of firefighting guidelines based on the location. Several other factors also determine the evacuation distance like presence of sprinklers and other fire fighting methods as well as the fire grade of the construction itself. Sometimes the regulations also call for the location of a certain number of shafts on the external periphery of the building for smoke exhaustion and fire fighting access.

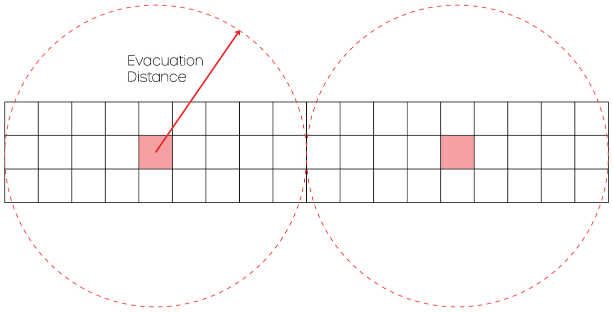


Figure 8.3: Rectangular shape vertical Shafts

If the shape of the building is more squarish in proportion then there might arise a need to place the shafts inside the shape along with the periphery. In these cases the shafts can cater to the requirement of connecting maximum zones in the building. This problem of connecting maximum zones to a shaft can be interpreted as a problem of centrality where the centroids of the cluster of voxels need to be determined for the location of the shaft. This can be done by using a clustering algorithm like K-Means clustering to determine central position with respect to the voxels of certain zones or the outermost voxels of certain zones, effectively reducing the number of shafts required to connect them.The shafts were also placed centrally so that they are accessible from both the sides of the layout.

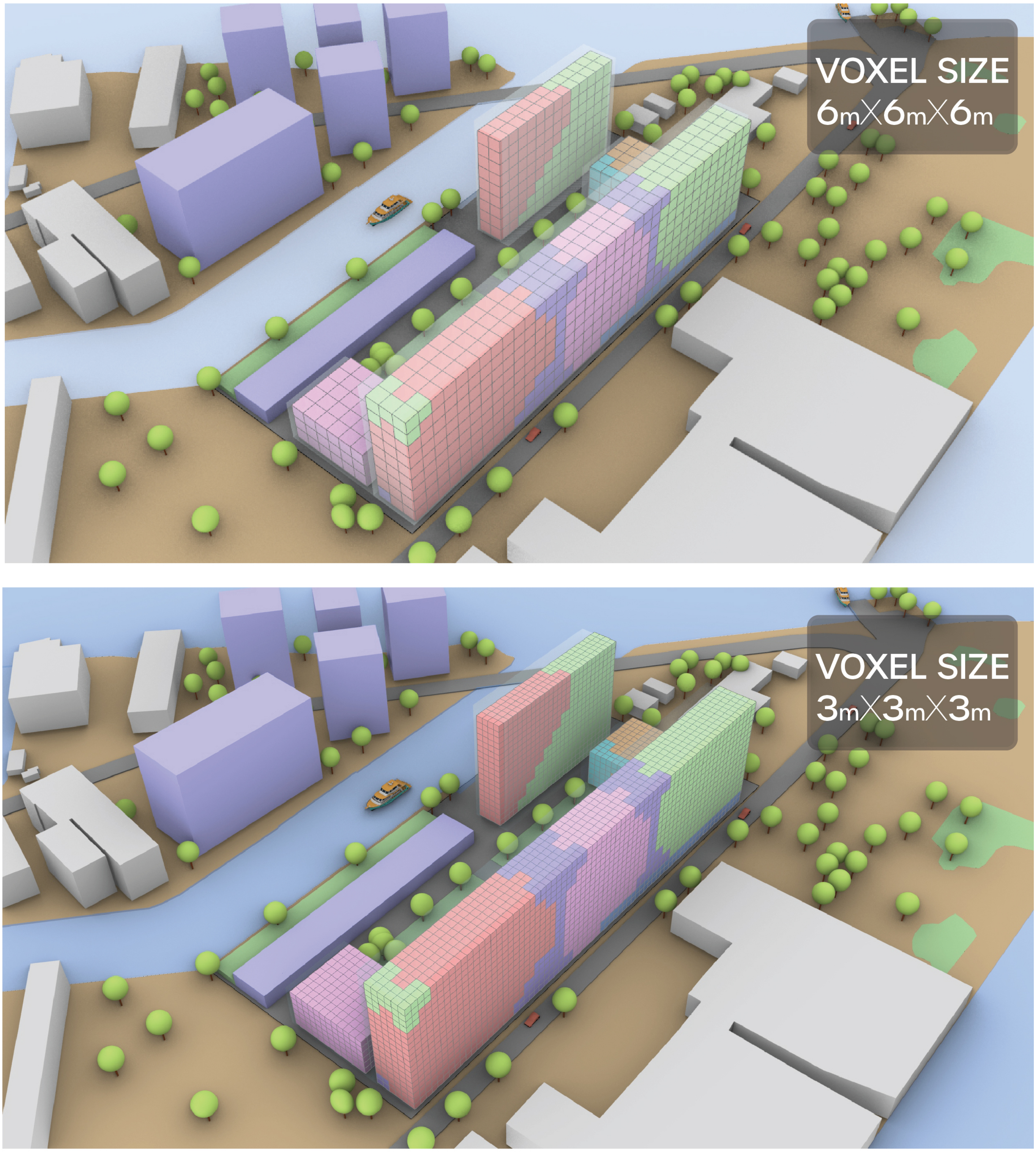


Figure 8.2: Comparison of the Interpolated output with the original one

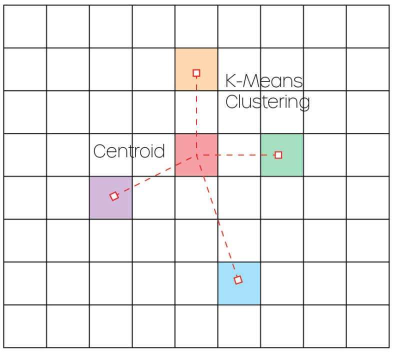


Figure 8.4: Squarish shape vertical Shafts

In the selected massing model for the case at Buiksloterham the shape of the buildings are rectangular so the first type of vertical circulation strategy seemed to work the best for it. The evacuation distance considered was 20m considerations the regulations at the site location.

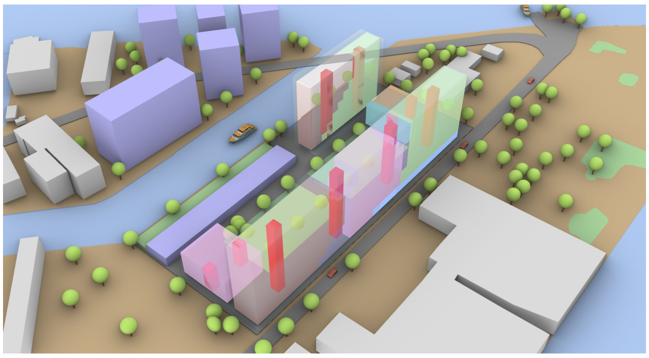


Figure 8.5: Vertical shafts location in the model

Step2: Assigning Horizontal shafts:
The process of locating the horizontal shafts is a tricky one. In the literature studied where previous attempts have been made to solve the problem of assigning shafts the approaches were mixed where in some approaches horizontal shafts were assigned first and then the units and in some examples it was the opposite.A good method of approaching this problem can be found in the book Form Space and Order by Francis DK Ching [(Ching, 1979)]. In the chapter of Path-Space relationships the relationship between the path (shaft) and the spaces(units) have been made in the following ways.

Pass by Spaces: In this type of relationship the spaces are located along the paths meaning that the paths are drawn first and then the spaces are assigned.The advantage of this type of relationship is that the integrity of the spaces are maintained, the path configuration remains flexible and mediating spaces like the vertical shafts can be used to link the path and the spaces.

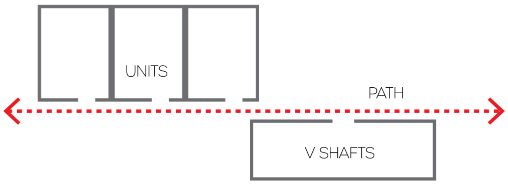


Figure 8.6: Pass by Spaces relationship diagram

Pass through Spaces: In this type of relationship the corridors or the horizontal shafts pass through the spaces or units. The path may pass through the space axially, obliquely or along its edge. In cutting through a space, the path creates patterns of rest and movement within it.

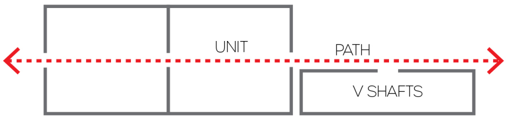


Figure 8.7: Pass through Spaces relationship diagram

Terminate in a Space: In this type of relationship the corridors or the horizontal shafts pass end in the space or unit. The location of the space establishes the path. This path-space relationship is generally used to approach and enter symbolically important spaces.

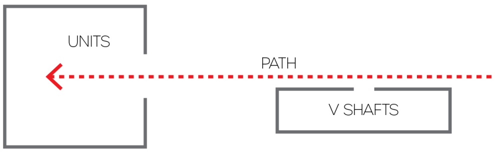


Figure 8.8: Terminate in a Space relationship diagram

In the case of the design for Buiksloterham the predominant function is the residential apartments which have a strong tendency to form the first type of relationship that is the Pass by relationship with the passages or the horizontal shafts.So the horizontal shafts were created at each floor interval of 3m in the building block connecting the vertical shafts. The spaces or zones where there is no need for this continuous connection or where the nature of the path-space relationship changes to a pass through spaces like in offices there this continuous connection is broken if required.

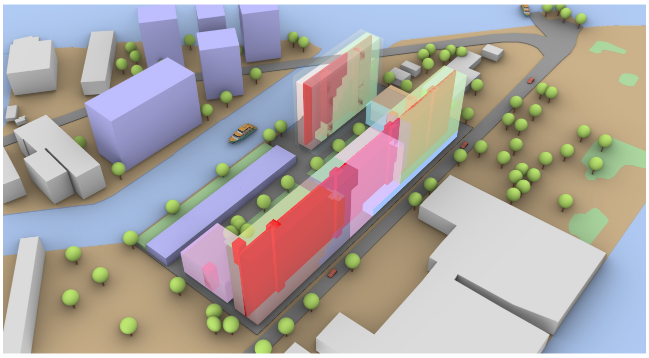


Figure 8.9: Horizontal shaft placement in the case

8.0.5 Assigning Green Spaces:

Green spaces or void spaces is also a design feature aimed in this project. The location of these spaces in the building mass ensures that there is a privacy buffer between zones like offices and housing and also acts as a means to improve the spatial quality of some of the spaces in the building. The green spaces are strategically located in the model where the aggregated desirability values of all the zones in the zoning model are the lowest or where there is a requirement of a buffer space.This is a design element which is specific to this project portraying the flexibility of the methodology and the system to adapt to different design features.

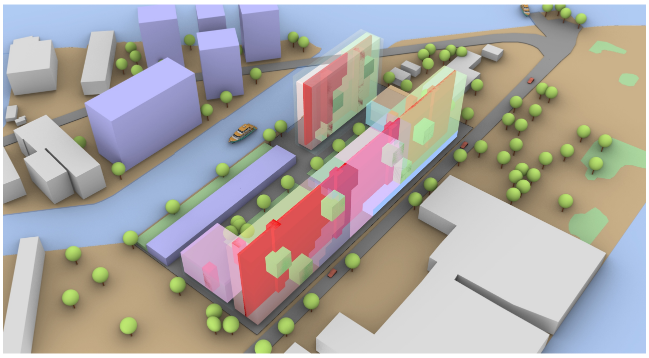


Figure 8.10: Placement of Green spaces in the Model

8.0.6 Assigning the first category of Units:

In the first category of units all the units under the residential function are assigned. This includes Privately owned housing, Social-sector Rental Housing, and Free-sector Rental Housing.The unit assignment process for this category will be explained by considering the example for the privately owned housing units.

Step1: Generating the modules for units:
The privately owned housing units have two categories as seen in [8.1] the Large house consisting of 18 voxels and the medium house consisting of 9 voxels. The zone for the privately owned housing is split in the center by the horizontal circulation space in the previous step and is symmetrical with one side containing a maximum of 3 voxels. Considering this dimension as the extent for each unit since intermediate units are not desirable due to poor access to the sun the units for the large and the medium can have a shape of 3x6 voxels and 3x3 voxels respectively.

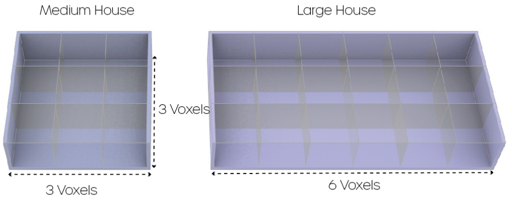


Figure 8.11: Basic Units

Step2: Generating the intermediate modules:
The counting of the voxels is done in each direction for the two symmetrical sections of the zone. One dimension is the same being 3 voxels but the other dimension will keep on varying. This variation in the other dimension will lead to instances where the dimension is not divisible by three which is the minumum size of the unit (Medium house). For these specific instances intermediate scaled units need to be designed. In the shape for the privately owned housing two such intermediate unit types need to be made of sizes 4x3 voxels and 5x3 voxels respectively.

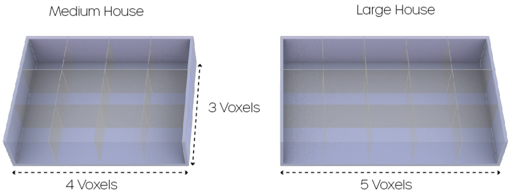


Figure 8.12: Intermediate Units

Step3: Assigning the modules according to the desirability matrix:
Once all the types of units are made the assignment process starts. The Large houses are given a priority and are assigned first sequentially floor after floor starting from the top floor since the maximum values from the desirability matrix as seen in the zoning problem for the privately owned housing were at the top of the lattice. [7.19]. At the end of each floor the intermediate modules are assigned if the length is not perfectly divisible by three.

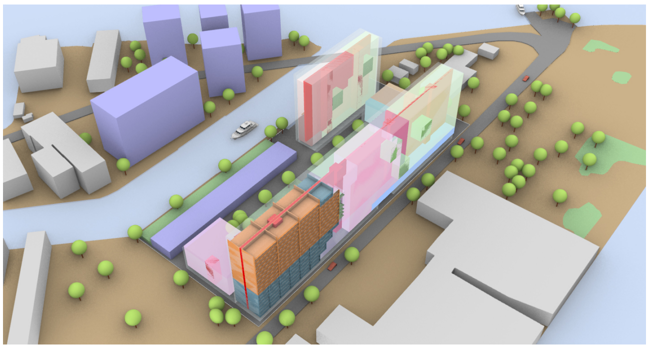


Figure 8.13: Final output of assigning Privately Owned Housing units

This process is repeated for the other two type of housing units, thus concluding the unit assignment process of the first category.

8.0.7 Assigning the Second category of Units:

The second category of units are the ones having closeness requirements with each other. An example of this is the zone for Offices and the restaurants and retails zones. To show the method the offices zone will be considered. The approach taken will be similar to the TCR method as seen in [7.0.33].

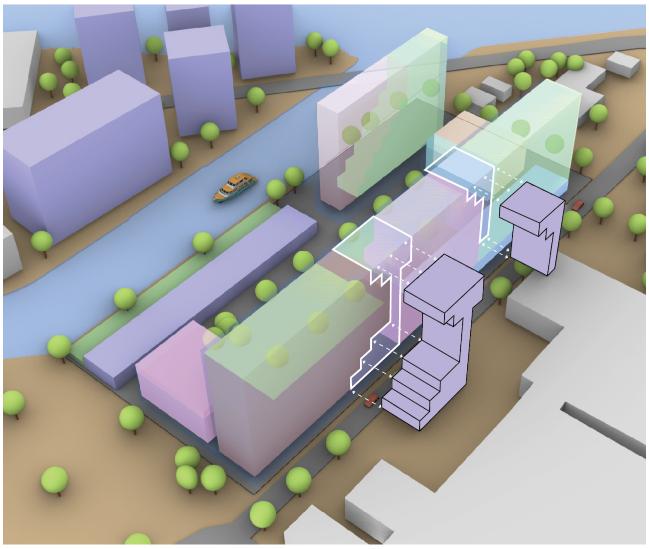


Figure 8.15: The zones for the offices from the zoning problem output

Step1: Problem Formulation:
In this step the various zones needed for the units will be determined. The number of voxels for each

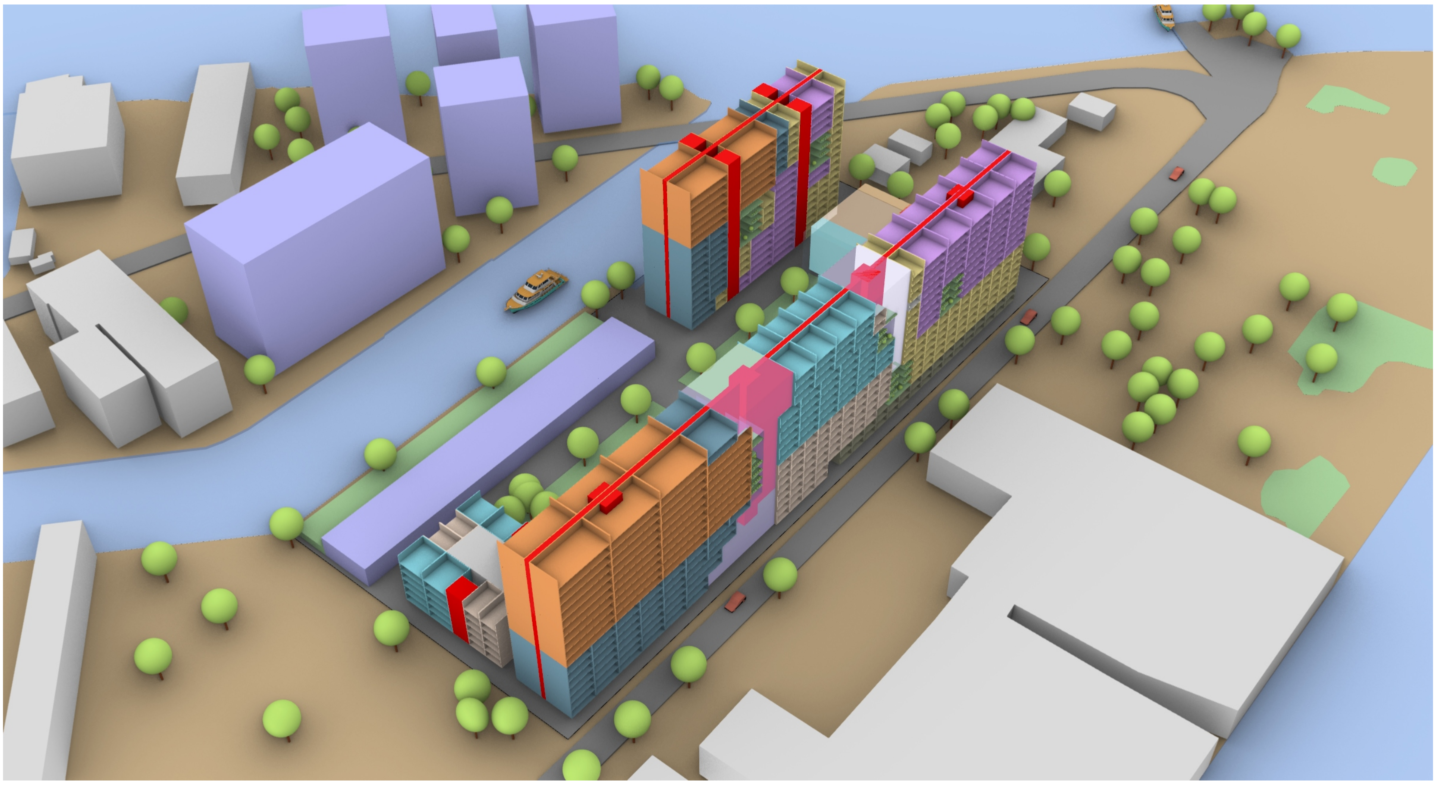
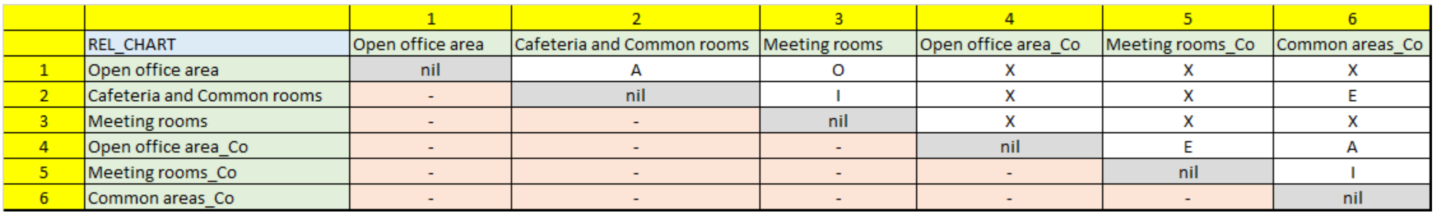


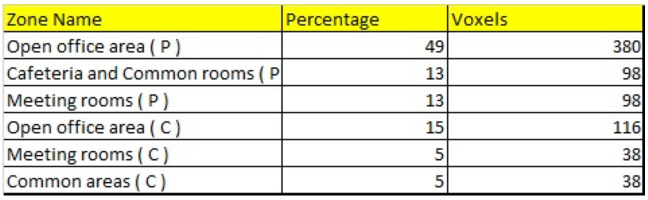
Figure 8.14: Output from the Category one Assignment



		1	2	3	4	5	6
	REL_CHART	Open office area	Cafeteria and Common rooms	Meeting rooms	Open office area_Co	Meeting rooms_Co	Common areas_Co
1	Open office area	nil	A	O	X	X	X
2	Cafeteria and Common rooms	-	nil	I	X	X	E
3	Meeting rooms	-	-	nil	X	X	X
4	Open office area_Co	-	-	-	nil	E	A
5	Meeting rooms_Co	-	-	-	-	nil	I
6	Common areas_Co	-	-	-	-	-	nil

Figure 8.16: Rel-Chart for the Offices

zone and the closeness requirements of the zones with respect to each other will also be stated. The Table [8.16] states the Closeness requirements for the problem . In the problem the office spaces can be classified into two categories one for private office and the other for the Co-working offices. The sub-categories under each of these categories are the same with different sizes as shown in below: (P) indicates private offices and (C) indicates Co-working offices.



Zone Name	Percentage	Voxels
Open office area (P)	49	380
Cafeteria and Common rooms (P	13	98
Meeting rooms (P)	13	98
Open office area (C)	15	116
Meeting rooms (C)	5	38
Common areas (C)	5	38

Figure 8.17: The details of zone sizes for the problem

Shape of the Lattice = 31x7x19 (LxHxW)
Size of the Lattice: 1953
Size of available voxels: 768
Objective: To find the best locations which satisfy the closeness constraints for the origin of six agents for the six zones in the problem, so as to maximise the total benefit which is the summation of all the values gained by the agents from their respective environment fields during the simulation.

Step2: Setting up the environment:
The environment needs to be setup for the specific part of the whole lattice comprising of the extents for the office zones. The relevant simulations then need to be run on this high resolution lattice for the office zones. The simulations which were chosen for this problem include:

Eucledian Distance based Simulations: The Eucledian distance based simulations include the closeness to the north and the south facade and to the roof and the floor. The closeness to the east and west facade were ignored since the zone for offices lies in between other zones and has no direct relationship with the east and the west facade. The simulations for the Quiteness lattice were also made considering the same sources of sound as [7.9] in the Zoning problem. One additional simulation was done for the closeness to the green areas in the building since the model at this stage had well defined green spaces. The closeness is again Euclidean closeness calculated in a similar manner to the Quiteness lattice.

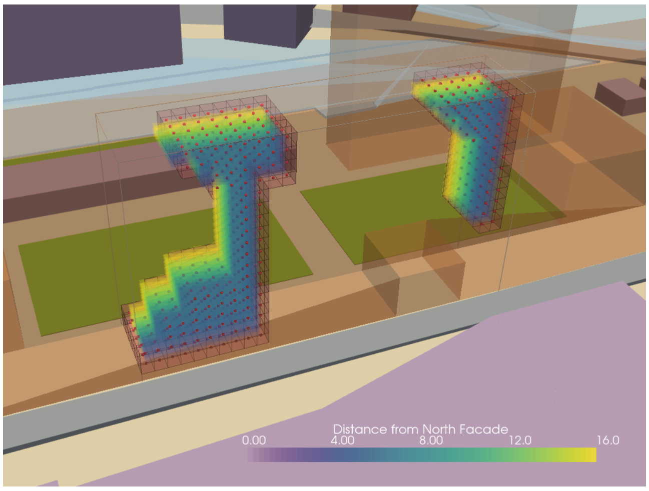


Figure 8.18: Distance from the North facade Lattice

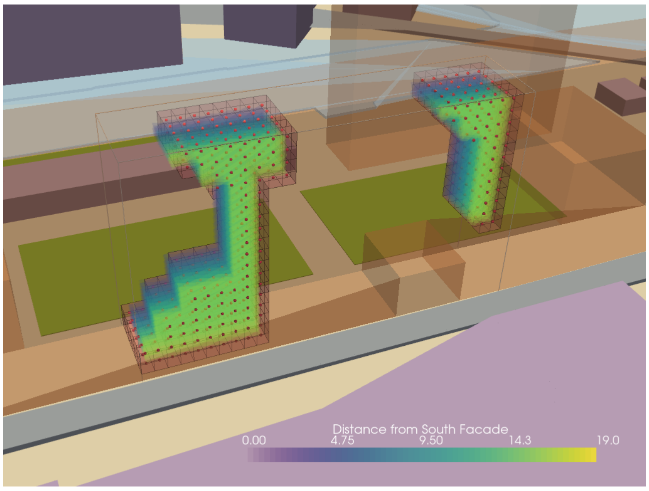


Figure 8.19: Distance from the South facade Lattice

Mesh Intersection based Simulations: The mesh intersection based simulations like Visibilty (Road,IJ) and access to sun were also performed on the refined lattice for the problem same as [6.1]. An additional constraint was added to the simulation where only the values for the outermost layer of voxels was considered since the inner layers technically have zero access / visibility.

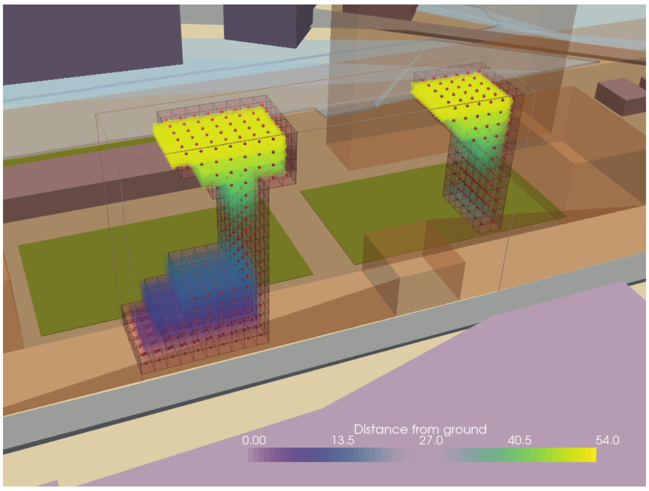


Figure 8.20: Distance from the Ground Lattice

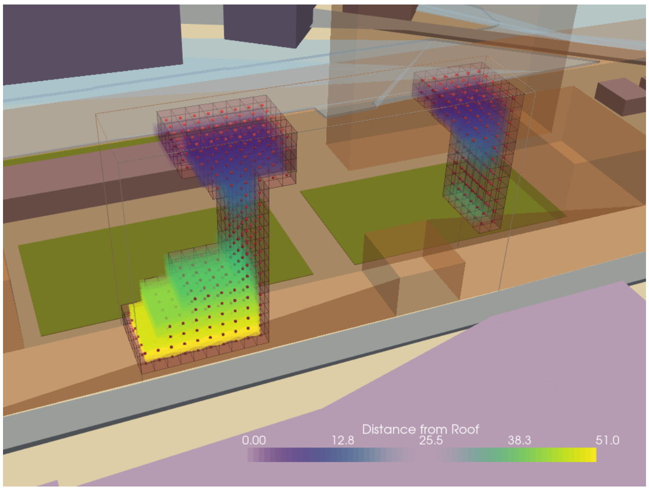


Figure 8.21: Distance from the Roof Lattice

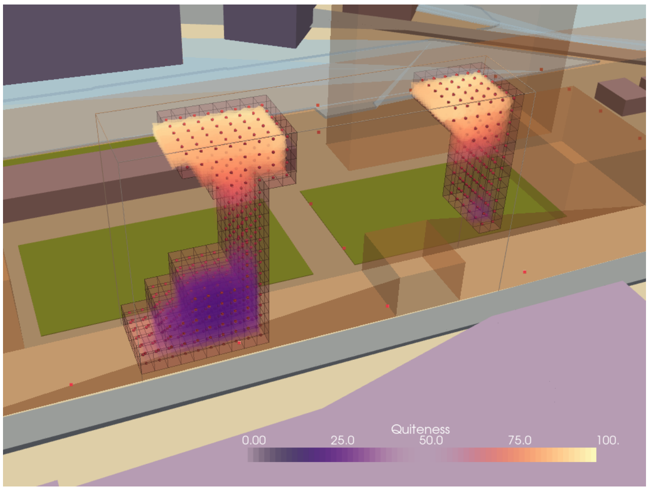


Figure 8.22: Quietness lattice

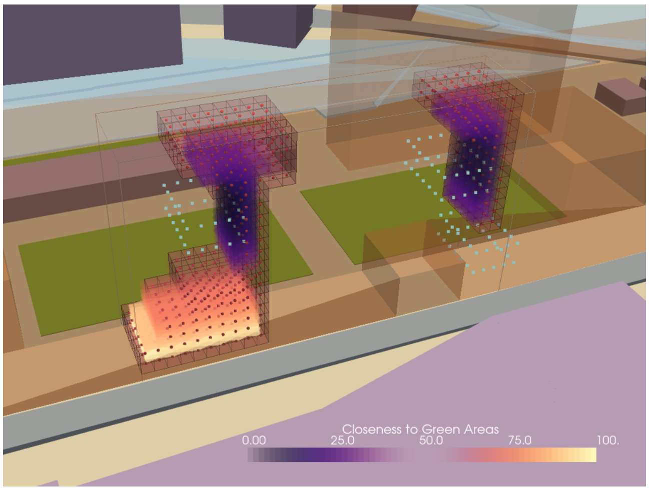


Figure 8.23: Closeness to Green Areas Lattice

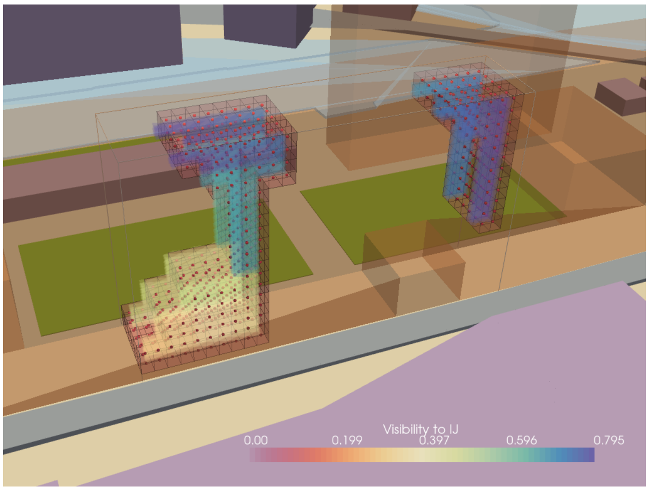


Figure 8.24: Visbility to IJ Lattice

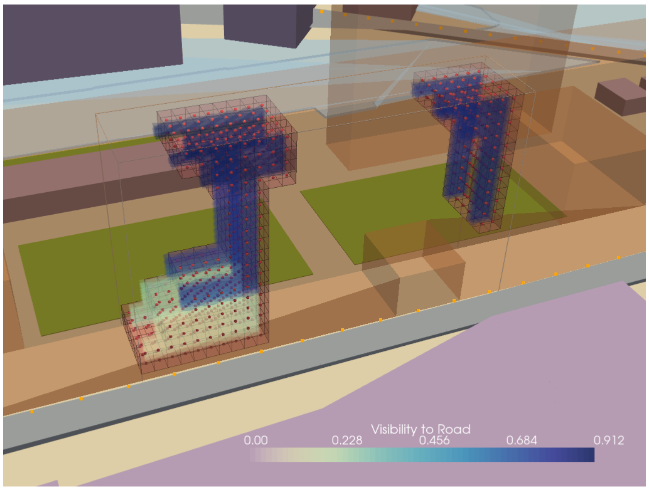


Figure 8.25: Visbility to Road Lattice

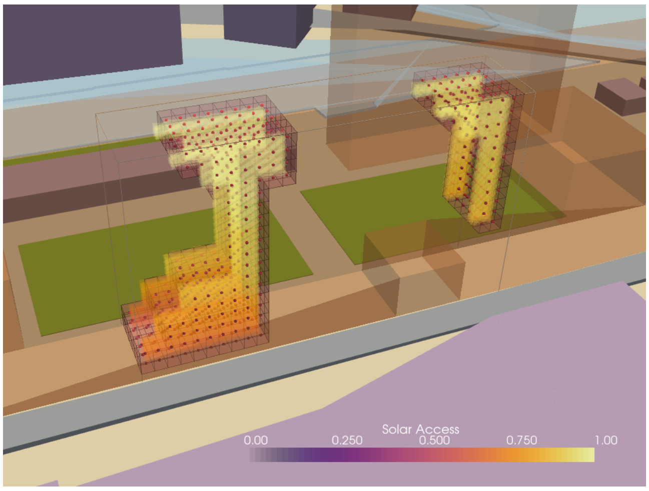
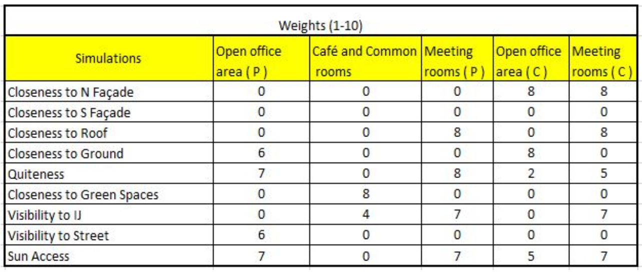


Figure 8.26: Sun Access lattice

Step3: Generation of the Agent Performance criteria:
The agent performance criteria for each of the zones in the problem will be created in the same manner as the zoning problem [7.0.4]. The choices for the type of simulations and the importance of the simulations in terms of weights will be assigned based on the previous step and the agent performance criteria in the form of a desirability lattice of the TOPSIS process will be created. This will form the base for the ABM simulations in the next step.



	Weights (1-10)				
Simulations	Open office area (P)	Café and Common rooms	Meeting rooms (P)	Open office area (C)	Meeting rooms (C)
Closeness to N Façade	0	0	0	8	8
Closeness to S Façade	0	0	0	0	0
Closeness to Roof	0	0	8	0	8
Closeness to Ground	6	0	0	8	0
Quiteness	7	0	8	2	5
Closeness to Green Spaces	0	8	0	0	0
Visibility to IJ	0	4	7	0	7
Visibility to Street	6	0	0	0	0
Sun Access	7	0	7	5	7

Figure 8.27: Weights for the various Simulations for the zones

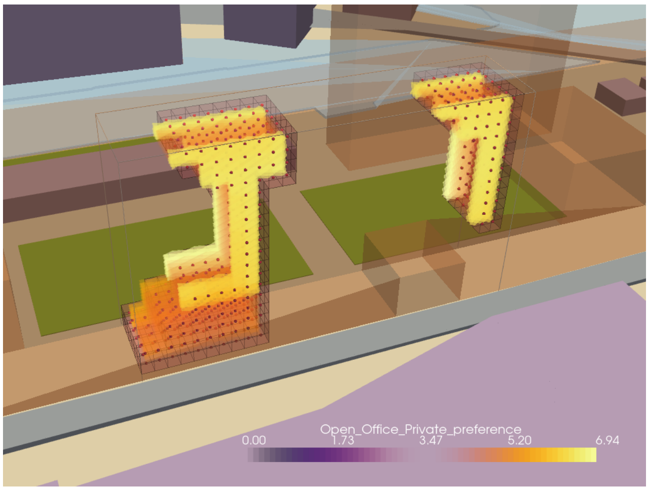


Figure 8.28: Desirability Lattice for Open Offices (P)

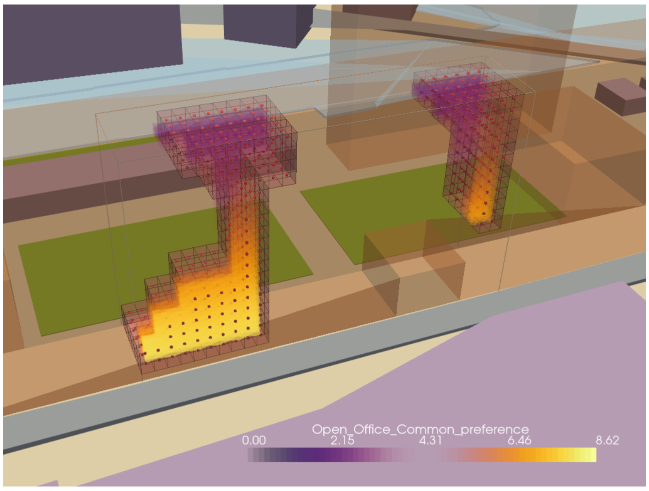


Figure 8.29: Desirability Lattice for Open Offices (C)

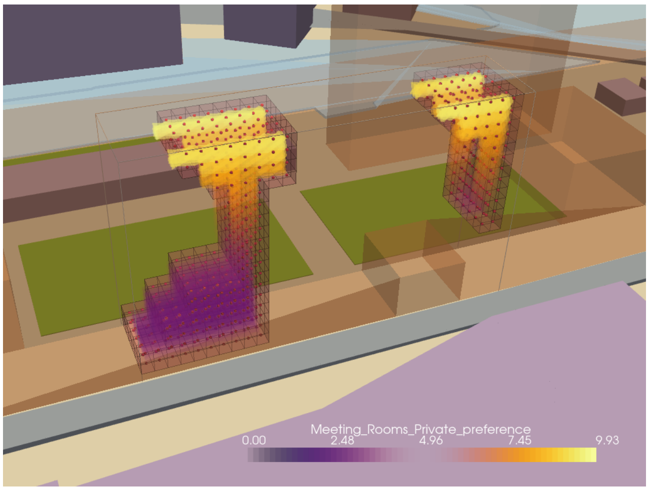


Figure 8.30: Desirability Lattice for Meeting Rooms (C)(P)

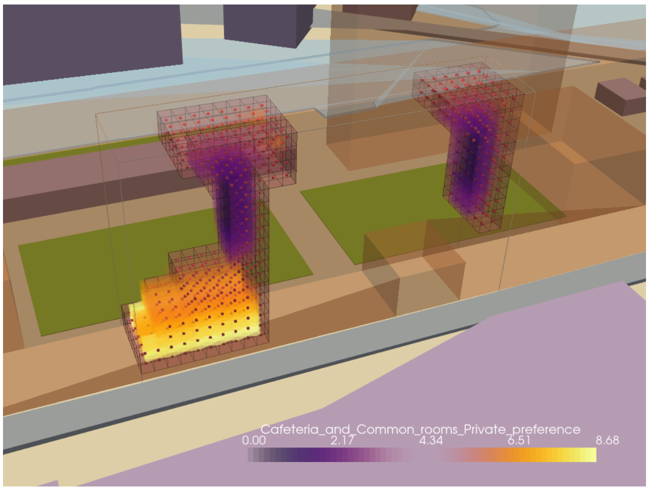


Figure 8.31: Desirability Lattice for Cafeteria and Common Spaces

Step4: ABM Simulation for generating the unit placement:
Two types of Agent based simulations were done to obtain the results for the unit assignment. The first one is the Linear assignment approach one

where the benefit of assigning the agent at certain number of locations in the lattice was first obtained and then the linear assignment process was undertaken. The second approach consisted of the TCR simulation one where the agents were deployed based on their TCR ratings. The second approach is similar to the sequential assignment approach with added TCR step. The lattice was parsed to find 82 different locations in the lattice distributed on both the North and the South side of the lattice with an interval of about 3 voxels between the cells on both the sides. The main reason for doing this was because the visibility,Sun access simulations performed in the previous step which have zero values in the interior voxels.

The conditionals for finding the permutations for the locations of the agent origins were set based on the relationship chart. This is similar to the approach taken in the toy problem for TCR Simulation[7.0.33] where a graph is created using the eight neighbourhood stencil for the lattice and the shortest distance from each voxel to all other voxels is calculated. The conditions for generating the locations of agent origins takes this calculation into account and finds the right iterations.

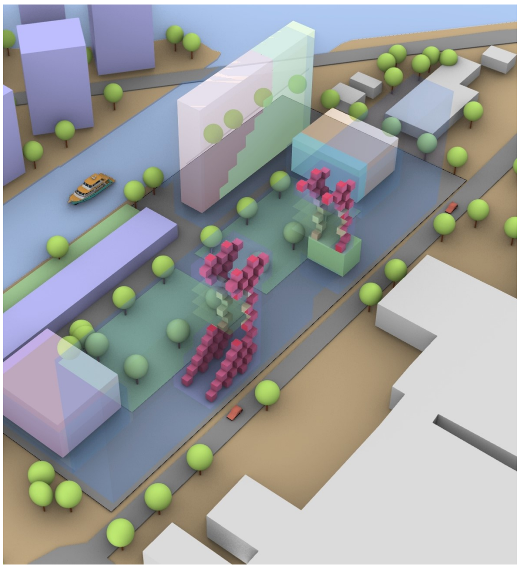


Figure 8.32: All possible Origins for the ABM Simulation

Step 5: Generate the Units from the Results:

The results from the both the simulations give similar results since the number of agents deployed in both are the same. It is interesting to note that for the relationships which are of the nature avoid tend to find positions in the two distinct disconnected segments of the zones for offices. The only common zone in the Private and the Co-working offices is the cafeteria which spreads into both the segments of the zone and is connected by the green space in the middle. This seemingly obvious solution to the Rel-Chart relationship is important to validate the efficiency of the method used in finding valid relations from the Rel-chart.

A similar improvement process is undertaken as seen in [7.0.45] and in the final zoning output the units are created according to the shape of the zone. Similar process can be done for the Retail and the Restaurant zone. With this step all the units are completed in the Unit assignment problem.

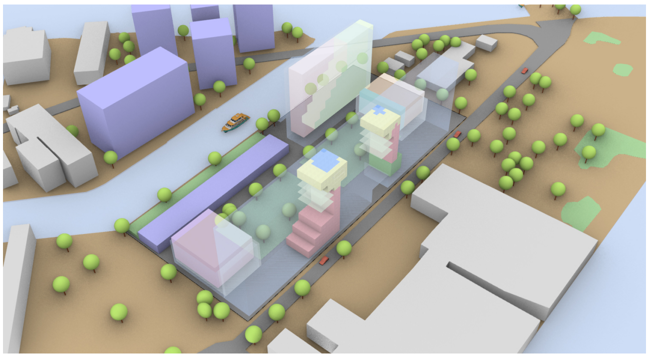
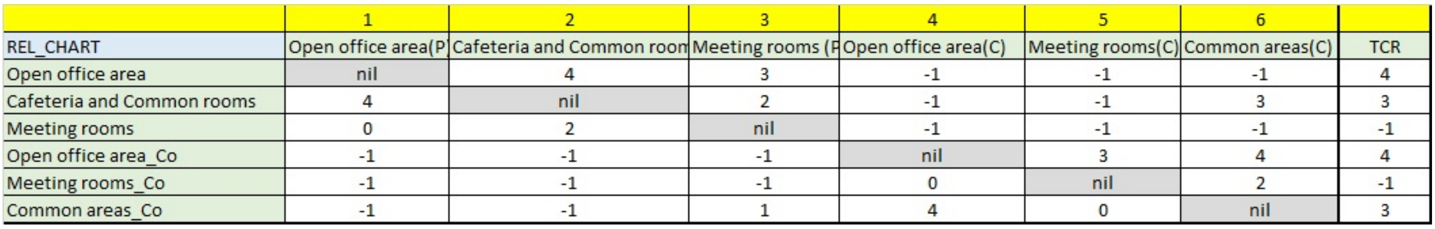


Figure 8.35: Final Output from the Linear assignment approach for Offices

8.0.8 Conclusion and Limitations for the Unit Assignment Problem:

The multi-scalar nature of the problem can be dealt with the multi-agent system effectively as seen in the Unit assignment problem . This shows the versatility and adaptability of the method to solve various levels of the configuration problem. The TCR method used in the second category of assignment problems is still bound by the same limitations as defined in the zoning problems and needs the same fixes for a improved result.



	1	2	3	4	5	6	
REL_CHART	Open office area(P	Cafeteria and Common roon	Meeting rooms (F	Open office area(C)	Meeting rooms(C)	Common areas(C)	TCR
Open office area	nil	4	3	-1	-1	-1	4
Cafeteria and Common rooms	4	nil	2	-1	-1	3	3
Meeting rooms	0	2	nil	-1	-1	-1	-1
Open office area_Co	-1	-1	-1	nil	3	4	4
Meeting rooms_Co	-1	-1	-1	0	nil	2	-1
Common areas_Co	-1	-1	1	4	0	nil	3

Figure 8.33: TCR chart for the Office Units

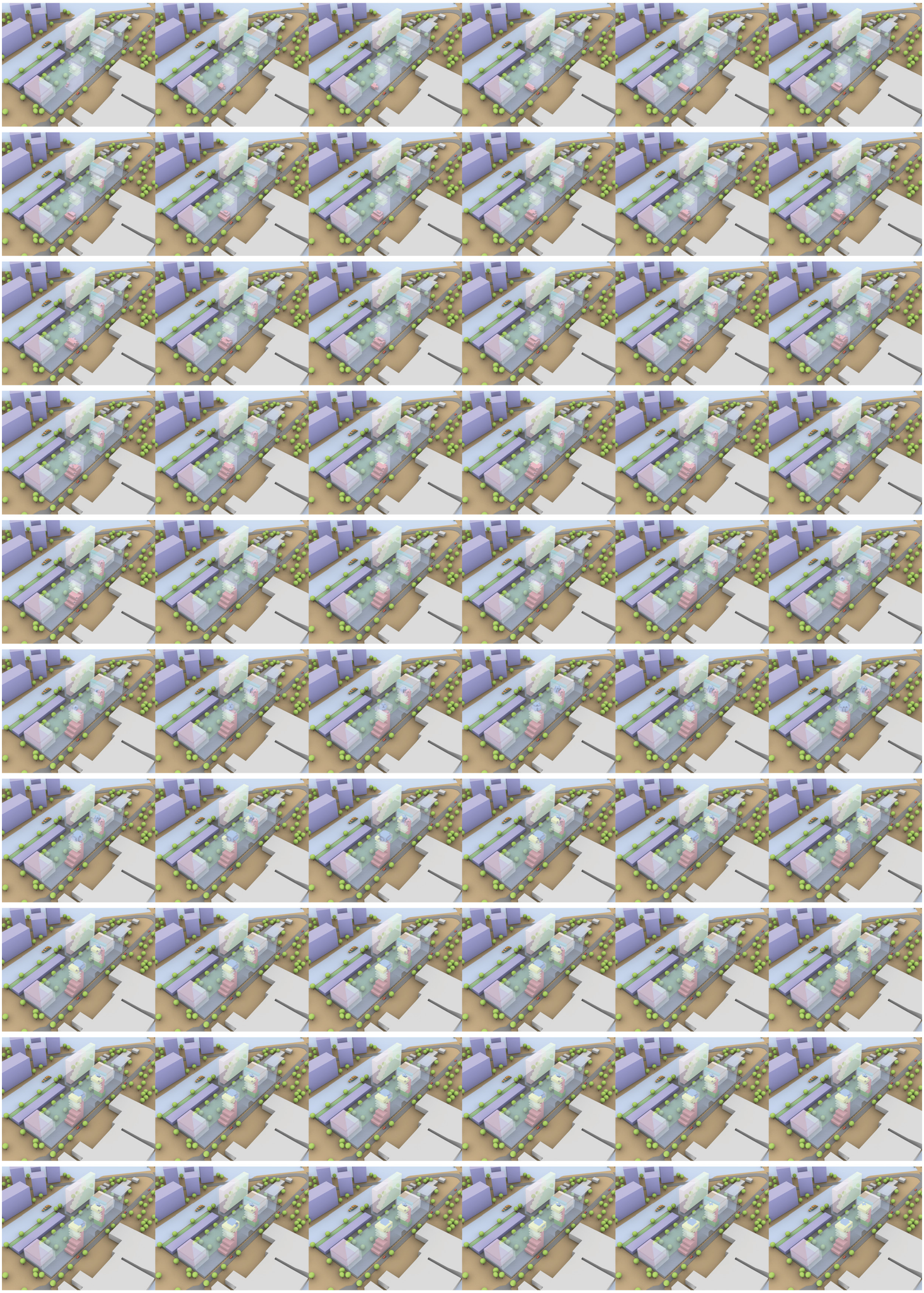


Figure 8.34: Simulation result for the Unit assignment problem for Offices

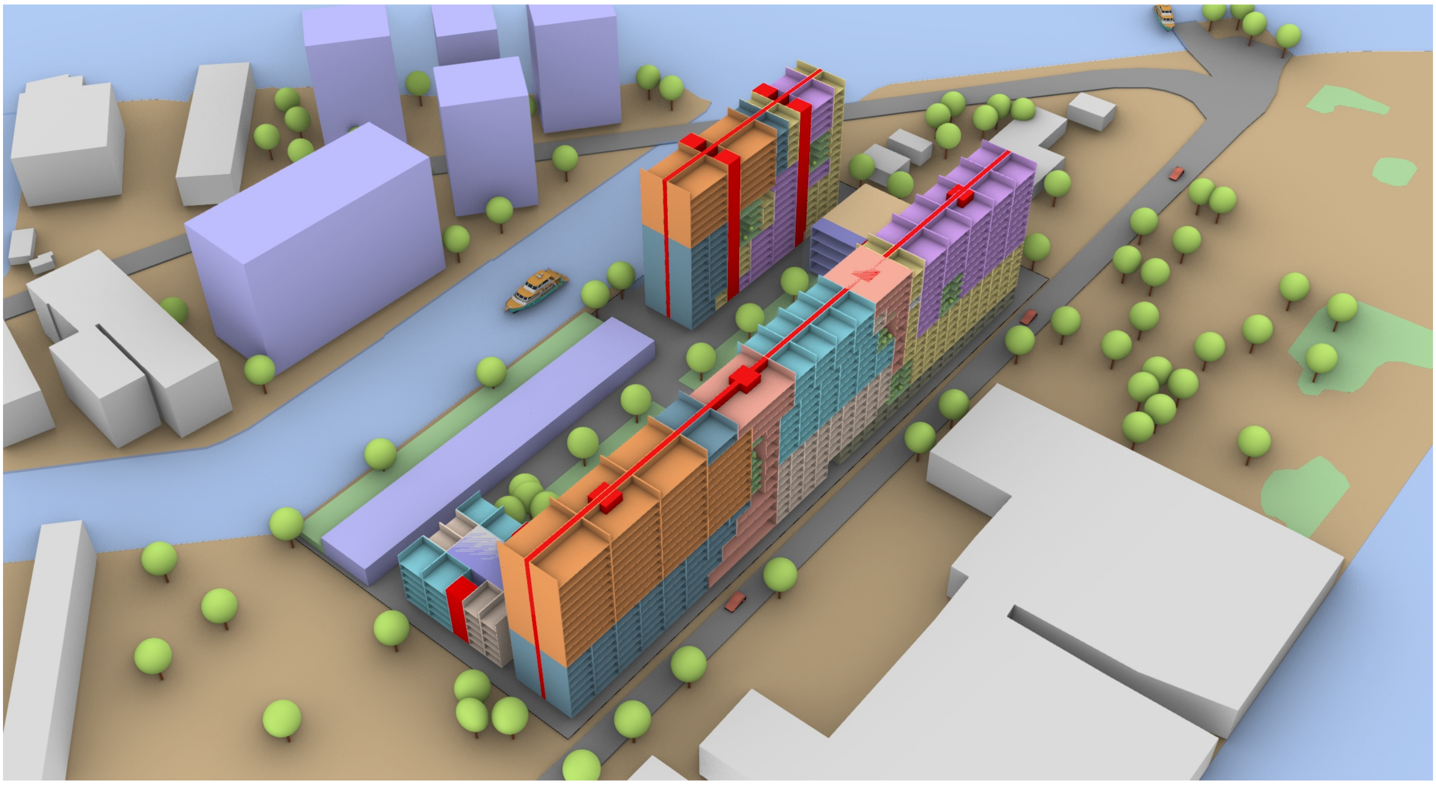


Figure 8.36: Final Output of the Unit assignment problem

The output from the unit assignment problem is clustering of voxels in the zones into numbered units. The box like representation shown in the diagrams is indicative of possible increase in the level of resolution of the design details. In the stage of Unit assignment instead of just representative boxes the structural resolution layer could also be added. The zones defined in the zoning problem have a certain activity defined for them and a load associated with it which can be used as a reference for preliminary calculations. The process of voxelization can generate the spans and distances accurately for the premilinary structural analysis. The same voxelised system can be used to design a structural sizing and design solution as well by considering a certain typology for structural design like a core based system or a simple frame based system. However this lies out of the scope of the thesis but would be certainly a great improvement in the unit assignment level of the configuration problem since it would help in making the problem of unit layout more detailed one.

The output from the unit assignment problem is further voxelised into smaller grid of 1x1x1m voxels or 1.5x1.5x1.5m sized voxels for the Unit Layout stage. The base for the unit layout stage would be a fine resolution voxelised lattice of each individual units from which the bottom layer will be considered for the unit layout problem.

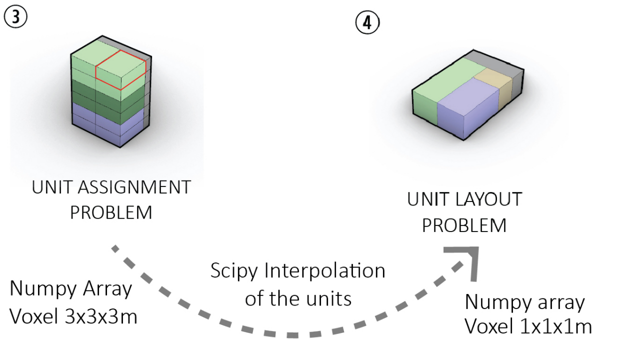
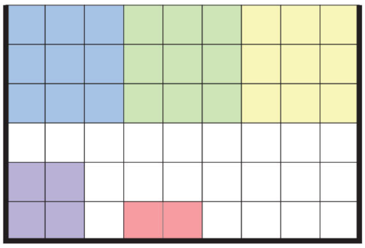
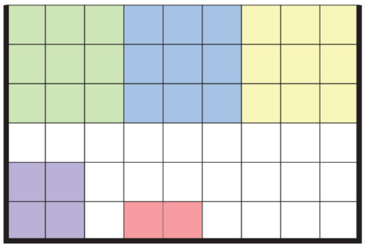
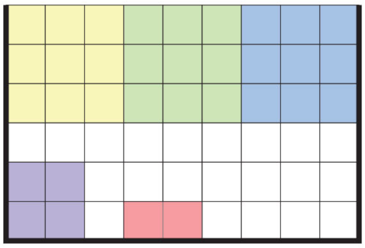


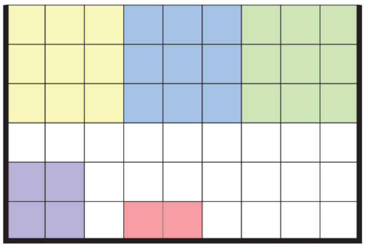
Figure 8.37: Unit Assignment problem to Unit Layout problem

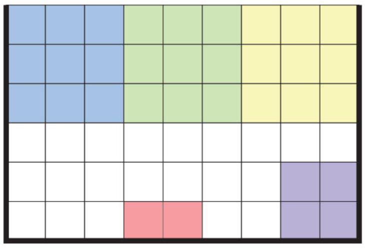
UNIT LAYOUT PROBLEM

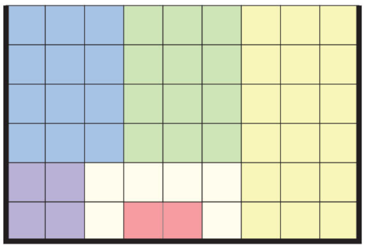


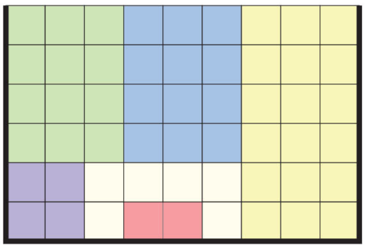


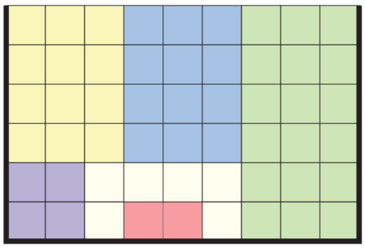


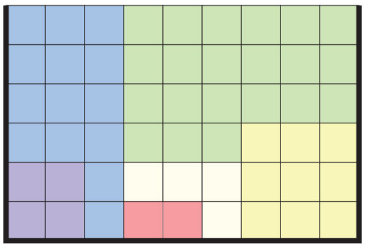


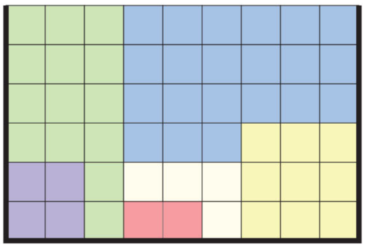


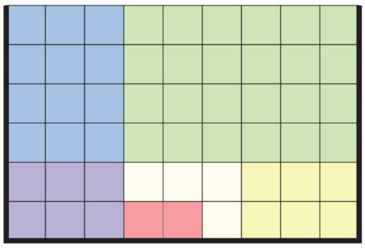


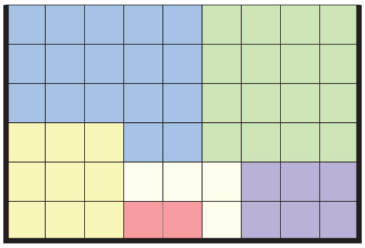


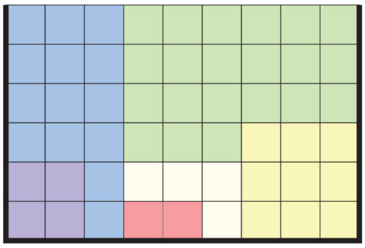


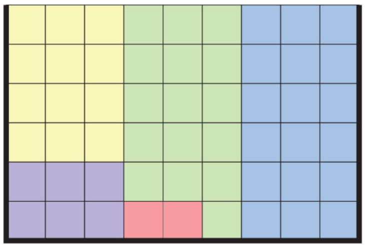


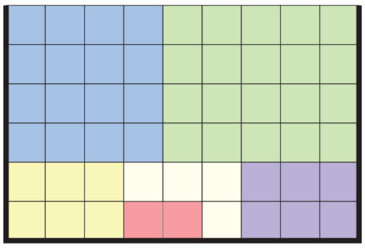


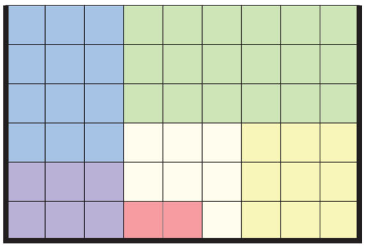












GEN-ARCH

9 Unit Layout problem

The Unit layout problem deals with the generation of layout for individual units as defined in the unit assignment problem. The unit layout is highly dependent on the use case scenario of the space so the interaction and the participation of the user of the space is extremely important at this stage. The unit layout and the subsequent unit detailing stage in its full complexity are outside the scope of the thesis. In this chapter certain possible approaches and ideas are proposed for further research into this problem.

9.0.1 Aim:

The main aim of this level of configuration problem is the generation of unit layouts based on the choices and desires of the final end user of the space along with respecting the design intention of the architect.

9.0.2 Background for the digital implementation:

Looking at the case studies performed [3.2] in the literature review chapter the past approaches towards the duality of this problem has been that the architect hosts a discussion with the end user, understands the spatial desires of the user and generates design options for them. The user can specify the details of the rooms inside the units in terms of the number and sizes and some specific choices for the aesthetics and materiality. This is indeed a good approach towards participatory customization when there is a limited number of units to be designed. A slight increase in the number of units may lead to a massive increase in the workload for the architect.This problem calls for development of a system that can understand the requirements and translate it into unit layout options for the user to choose from if the problem is to be tackled at a larger scale. Some of the approaches that can be taken are listed below:

Gamification of design could be an approach which can potentially solve this problem. In the gamified approach the Architect would act as the game master responsible for setting up the rules and the game elements which are acceptable according to the design vision and also the desires of the end users. The end users will play the role of the participants where they play the game till they are satisfied with the result of the game. This co-design mass customization approach can potentially cater to a large number of units to be designed together by the respective end users in collaboration with a small number of designers involved.The subject of design games can be a thesis in itself so the complete digital implementation of it is out of the scope of the project ,however a few methods and ideas are presented on how it can be implemented.

Rule based systems for generating procedural content is also an option to create logical options of configurations. The development in procedural content generation which is logical and spatially sensible has been a question in the video gaming industry for a long time. The need to develop and generate massive 3d environments have always been a challenge and rule based systems offer a potential solution for that. Inspiration can be taken from this research to design a system for unit layout generation.

One of the interesting algorithms in this domain is the Wave function collapse algorithm (WFC) which has the ability to produce a large amount of generative content based on the rules of aggregation.The system for the same is a tile based in two dimensions or a voxel based system in three dimensions with adjacency rules. The WFC algorithm has a special approach towards constraint solving via elimination. Each location in the grid stores the information on what the tile can and cannot be and during the solving phase, one tile is selected and given a single random solution from the remaining possibilities. This choice is then propagated throughout the grid, eliminating adjacent possibilities that don't match the input model.The algorithms also has the feature of backtracking where if the selection choices lead to a contradiction, they're reverted and a different observation is attempted. The algorithm doesn't always generate logical outputs but manages to solve into logical outputs for a lot of instances.

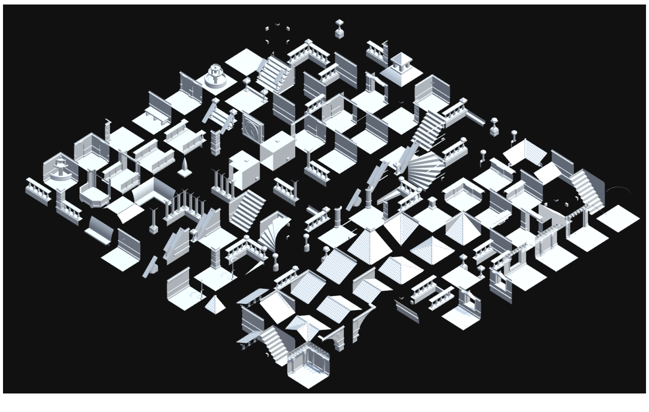


Figure 9.1: Tile set for the WFC implementation [(Kleineberg, 2019)

A classic example of this is the work of [(Kleineberg, 2019)] where a procedural city was developed by using tile sets and a set of adjacency rules as seen in the figures [9.1],[9.2],[9.3].

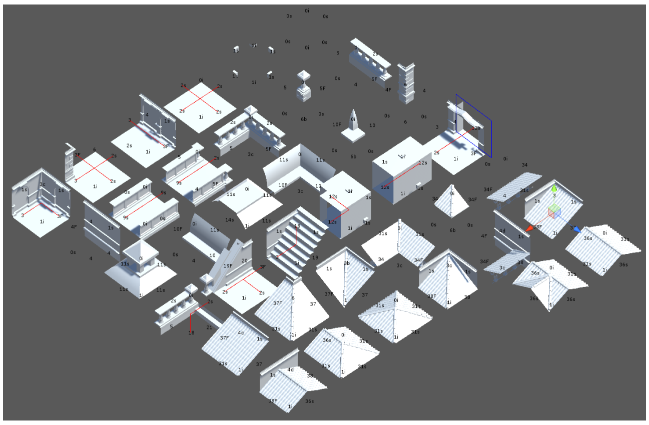


Figure 9.2: Adjacency rules for the WFC implementation [(Kleineberg, 2019)

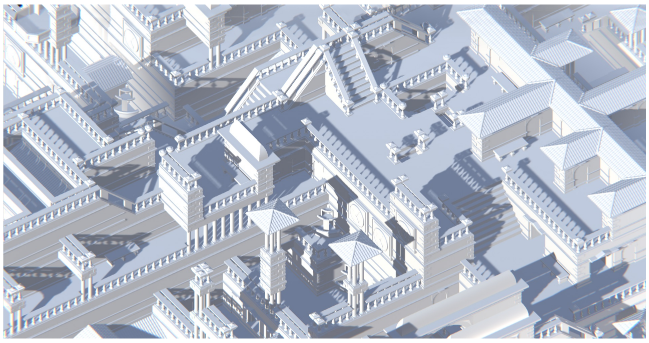


Figure 9.3: Output of the WFC implementation [(Kleineberg, 2019)

This algorithm as well as similar rule based grammar or systems have been explored and developed into the tools like Monoceroes and Wasp for the Rhino-Grasshopper platform which would be a good avenue to explore. However there are certain concerns with these type of systems. The first one being the level of control provide by these systems is quite limited. Using them for design exploration or form exploration is quite interesting but for generation of layouts where the sizes and shape of the rooms are bound by size constraints and location constraints it becomes very difficult to use it effectively. Secondly these systems combine the Unit layout and Unit detailing levels together by means of creating 3d blocks which is counter intuitive to the approach taken in the project of separating the layout problem into levels for effective tackling of problem at each level.
The problem can also be looked at from a point of view of a packing problem in the domain of Operations research / Computer science. A possible solution for this approach is explained in the next section.

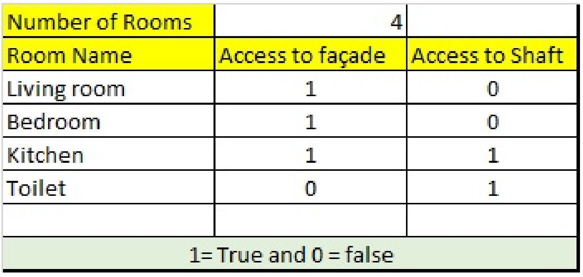
9.0.3 Configuration Approach:

The configuration approach that is proposed for the Unit layout problem resembles the Rectangle packing problem in computer science again inspired from the problem in the domain of operation research in industrial design. Rectangle packing is a packing problem where the objective is to determine whether a given set of small rectangles can be placed inside a given large polygon, such that no two small rectangles overlap. The rectangular packing problem in its full sophistication is a NP hard problem and become a very complex when the rotation and the sizes of the rectangles inside the in rectangle are made variable.

The approach proposed works on simplifications which make it a feasible problem to solve. The simplifications are methodical steps which are based on spatial logic and can be modified or deleted according to the participants requirement. This makes the approach flexible and transparent for adaptation to various design styles and requirements. The unit layout configuration for a small unit of size **9x6m** is considered as an test case or a toy problem and the following steps are followed:

Step1: Initial User preference/participant input: The participants or the end users of the house are given a list of preferences that they have to input based on their requirements. This will form the basis of the steps further in the process. The following inputs need to be specified for the system to generate the design options for the participants:

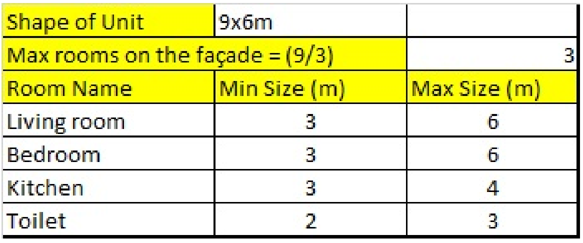
The users have to define **number of rooms** they will be needing in the house and their details. The details include the **function of the room and the requirement for a closeness to a service shaft**. the closeness requirement to the service shaft will be determined by the function of the room like for example a bathroom or a kitchen. The second preference which the users need to specify is the requirement of **direct access to the facade / sunlight / ventilation for the rooms.** In these two preferences the maximum number possible will be determined by the game master or the Architect based on the dimensions of the unit and the ergonomic/ proportions which fit in the design vision. For the toy problem the following requirements will be considered :



Number of Rooms	4	
Room Name	Access to façade	Access to Shaft
Living room	1	0
Bedroom	1	0
Kitchen	1	1
Toilet	0	1
1= True and 0 = false		

Figure 9.4: Initial user input by the participants

Step2: Initial master inputs:
The Architect or the designer is the controller of the process and needs to generate the following inputs for the system to initialize. The first one being the maximum number of rooms that can have direct access to the facade. This can be determined by the minimum dimensions of the room and the dimensions of the facade. The second input which the architect has to give is the maximum and the minimum room sizes for the possible rooms which the users can define. This is based on ergonomic principles of fitting the right furniture layouts inside the rooms and catering to the proportions which are comfortable for the occupants. Finally the architect also has to select the location for the Vertical service shaft which will carry all the ducting necessary for ventilation, plumbing , fire-fighting into the unit.*In open buildings locating the shaft centrally is very important for maintaining the flexibility of the layout. This approach is inspired by the Superlofts project mentioned in the literature review.* [3.23]



Shape of Unit	9x6m	
Max rooms on the façade = (9/3)		3
Room Name	Min Size (m)	Max Size (m)
Living room	3	6
Bedroom	3	6
Kitchen	3	4
Toilet	2	3

Figure 9.5: Initial master input by the architect

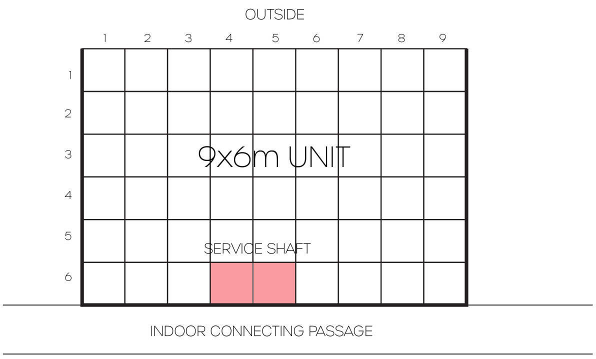


Figure 9.6: Initial layout of the unit

Step3: Generation of Base blocks:
The initialization process is completed at step two so all the necessary input for the floor plan generation is ready. In this step the base blocks for all the spaces is initialized and enumerated. The base block considered for each room is the **minimum size of the room which is needed.** The blocks are located such that they have access to the facade if specified in the previous step. The enumerated options for the base blocks can be see in the first part of the image in [9.13]. The options for the Toilet block to be places on the left /right side of the service block is also explored with the enumerations.

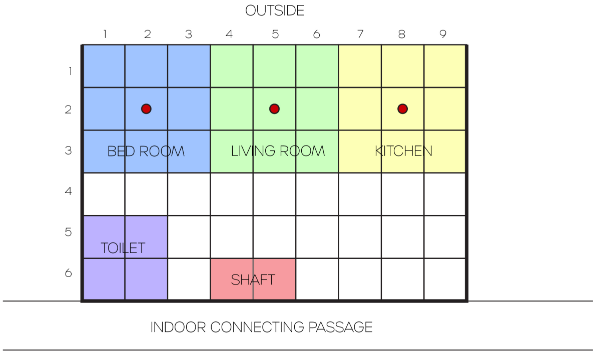


Figure 9.7: Base blocks initialization for the rooms in the unit

Step4: Distribution of remaining voxels:
After all the possible options are enumerated for the base blocks the question for distribution of remaining voxels is tackled. The total number of voxels in the base grid is 9x6 = 54 voxels, when subracted with the base blocks the number of voxels which are unassigned are 23 cells. These 23 cells have to be distributed among the 4 rooms in all possible ways. At this stage a constraint is added that the shape of the rooms should remain largely rectangular which means that at a time the cells corresponding to the minimum size for the room have to be added to the base block. This constrained coupled with the maximum possibility of having 3 rooms on the facade and the requirement of 3 rooms connected to the facade reduces the search space considerably. The possible floor plans because of this is around 7x4 = 28 floor plans. The base enumeration results in 7 options and the added options are based in the possible enumerations of toilet sizes and locations on either the left or the right side.

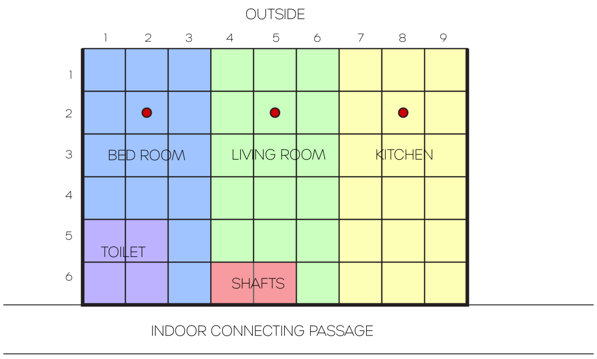


Figure 9.8: Distribution of remaining voxels for the rooms in the unit

Step5: Creation of Passages in the Layout:
Passages can be created to create more layout options and to improve the connectivity of the enumerated layouts in the step 4. Passages Can be created by specifying the path start and the end voxel and the shortest path can be created using the same method as described in the unit assignment problem horizontal shaft creation.

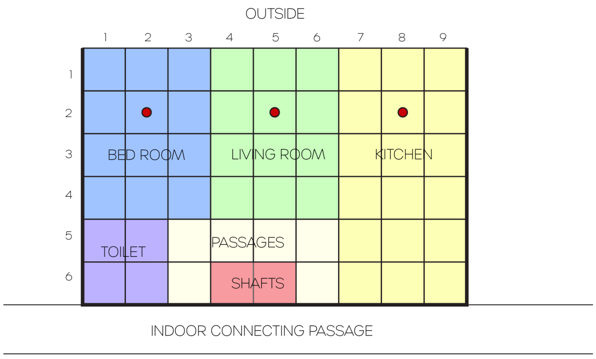
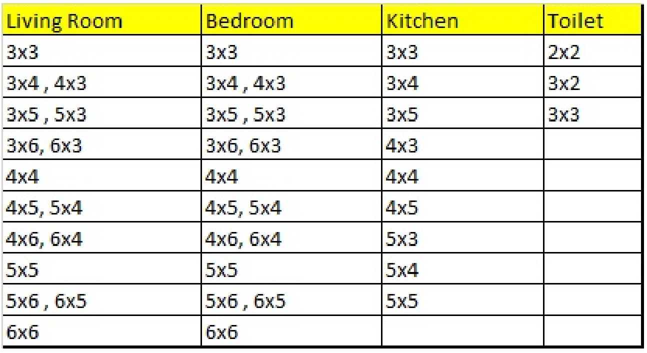


Figure 9.9: Creation of Passages in the Layout

Alternative Design criteria:
The layout possibilities become more interesting if the requirement of the rooms needing direct access to the facade is reduced . If the Toy problem is modified where the Kitchen no longer needs access to the facade the possibilities will increase considerably. The enumeration possibilities can be determined by creating a possible shape options for the rooms based on the minimum and the maximum sizes as shown in the table [9.10] The enumerations because of this process can be seen in the last part of the image [9.13] where the dimension of the room vary in X axis as well.



Living Room	Bedroom	Kitchen	Toilet
3x3	3x3	3x3	2x2
3x4 , 4x3	3x4 , 4x3	3x4	3x2
3x5 , 5x3	3x5 , 5x3	3x5	3x3
3x6, 6x3	3x6, 6x3	4x3	
4x4	4x4	4x4	
4x5, 5x4	4x5, 5x4	4x5	
4x6, 6x4	4x6, 6x4	5x3	
5x5	5x5	5x4	
5x6 , 6x5	5x6 , 6x5	5x5	
6x6	6x6		

Figure 9.10: Enumeration Possibilities

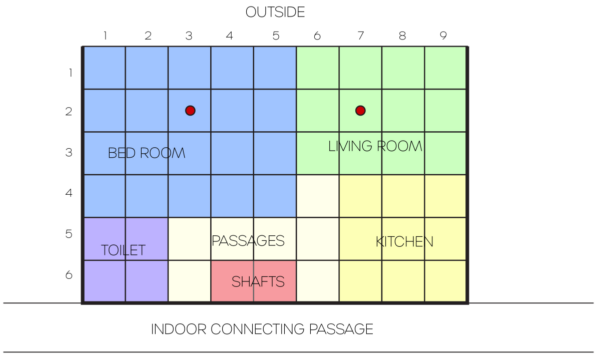


Figure 9.11: Design Alternative

Step5: MCDA and filtration:
Once all the possible layouts are generated based on the initial requirements the participants are given further choices like ordering the rooms based on their desired size , flow of the apartment in terms of which space comes first when you enter and the circulation distribution if the circulation in the apartment is central / linear (left oriented / right oriented). Finally the participants can select and give weights to these criteria. Based on the choices the calculations are done for all the layouts and a TOPSIS based MCDM model is developed for ranking all the layout options for the participants. The participants can choose between the options and the selected option will be further developed in the unit detailing problem. The MCDM is done to sort the layouts based on preferences of the participants which makes the selection process easier for them.

9.0.4 Conclusions and Limitations:

The approach proposed in the thesis offers a controllable generation of the unit layouts and the level of control can be increased or decreased based on the number of constraints and simplifications which are done to the rectangular assignment procedure. This process seems to work well for housing typology of layouts but for other types of layouts the constraints might not be enough.

For example in programs where there is a closeness requirements between the rooms this process will not work there is a need of an additional constraint of closeness.

This method is similar to the brute force method with constraints to limit the options and the time required and is bound by its limitation. The space-splitting method or the Physically based method where the spaces are represented by 2d bubbles and forces are applied based on the closeness requirements to generate the layout or the space splitting method where the data tree principles like Kd-trees are used to organise the floor plans could also be explored since they have proven to generate good quality 2d layouts in fixed boundaries as seen in the literature review chapter [5.3].

Alternatively the participants can be given a minecraft like game setting where they can do the assignment process of the rooms themselves to get familiarised with the process and get a better understanding of the participation process.The final out of this step is the selected high resolution voxelated layout showing the room placement inside the unit for the next step which is the unit detailing stage.

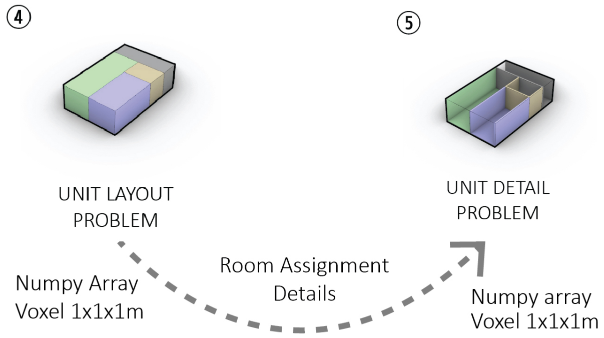


Figure 9.12: Unit Layout to Unit Detailing stage

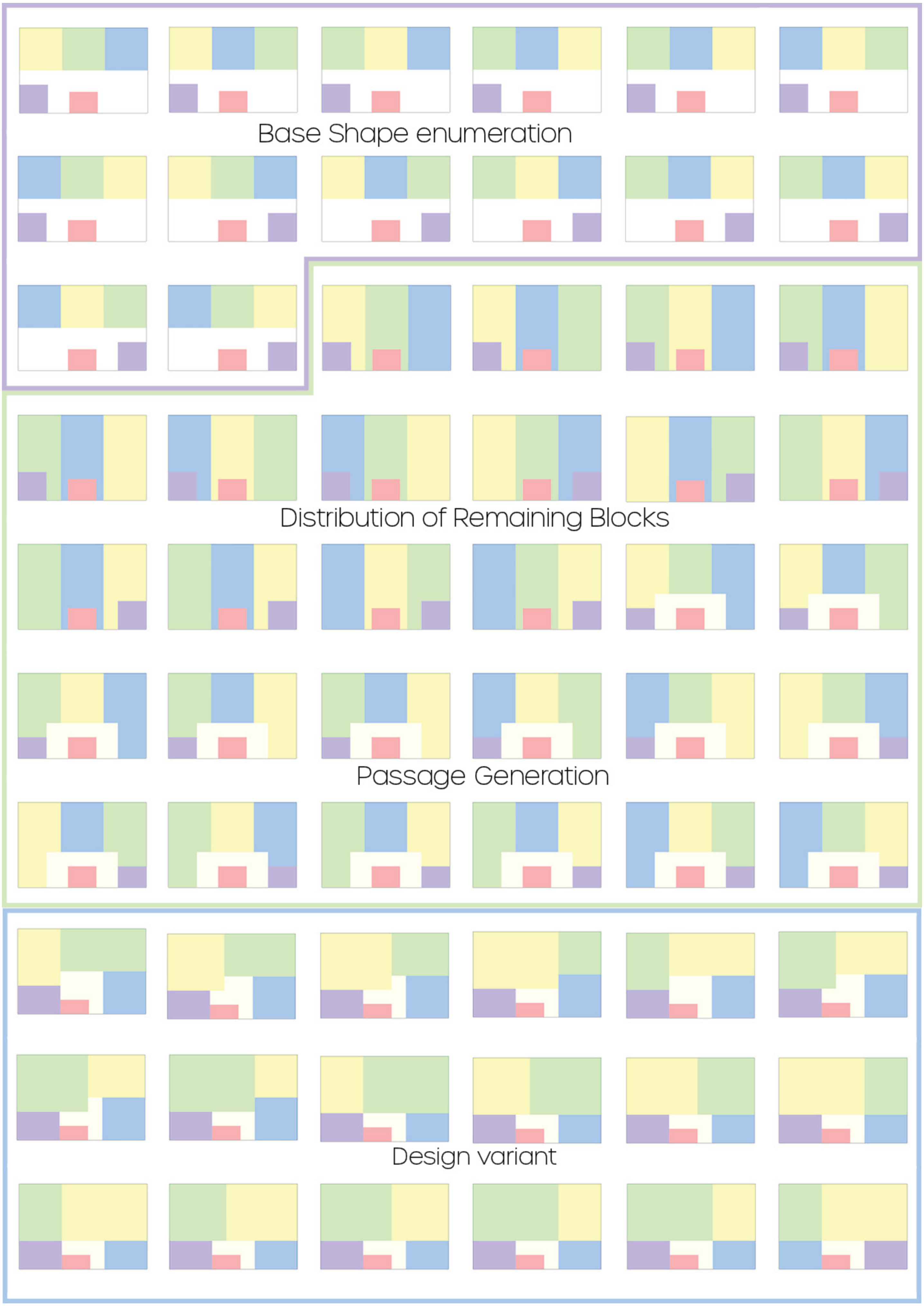
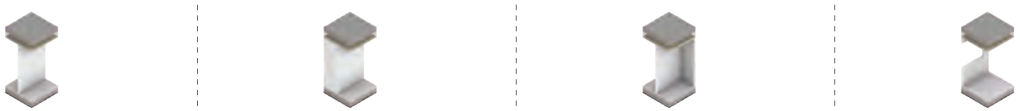
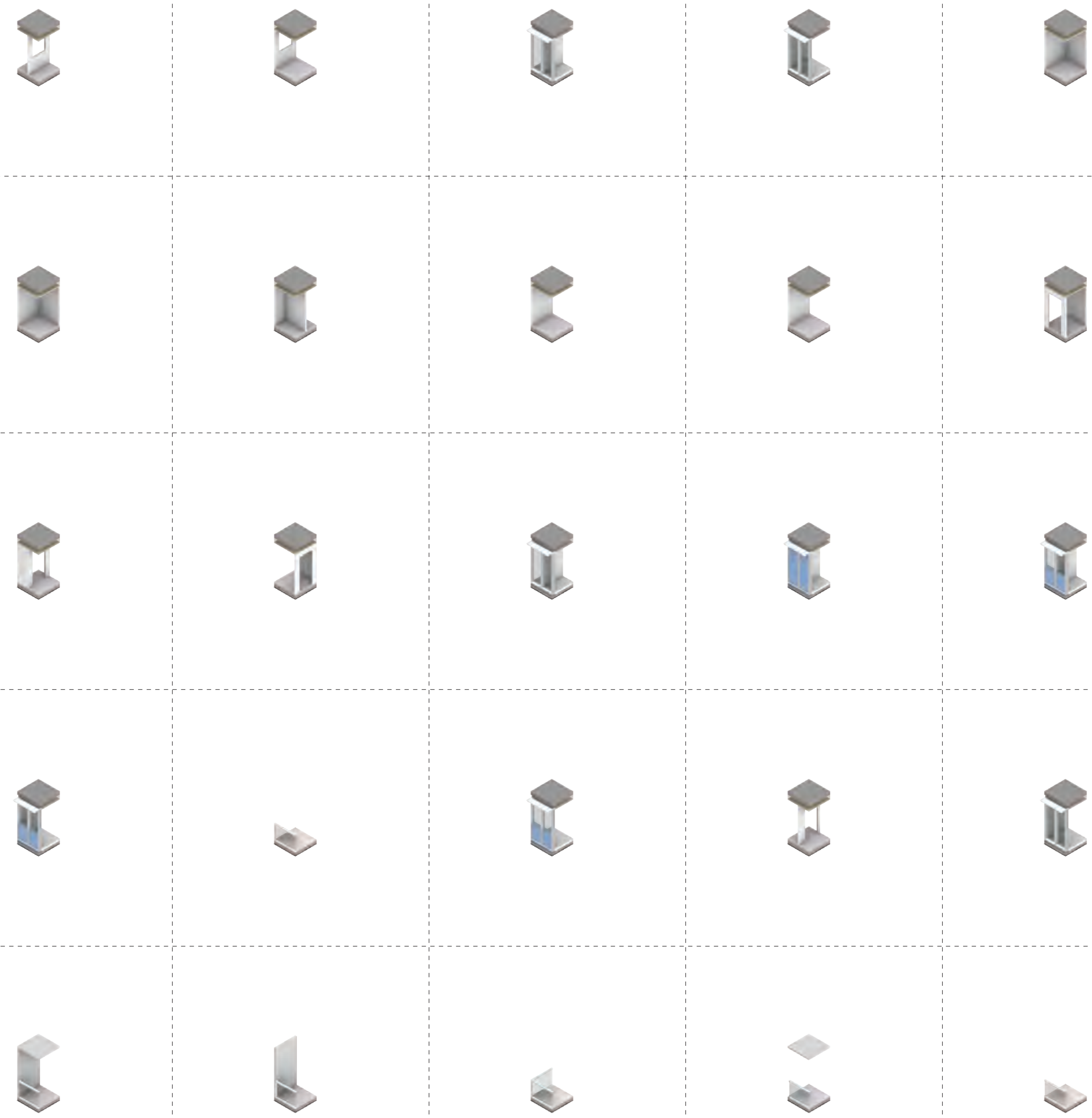
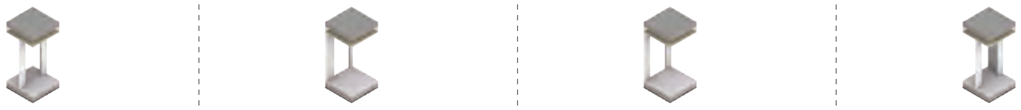


Figure 9.13: Unit layout problem process

UNIT DETAILING PROBLEM







GEN-ARCH

10 Unit Detailing problem

The Unit detailing problem deals with generation of design details for the units developed in the unit layout problem. Design details may consists of the following things: *Materials and components of the various infill layers in the unit like Walls, Windows, facades, floors, ceilings, installations, furniture etc.* The clear distinction between the structure and infill has been made in the open building concept and the infill layer is the main element which will be detailed in the unit detailing problem. The structure layer should ideally be resolved in the unit assignment problem itself so at this stage there is already information regarding the structure of the building.

The stakeholders involved in this stage of the layout problem are the Architects, Contractors, End users of the space/Home-owners, Building Component manufacturers. The system created for solving this problem would make the Architect/Designer as the mediator between the end user and the rest of the stakeholder generating the necessary inputs for the clear communication.

The nature of the problem at this stage will involve background knowledge about a lot of subjects like construction processes, building material properties, fabrication and assembly processes, cost estimation etc. This makes the scale of the problem very large and not manageable to solve in the time frame of this thesis, However a possible initial framework for solving this problem is proposed in this thesis.

10.0.1 Background for the digital implementation:

A feedback loop is essential for this stage of the configuration problem. The feedback can consist of various things like *building costs, material passport, material details and quantities, circularity index etc.* This will ensure that the user knows the impact of the selection decisions on the project.

Generation of 3d blocks which can store this information while displaying the final result visually will help in the decision making process. The approach of creating the 3d blocks is similar to BIM modelling where details of all the components are mentioned while creating families and blocks. Two kind of approaches can be taken to assign the created blocks into the floor layout . The automatic assignment approach and the manual one.

The automatic layout assignment approach is meant for the users who do not want to get into the customised the detailing of the apartments but are interested in the spatial indicators for making a decision like overall project cost, construction time, aesthetics and architectural style, etc. In this approach they can pick from the pre-designed custom sets by the Architect to suit their requirements.

In the manual design approach a high level of customization will be possible. The details for the room wise and boundary condition wise will be recorded from the participants or the end users and the necessary 3d blocks with the information will be first created with all the necessary design details. The user will be given a choice between the materials for each category of components like the walls , floors, etc and the result of the selection will be shown in the feedback loop. The selection will not only be based on material but on boundary conditions as well like internal walls, room wise details, edge of wet areas and dry areas, facades etc. Once all the details are recorded based on the boundary conditions defined the blocks will be assigned automatically or the user will be able to see and assign the created blocks in the floor layout to the specific voxels. This approach of 3D block creation is something similar to the reference of the game Block-hood [(Sanchez, 2021)] where assigning the blocks generates feedback about the resources utilized and the assignment process itself is based on certain adjacency conditions.

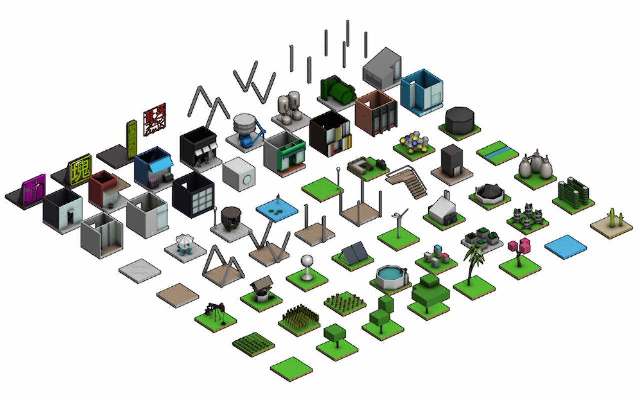


Figure 10.1: All possible blocks in the Block hood game [(Sanchez, 2021)]

10.0.2 Configuration Approach:

In the configuration approach for the unit detailing problem there will be a distinction made between the objects and the infill layer. All the building elements will will make up the infill layer and the furniture, appliances, etc. will come under the object layer. The configuration problem stops at the infill level and the object level goes beyond the scope of the configuration problem.

The 3d block generation can happen in either 2d environment or a 3d environment. The two dimensional environment generally consists of generation of blocks for single height spaces and the 3d environment will consist of larger volumes and multi-story spaces. The minimum number of blocks which need to be defined to create a space vary for both the cases.A two dimensional case would be considered to illustrate the system for unit detailing.

The following steps are taken to define the unit detailing:
Step1: Initialization and Block creation:
As a first step a plan is selected from the unit layout problem stage for further developing in the unit detailing stage. The blocks are configured in the following manner: Every block will have flooring slab , flooring finish , Ceiling finish . There is a further option of creating a block with wall segment for internal walls and facade segment for external walls. The blocks with wall segment can be created either as a peripheral wall block or a central wall block. To create a set of blocks which will generate any type of an layout both the types of blocks are necessary. Furthermore in case of a block with wall the blocks need to be made with a single wall, with double wall for corner junctions for both the centrally placed walls as well as the peripheral walls. The walls themselves can have openings which can be plain empty openings, doors, and windows. A minimum set of blocks for each material and structure option needs to be created so that floor plan of any shape can be generated from it.

The blocks with facade elements need to be made in a similar manner. The corner element needs to be made. the central connecting element need to be made, the element with the connecting wall on the facade needs to be made as base for creating a facade block. The facade block can also vary in terms of infill elements which it can host like solid infills, glass infills, windows, solar panels, awnings etc.

Some special blocks were also created like the blocks necessary for generating balconies in the floor plan. Special blocks can include balconies, Architectural features and projections, Green walls and external flowerbeds, etc. Any block that can enhance the architectural quality of the spaces and which is not part of the floor plan generation with the exception of balconies can come under special blocks.

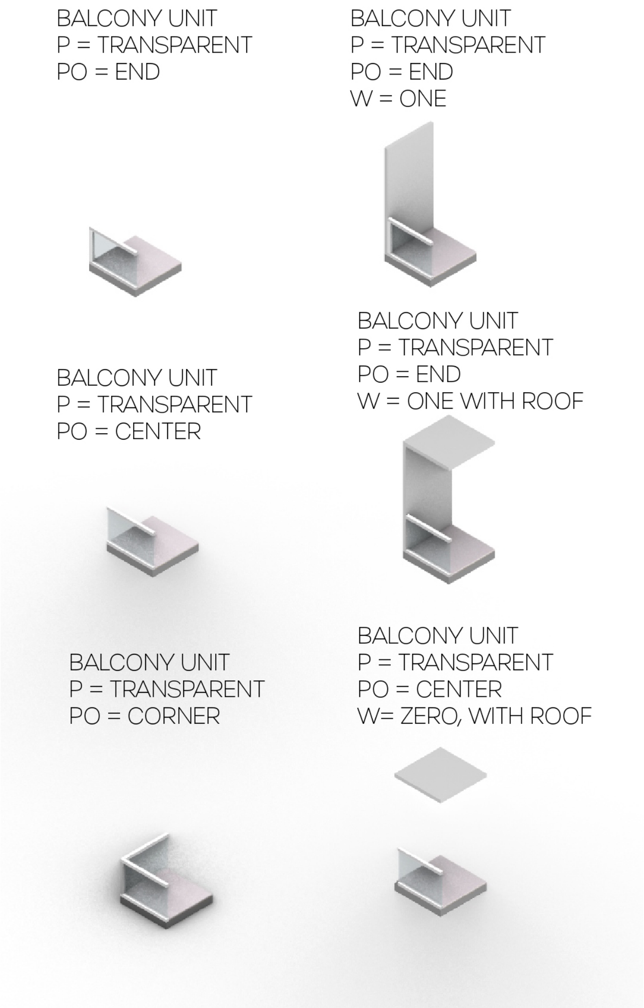


Figure 10.2: Special Blocks for 2d Configuration

In the images [10.3]below all the blocks created can be seen.

Step2: Block information input:
Along with the creation of blocks the information stored in each one of the blocks also needs to be filled. The level of information depends on what kind of dashboards or feedback the stakeholders expect. The dashboard will keep on updating the necessary details when the blocks are assigned to inform the user about the selected performance parameters. Based on that the user can modify/ create new blocks for meeting the goals.

These levels of information can be embedded in the form of voxel surface ids. Each voxel can be divided into the required number of surfaces and each surface could be given a id. Information can be classified into types and can be feed-ed into the id reserved for the surface.This will keep all the

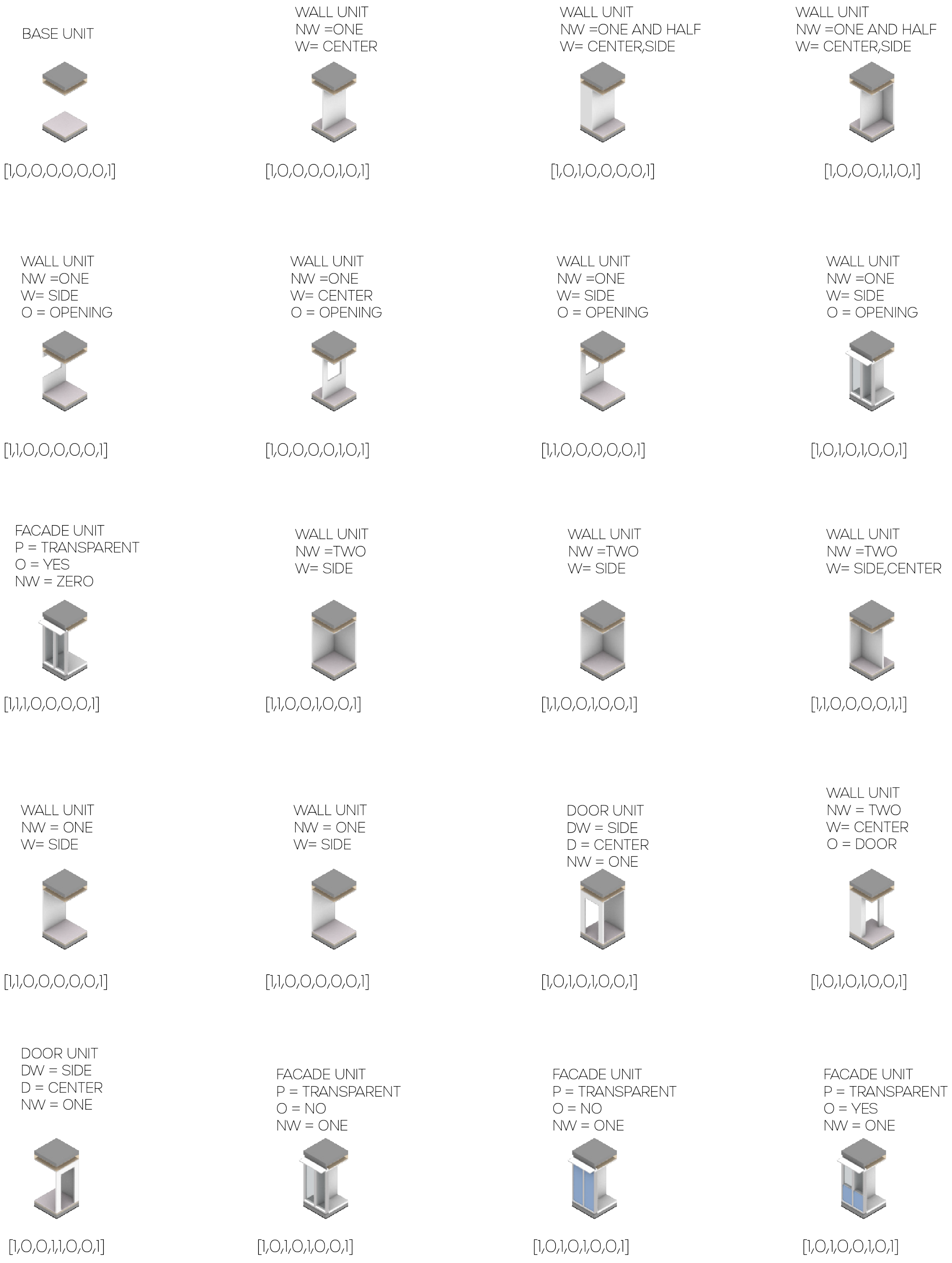


Figure 10.3: All the blocks created for the problem

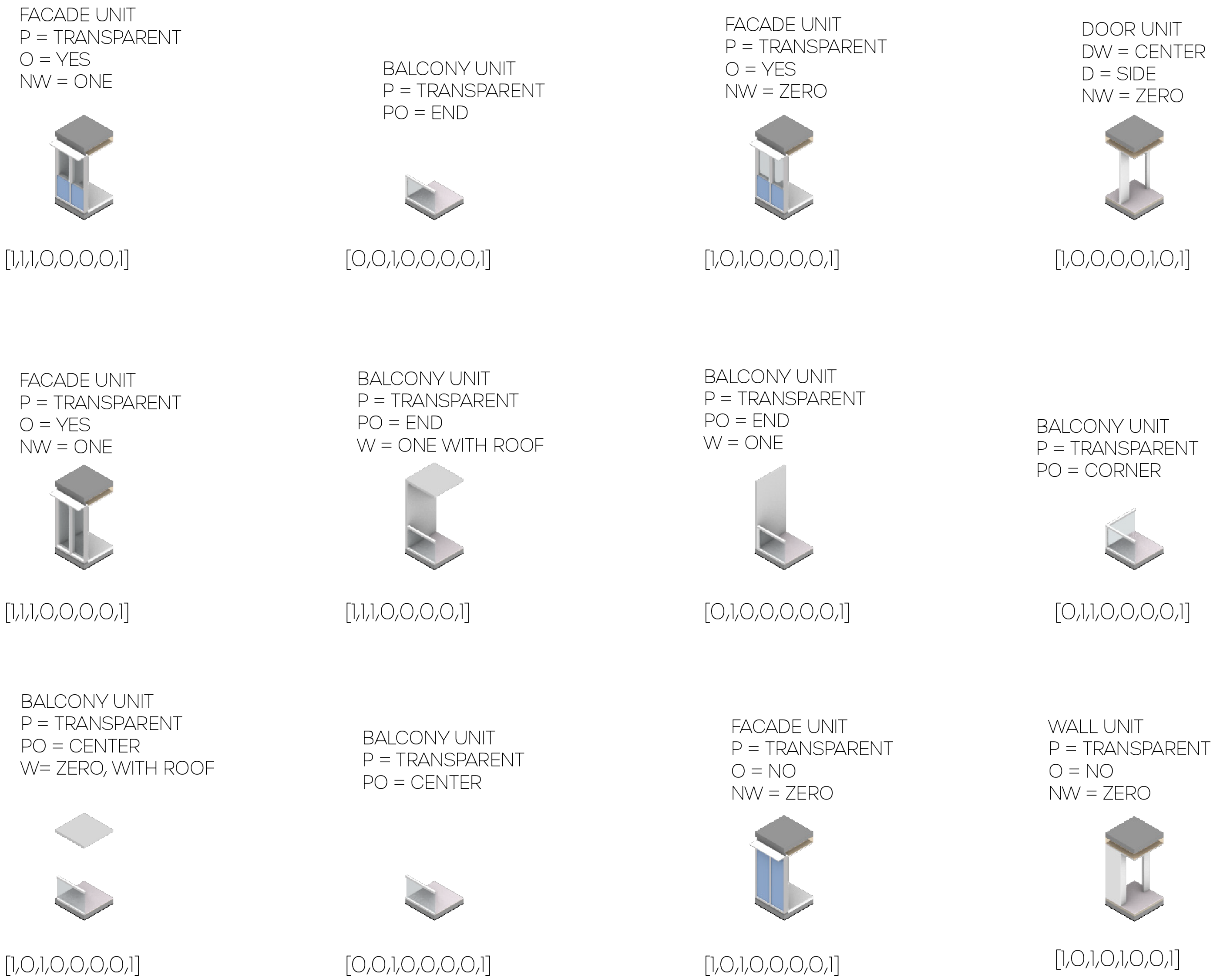


Figure 10.4: All the blocks created for the problem

information in its basic numeric form which can be read by a computer modelling software to render the necessary image or assign the necessary block. The example below can showcase the concept of surface id embedding in voxels:

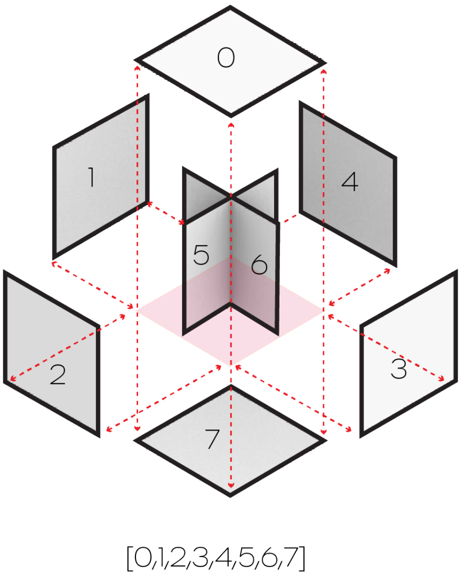


Figure 10.5: Voxel surface id

Step3: Block Assignment:
The blocks are assigned in the voxelated grid either room by room or by simply enumerating through the array in row wise or column wise manner. The blocks are picked based on the location of the voxel in the floor plan. The voxels in the room clusters store information regarding their position (corner/intermediate/junction) and requirement like doors and windows based on which appropriate block is assigned.

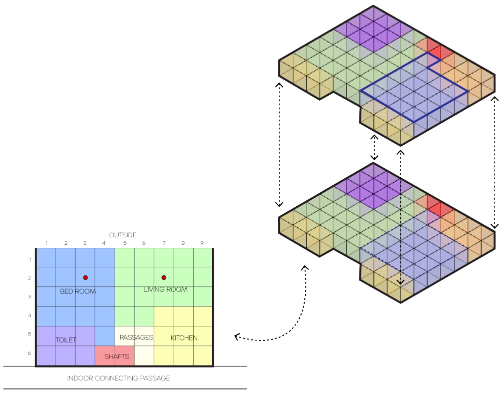


Figure 10.6: Voxel surface id

The information is stored in each voxel in a similar manner like the surface id but in this case the Boolean value represents if an element is present at the surface location or not.

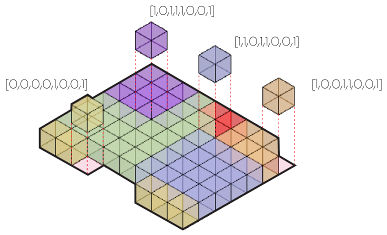


Figure 10.7: Voxel surface id

Alternatively in the manual process blocks can be picked manually and assigned to the voxels to create a layout.

Step4: Design Iterations and final selection:
In the last step of the configuration process based on the previous steps a feedback loop will be created which will indicate a lot of indexes for the choices made so far. These indices will be helpful for the user to decide if the resolution of the design detailing is satisfactory or not.if it is satisfactory then the configuration process will end and if it is not satisfactory then the previous steps can be repeated till a satisfactory solution is developed.

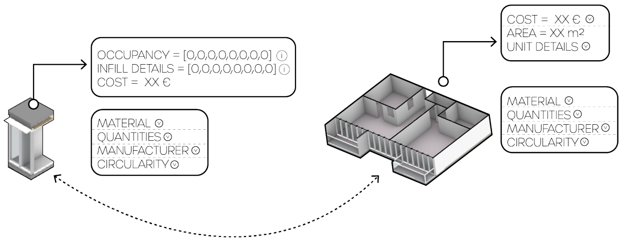


Figure 10.8: Feedback loop

10.0.3 Conclusions and Limitations:

The approach shown in this stage of the configuration problem indicates only a system for assigning of details to the voxels but does not talk about the complexity of generating the details. The voxel based assignment process is indeed a good solution for assigning modular 3D details, but will have its limitations if a non-modular custom 3D detailing is to be implemented.Finally as indicated in the introduction of this problem , a comprehensive solution to unit detailing problem will need a lot of background studies which lies out of the scope of the thesis project.

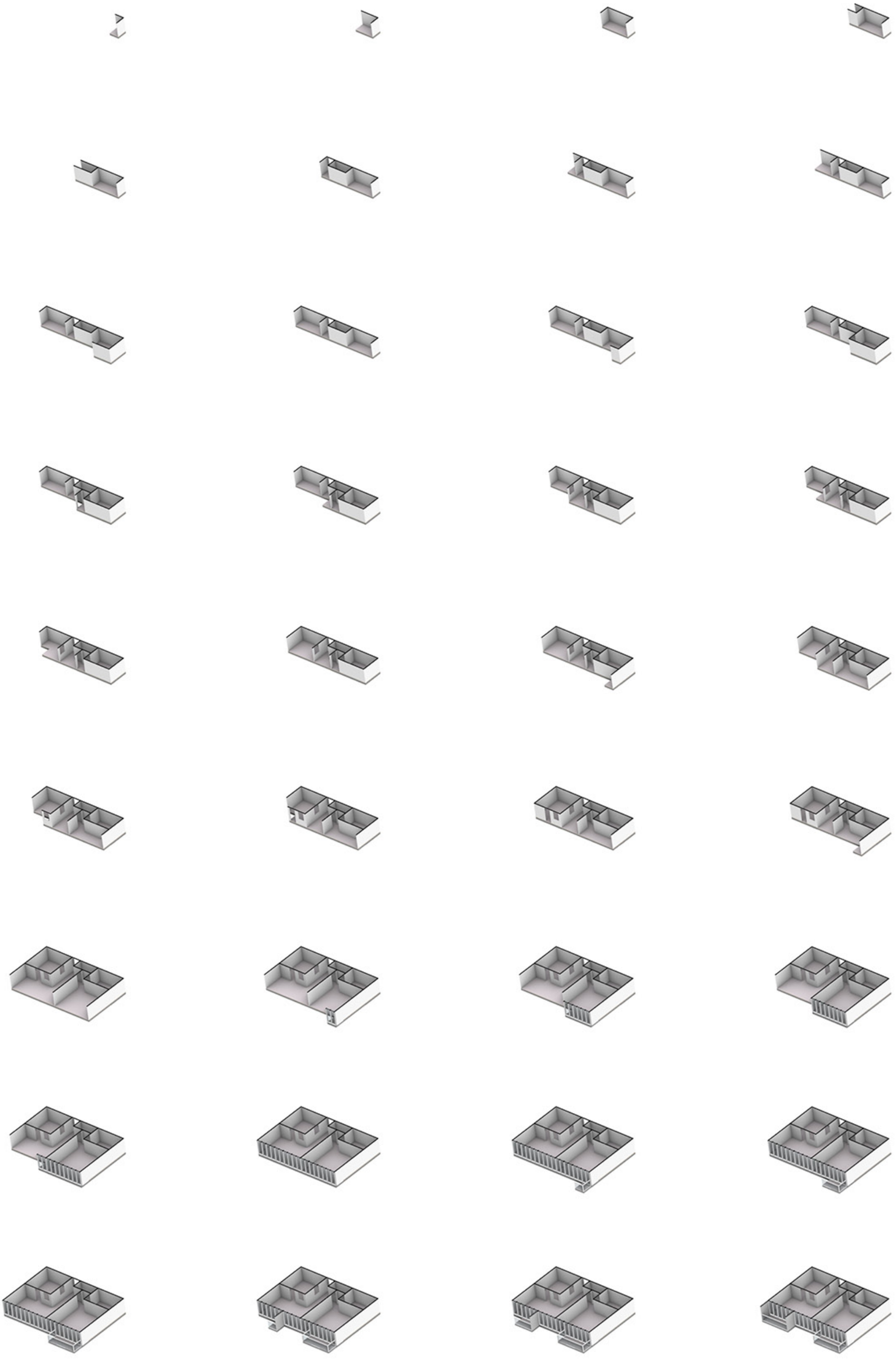


Figure 10.9: Unit Detailing step by step assignment

GEN-ARCH

PLATFORM FOR DEVELOPING ARCHITECTURAL CONFIGURATIONS USING GENERATIVE DESIGN METHODOLOGIES.

Conclusions | Reflections | Future Development

11 Conclusions

The final conclusions for the thesis would be to answer the research questions and summarise the solution presented for the problems identified in the beginning of the thesis.The following problems were identified at the beginning of the thesis:

1.The lack of a definitive computational methodology to tackle the problem of mass customization in generating Architectural configurations

A framework was proposed in the thesis which breaks down the massive complexity of the Architectural configuration problem in to smaller achievable goals and makes the goals of each level explicit and the input and the output of each stage of the problem clear. This helps in defining the problem clearly and makes the task of generating the computational solution easier.

The case of designing a mixed use building with a mix of housing and commercial functions at Buiksloterham was considered as a test scenario to formulate and validate a methodology. The Layout problem was effectively divided into five stages , Massing problem, Zoning problem, Unit assignment problem, Unit layout problem, Unit detailing problem each having its own set of goals and objectives. Four things were defined for each set of problems, The input needed, output generated, Computational methodology used, Participation framework in the methodology. Diving the problem into these four criteria makes the exploration process modular and transparent.

Various computational methodologies were explored in the thesis. The methodologies in a broad sense were divided into two categories namely, for generating design solutions and secondly for evaluation of the design solution. The Multi criteria decision analysis strategies were used in almost all the stages of the problem to evaluate the performance of the design solutions . For the generation of design methods like Multi-agent systems and agent based modelling were explored. Inspiration were taken from the solutions derived for the Facility layout planning problems,Operations research problems and Procedural content generation problems to derive unique solutions to address the nested set of configuration problems.

So finally to address the question of a definitive computational methodology and the sub question of Selection of an appropriate computational approach for solving the problem the answer is as follows:*There is no one algorithm or solution to the problem.* There have been attempts to solve this question of automated floor layouts for over fifty years and in most of the cases the authors have tried to address the complete problem at one go using one method. This makes most of the solutions very rigid and limited to certain cases. The solution needs to be adaptable and for that the Layout problem needs to be broken down into several problems to solve it efficiently. The process of selection of a computational methodology will depend on the nature of the problem but if the problem is well defined and clear then several methods can be explored to pick the suitable one.

Furthermore creation of this platform where the nature of problem is clear and definite the work of all the previous authors who have tried to address this problem can be unified and the user of the platform could be able to select the best alternative for the problem. This knowledge sharing and crowd sourced knowledge will lead to a collective digital intelligence which could become the definitive source for exploration of the various options for the mass customization process.

2.Lack of inclusiveness for the stakeholders in the design configuration process.

Co-design or inclusiveness in design was analysed by looking at various case studies in the open building movement. The Open building concept itself offers a sustainable way of participatory design as seen in the various examples. In the thesis after studying literature and performing various case studies about open building projects the stakeholder influence for each stage of the design was derived. Understanding which are the most relevant factors in the whole process for the stakeholders was the most essential part of the thesis. In the configuration method portrayed in the thesis at all the levels of the configuration problem the stakeholders could decide the desired performance goals of the problem and could participate in the process.

In the first stage of the Layout problem that is the massing problem the stakeholder involved were the municipal authorities with a primary aim of securing the city's interest in the development followed by the developer trying to maximise the profit by utilizing the full extents of the site and the Architect who was responsible for the design of the building massing. The participatory framework here was that of design validation. The Architect generates various design options for the massing considering the municipal authorities development regulations and the developers intentions. The developers and the Municipal authorities at this stage could mention the criteria in which they were interested in and the calculations would be done for the same. Based on the criteria of the developers and the authorities the architect could build a decision model to rank the alternatives designed in the process. The final selected output is a collaborative acceptance of the design options backed by traceable design steps and robust mathematical models.

In the Zoning problem the stakeholders involved are the developers and the consultants where the developers suggest the quantities of space needed for activities and the consultants can mention the closeness requirement between zones and also the necessary simulations for some zones. The Architect is responsible again to convert these requirements and adding the architectural criteria like closeness , form, shape to the zoning problem and develop a MCDA model which can generate the result for desirability of location of each zone in the massing model.

In the Unit assignment problem the architect and the developers are the stakeholders involved in the process. The develop mentions the area ratios needed for the various units and the range of volumes for units , since his primary interest is to sell the units of specific volumes and size. The Architect is responsible for placing the units in the respective zone where the spatial quality for them will be the maximum. Till this point the end user of the space is not involved.

In the final two stage of the Layout problem the end user or the participant who is going to occupy the space is involved. In the unit layout stage there is a gamified process of collaboration where the Architect acts as the game master setting up all the moves and the rules for the game and the participants who are the end users of the space have top make choices and moves to generate their preferred layout options. This process can lead to a mass collaboration process where only a handful of designers can cater to the requirement of a large pool of participants. Similarly in the unit detailing process. The architect has to first get involved with the building contractors and the building component manufacturers and generate all the necessary information for the unit detailing stage. This information is further processed in game moves and required feedback loops for the participants who can then make choices and moves to achieve their targeted goals.

Method Specific Conclusions:

1.Multi-criteria-decision-analysis.

MCDA was used in almost all the stages of the Layout problem. The use of MCDA methods is justifiable only when the decision making is purely based on values that can be measured. However architectural problems more often than not have an intangible element to it which also needs to be considered. The results coming out of the MCDA system should not be considered as absolute but as a guidelines for the Human designer to make decisions based on the tangible parts. The normalization method of TOPSIS was used as the preferred method of MCDA in most of the problems mainly because of its ability to compare quantities with different units and scales of measurement. However the MCDA method should always be chosen based on the nature of the problem.

2.Multi-Agent system.

The multi-agent system developed in the thesis was quite effective in generating the desired clusters for the zones in the zoning problem and the units in the unit assignment problem. However the system itself has certain drawbacks. In any multi-agent system the interaction between the agents is crucial and the negotiation part in the whole system was missing. Each agent was trying to achieve its own clustering goals which led to interesting cluster formations but for the requirement of a strict clustering form like a cuboid was very difficult to achieve because of it. If for every stage of the Agent based simulation the agents interacted and adapted to generating the best possible shape form for them then the improvement step proposed in the thesis would not be needed. The whole clustering problem can also be looked at from the perspective of cuboidal voronoi zoning. It would maintain the requirement of strict cuboidal clusters and the results from the same would be interesting to analyze.

3.Assignment problem.

Conversion of the zoning problem into an assignment problem generated good solutions when the assignment goals were evenly distributed between the agents. Which means more the agents better the result in terms of clustering. However the increase in the number of agents led to increase in the abstraction in the topological formation of the clusters which was often unacceptable inspite of them performing highly in achieving their clustering goals. The possible improvement in this would be the the improvement proposed in the multi-agent system itself . Linear assignment as a strategy was explored for zoning problems without any closeness requirements between the zones however Quadratic assignment would also be an interesting approach to look at when the zoning/clustering problem has closeness requirement.

4.Closeness rating method.

The TCR or the total closeness rating method was explored in the zoning/ clustering problems where there was a closeness requirement. It is loosely based on the 2d configuration approach for the facility layout planning problem in which spaces are places sequentially from the center based on their closeness requirement. This was interpreted as a 3d problem based on the same logic but the agent origin locations were based on the best possible locations available for them. The approach resulted in clusters having good closeness requirement satisfaction but the method doesn't always yield good results especially if there are similar competing agents in the system. An idea to improve this process would be to generate all the possible valid options for agent origin locations via constraint programming and then iterating though these possible options for a solution.

5.Rectangle packing method.

The rectangle packing method was used for the unit layout problem in the thesis and worked out quite well to produce results. The problem in its full sophistication when rotation , translation and variable dimensions is actually impossible to solve but the simplifications done makes the method converge into creating valid solutions. However the iteration power of the whole system is based on the rules/simplifications set in the system and the user should be careful so as to not put to many constraints in the generation process.Multi agent system /data tree methods like the kd tree splitting or the physically based method involving usage of physics simulations to achieve equilibrium state could also be explored as methods to solve the problem.Finally the adjacency and the element based information stored in each voxels in the unit detailing phase should be integrated in this method itself for making the unit detailing phase faster.

12 Reflection

The main objective of the research was to develop a Participatory design tool for mass customization and generative configurations.The problem of mass customization is an extremely relevant one in today's world where the world has to build millions of new homes and buildings.The research concluded in a new methodology for co-design where the human designer, the machine and the end user all contributed and collaborated to create customised buildings.

Problem solving process:
The process undertaken for solving this 3D-Layout problem was to fist understand the full complexity of the problem itself and then divide it down into smaller problems. The smaller problems were generalised and stripped off any domain specific qualities by defining the clear goals and objectives and expressing it in the universal language of mathematics. The main aim of this generalization was to tap into the relevant research and development happening in other disciplines which are involved in solving similar problems. In this search phase strategies/methods, tool kits were explored which have the potential to solve the problems and a list of possible strategies which could be used was prepared. This was followed by the prototyping stage where the tools and strategies were used to solve toy problems and results were derived.The strategies were modified and developed till a satisfactory results were obtained and then the toy problems were converted into real problems for a test case. After conclusive results were obtained for the test case the prototype was converted into a first release version for testing it out on multiple cases. This process is similar in essence with the goals of the master track and of the profession of a building technology expert where the building technology expert has to understand the various disciplines involved in Architecture and act as a bridge between Engineering and Design.

Effectiveness of the process:
The stage of computational design and especially generative design in architecture is at its stage of infancy where there is a lot for scope for research and development. I would conclude that the process taken for solving the problem was indeed effective since the approach allowed me to explore the research and developments from other disciplines like Operations research , Industrial engineering, Mathematics, Computer science, Computer gaming industry for solving certain types of problems. This interdisciplinary development opens up new possibilities for collaboration with experts in those particular domains since the communication would be more effective due to the conversion of nature of the problem from an abstract one to an explicit one.

The whole process was however restricted by the knowledge that I could gather in the limited time available for the thesis about the various discipline and the methods identified for the problem solving and would certainly lead to new explorations and improvements if collaborations are initiated with the domain experts in the particular field.

Mentor Feedback:
The mentors acted as my collaborators through the whole process. The conversion of the problem into more methodical and mathematical manner was done with my first mentor Dr.ir. P. Nourian and my third mentor ir. S. Azadi. Their feedback about formulating the questions and defining the goals for each stage helped me in finding and developing all the strategies seen in the thesis. Another important part of their feedback was about potentially avoiding the rabbit holes where the exploration of certain method would take a long time and discarding certain approaches because of their lack of flexibility towards solving the problem. The collaboration with my second mentor Prof.ir. Thijs (M.F.) Asselbergs was about the essential features required in the developed methodology/tool from an architects point of view, who is the main intended user of this method. His feedback about what is relevant led to the refinement of scope of the project and focused development on only the essential aspects of the problem.

Relation of the thesis to the master track and the master program:
The intention of the master track of building technology is to reduce the gap between architecture and engineering by integrating architectural design with technical disciplines. The methodology developed in this project uses the power of Design informatics to combine all the associated disciples of building technology to develop, data informed, integrated, sustainable architectural design options which is complimentary to the aim of the master track.The project also has a strong relevance to the overall master program of Architecture, Urbanism and Building Sciences. The built environment in the world is developing faster than

ever and most of the new buildings-built lack the access to good Architectural design. A platform for generative mass customisation will enable informed decision making for these buildings. It is crucial for these buildings to have that access to these technologies since they have the biggest impact on sustainability and well-being of the urban environment.

Scientific/Societal relevance :
Societal relevance: The process of Architectural design is most of the times driven by a single person or a small team of decision makers. The future inhabitants of the space don't get a chance to participate and decide on how the space is designed. The methodology proposed in the design gives all the stakeholders the opportunity to participate in designing the spaces they will live in and pushes for urban equality.

Scientific relevance: The use of decision support system for architecture is a novel one. Generally the architectural problems are formulated in an abstract manner where judging the design becomes very subjective. Using decision theory to evaluate the various aspects of design leads to a transparent and collaborative evaluation of architectural design. Secondly by interpreting Architectural problems as mathematical or computer science problems like the assignment problem , packing problems etc. opens up the possibilities of generating design solutions which would have not been possible by traditional methods. One example of this is assigning of zones based on a MCDA based decision result for multiple criteria, this traditionally by a manual process is impossible to achieve. Using the power of the machines to iterate though the vast combinatorial search space that encompasses any architectural configuration problem itself leads to possibilities of generating and evaluating mass solutions for a problem. Conversion of the problem of configuration into an adaptation one by a methodical process opens up new avenues for design.

Ethical issues :
The biggest ethical issue with any generative design project is that will the tool developed in the project make the role of a human designer irrelevant ? The intention throughout the project is not to make the human designers role redundant in the whole process. One of the biggest learning's in the thesis was that the intangible parts of the design process like aesthetics is very difficult to digitise and should be left to the human designer. The idea is to make the repetitive non intellectual tasks in the design process redundant. Providing control over the generation of the design output was the main essence of all the methods developed in the thesis. Certain design tasks which are seemingly impossible by a human designer to achieve were digitised with all the controls over the generation maintained by the human designer. The idea for this project as well as for the future development is co-design and not replacement of the human designer.

Self development:
The complete thesis project was built in the python programming language and throughout the process I was able to develop my python programming skills. The major advantage of learning programming skills in python is that the development community is very strong and opens up the chance of collaboration and discovery of new techniques and libraries of tools which can be used to solve the problems. There was also a significant development in my own understanding of architecture specifically architectural planning and the methodical process followed where each step in design was questioned to avoid the logical leap in design led me to develop a process for design where the steps are traceable and the design options generated were comparable.

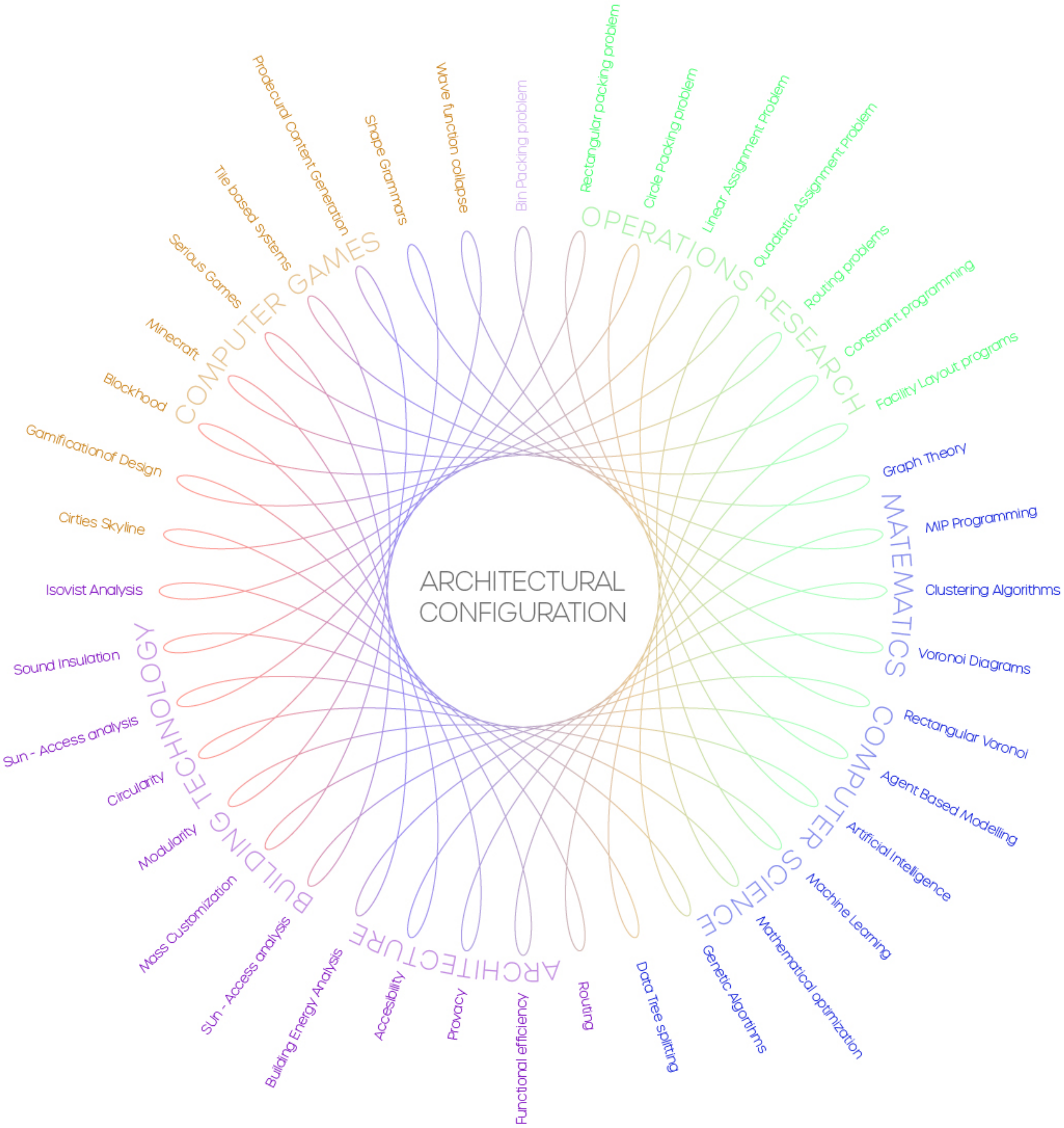


Figure 12.1: Various streams of knowledge involved in the configuration process

13 Future Developments

The configuration problem as stated in the thesis is a set of nested problems which are linear in scale ranging from the urban level massing to the smallest scale of configuring the details of the rooms inside the units. However these problems and the solutions proposed for them can even be used individually to solve a problem. For example if the configuration problem involves re-using or renovating an existing building then the data from the existing site conditions and the building will act as a base so the level of problem would start from the zoning problem itself since the existing building will be the base massing. Hence, developing a platform would be beneficial where a wide variety of projects at various stages could be configured.

The development of an online configurator where the users can make an account and start building their configurations would be an ideal start towards development of the platform. There have been a lot of recent developments in the online configuration platforms which utilize the existing ecosystem of visual programming languages like grasshopper for rhinoceros to develop configurators for parametric design. Pushing these platforms for a wider integration of tools and technologies would be needed for the development of this platform. Python as a programming language has been used extensively for developing the various methods showcased in the project . The primary advantage of using python is the massive open source development community which contributes towards the open sciences.

Once the platform is built then the development would be guided by the users of the platform themselves. The more people use this platform to develop their projects the more the platform would be developed via feature requests and user contributions. Keeping the project open source can lead to open development ideas where users from various projects can use the ideas used in other projects for developing their own projects. This common pool of knowledge will lead to better projects and better cities around the world.

Where would this platform stand with respect to the currently available products in the market ?
Currently there a lot of online configurator platforms available that use the technologies of artificial intelligence and generative design to produce building options some of the selected ones which are developing rapidly are as follows:

Spacemaker : Spacemaker AI is an autodesk company which developed a cloud based collaborative tool to develop real estate sites. The tool empowers the users to use data available for the site to generate thousands of massing design options and run simulations/ optimization on them for decision making. The tool is currently primarily based on an urban level and tries to generate various indices like energy consumption and costing and built up area to analyse options.

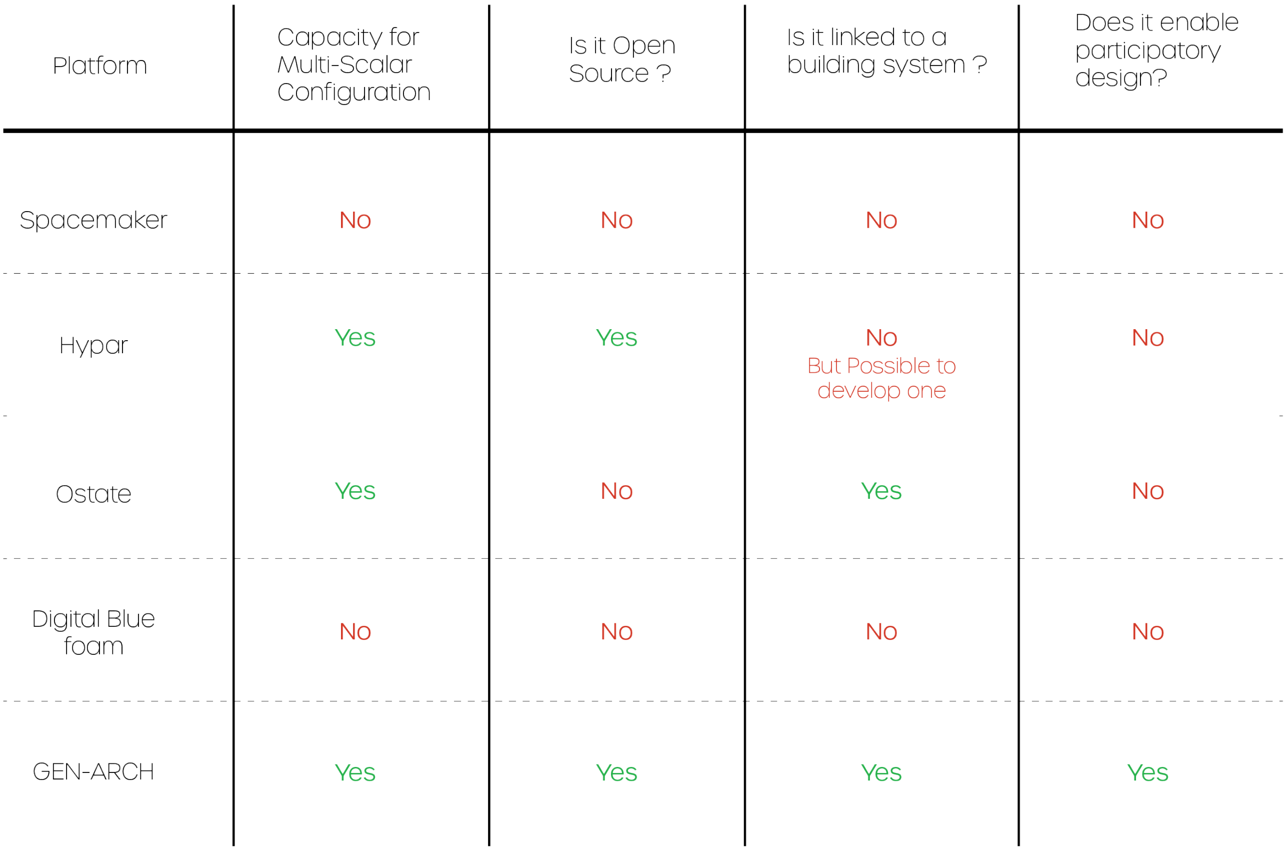
Hypar: Hypar is a customizable platform for generating architectural configurations. Hypar has functions some of which are developed by the developers of the product themselves and others are community driven components. These generative components range from generating the site to generating structure. The community driven open source approach makes this a good platform for developing new ideas and concepts in generative design.

OMRT (Ostate) : Omrt's Ostate is a configurator that can deal with urban level problems as well as generate potential floor plans in massing models. It also has the capability to run environmental simulations for the floor plans to determine the better options. The aspect of materiality is also embedded in the tool to a certain extent where the 3d products can be integrated into the models themselves.

Digital Blue foam: Digital Blue foam is a Singapore based company who has developed a configurator which can design urban level massing or a building level massing. The configurator has a gamified appearance and has a similar use case as the configurator developed by spacemaker.ai.

GEN-ARCH: If the idea for the platform as proposed in this thesis is compared to the rest of the products it has several advantages over them. The biggest advantage is the multi-scalar nature of the platform where not only the urban level massing problems can be resolved but even small scale room detailing can be solved. Secondly the platform proposed is an open source one where collective design intelligence can be built thus making it more diverse than other platforms. Thirdly it is linked to an existing style of design of open buildings and can potentially link to the vast information about projects and research and with material and construction systems which are inherently pushing for circularity and sustainability. Finally the whole idea of participatory design or democratising design is a unique one and is not part of the agenda of any of the platforms till now.

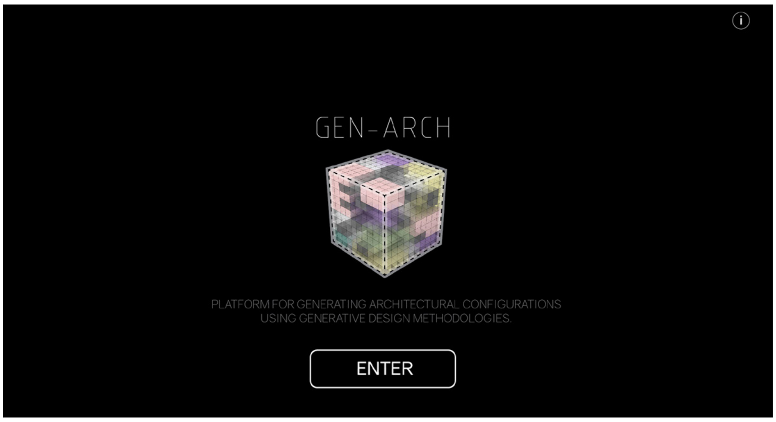
The table shows the comparison of the proposed platform with the other existing platforms.



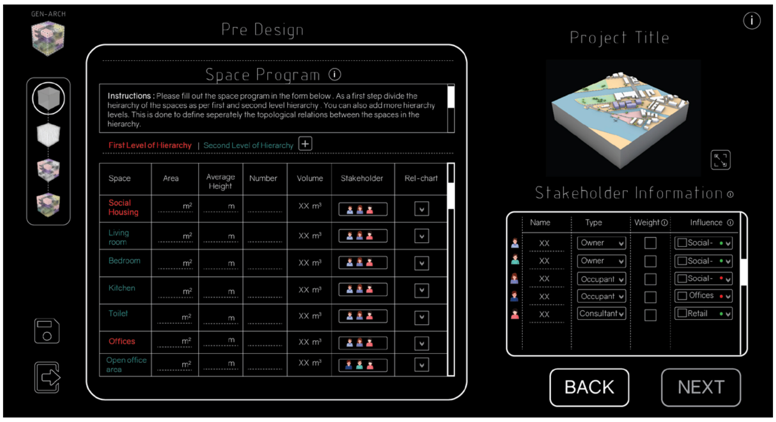
Platform	Capacity for Multi-Scalar Configuration	Is it Open Source ?	Is it linked to a building system ?	Does it enable participatory design?
Spacemaker	No	No	No	No
Hypar	Yes	Yes	No But Possible to develop one	No
Ostate	Yes	No	Yes	No
Digital Blue foam	No	No	No	No
GEN-ARCH	Yes	Yes	Yes	Yes

Figure 13.1: Comparison of all the platforms with GEN-ARCH

Finally Some initial ideas about the user interface of the platform can be seen in the images below where an user interface for a potential online configurator is designed which goes from the massing scale to the unit detailing scale in various dashboards.



PLATFORM FOR GENERATING ARCHITECTURAL CONFIGURATIONS USING GENERATIVE DESIGN METHODOLOGIES.



Space Program

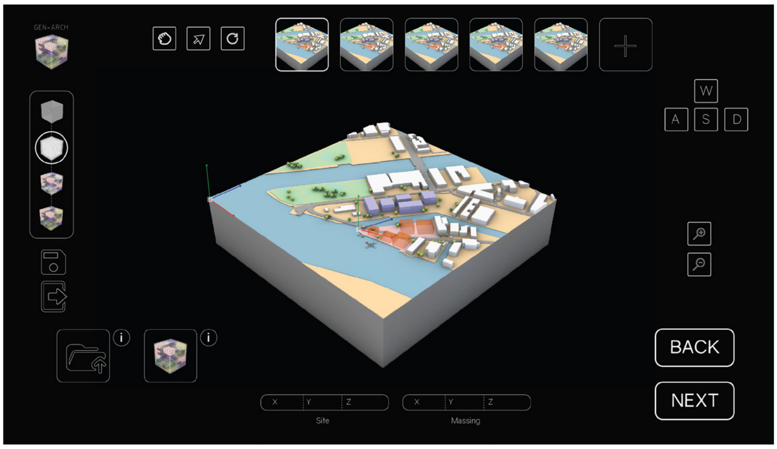
Instructions : Please fill out the space program in the form below . As a first step divide the heirarchy of the spaces as per first and second level hierarchy . You can also add more hierarchy levels. This is done to define seperately the topological relations between the spaces in the hierarchy.

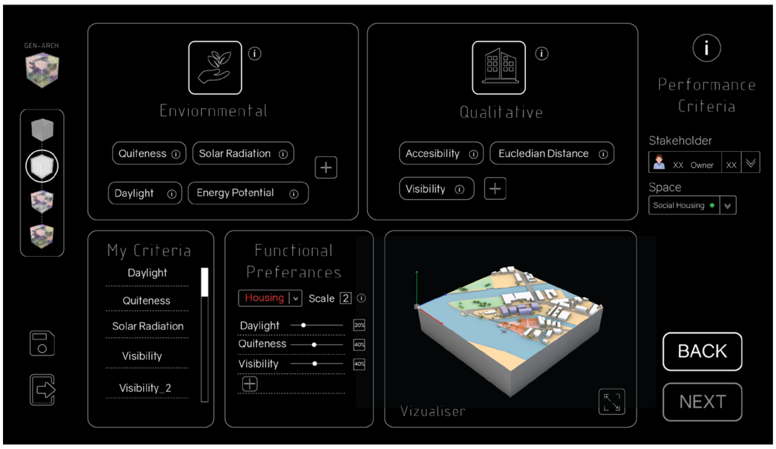
First Level of Hierarchy | Second Level of Hierarchy

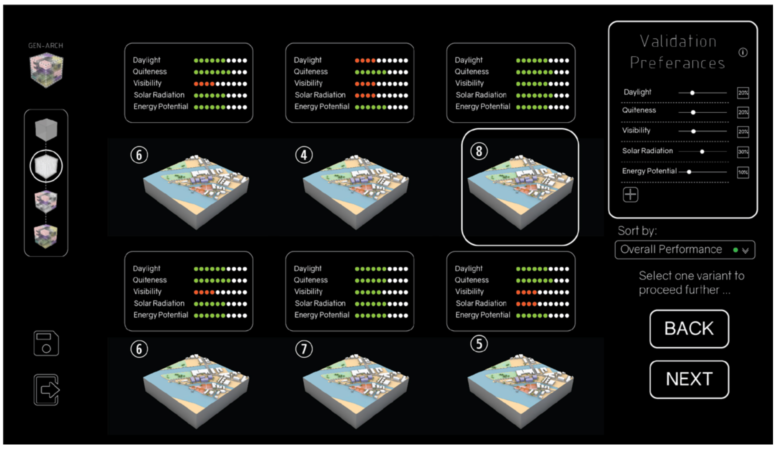
Space	Area	Average Height	Number	Volume	Stakeholder	Rel-chart
Social Housing	m²	m		XX m³		
Living room	m²	m		XX m³		
Bedroom	m²	m		XX m³		
Kitchen	m²	m		XX m³		
Toilet	m²	m		XX m³		
Offices	m²	m		XX m³		
Open office area	m²	m		XX m³		

Stakeholder Information

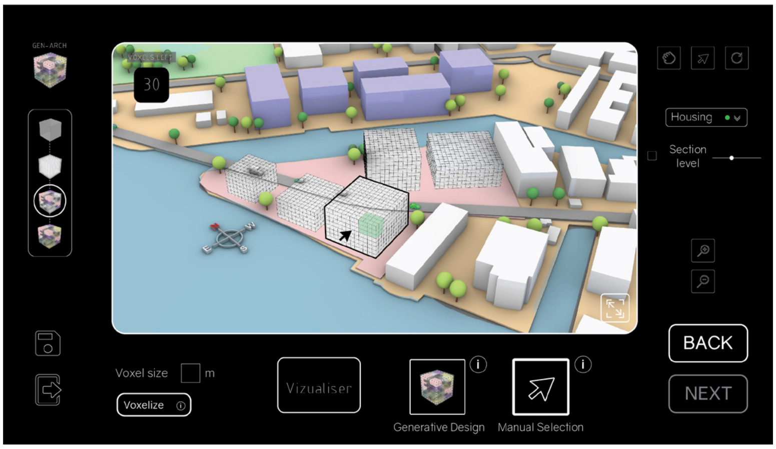
Name	Type	Weight	Influence
XX	Owner	□	□ Social-
XX	Owner	□	□ Social-
XX	Occupant	□	□ Social-
XX	Occupant	□	□ Offices
XX	Consultant	□	□ Retail

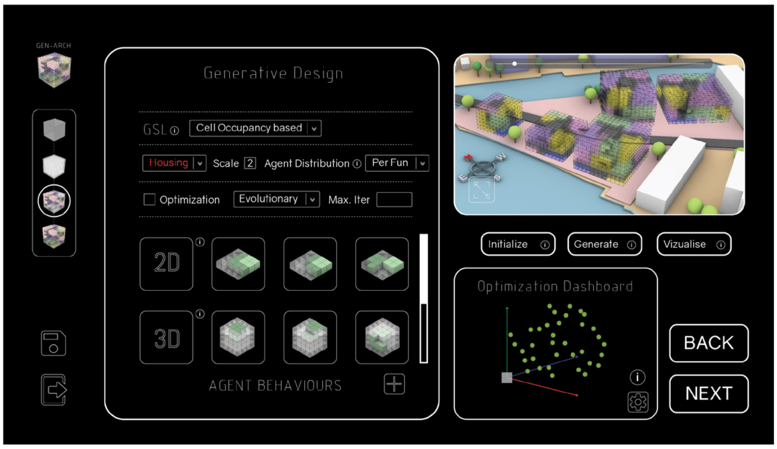


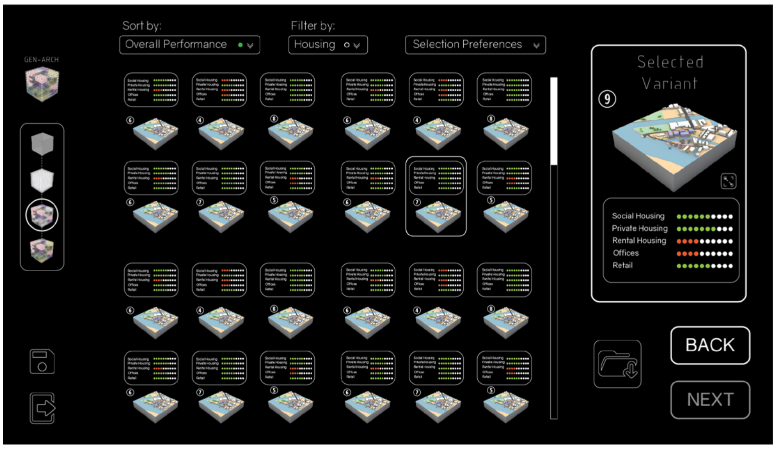




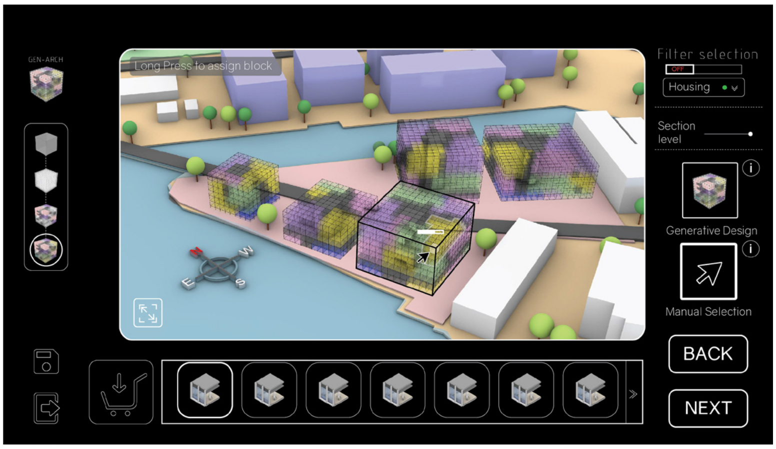
Select one variant to proceed further ...

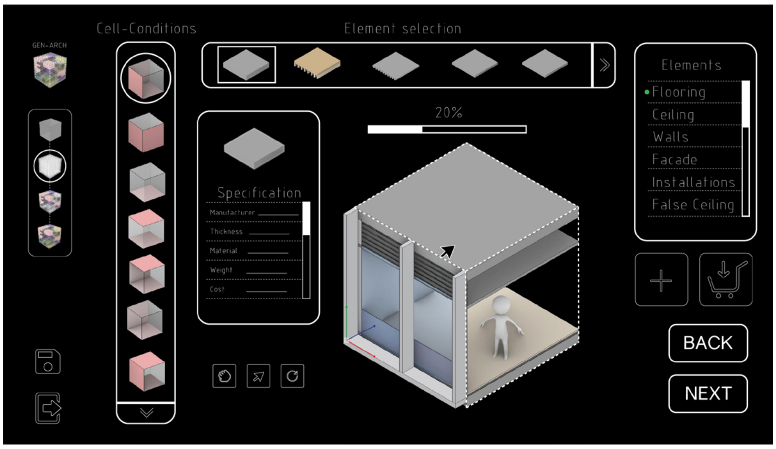






Social Housing	●●●●●●●○○○○
Private Housing	●●●●●●●○○○
Rental Housing	●●●●○○○○○○
Offices	●●●●○○○○○○
Retail	●●●●●●○○○○





Specification

Manufacturer ________

Thickness ________

Material ________

Weight ________

Cost ________

14 Appendix

LIST OF PYTHON LIBRARIES USED IN THE PROJECT:

1.Visualization

PyVista:

PyVista is a helper module for the Visualization Toolkit (VTK) that takes a different approach on interfacing with VTK through NumPy and direct array access. This package provides a Pythonic, well-documented interface exposing VTK's powerful visualization backend to facilitate rapid prototyping, analysis, and visual integration of spatially referenced datasets.

Features used:

Volume Rendering
Grid plotting
Mesh Plotting
Widgets
`https://docs.pyvista.org/`

2.Meshing

Trimesh:

Trimesh is a pure Python (2.7-3.4+) library for loading and using triangular meshes with an emphasis on watertight surfaces. The goal of the library is to provide a full featured and well tested Trimesh object which allows for easy manipulation and analysis, in the style of the Polygon object in the Shapely library.

Features used:

Import and Export mesh objects
Determine if a mesh is watertight, convex, etc.
Determine if a point lies inside or outside of a well constructed mesh using signed distance
Determine intersection of rays with the mesh
`https://trimsh.org/index.html`

3.Enviornmental analysis

Ladybug Tools:

Ladybug Tools is a collection of free computer applications that support environmental design and education. Of all the available environmental design software packages, Ladybug Tools is among the most comprehensive, connecting 3D Computer-Aided Design (CAD) interfaces to a host of validated simulation engines. The LBT-Ladybug library was used which includes Collection of all Ladybug core Python libraries.

Features used:

Sun-Path
Load EPW
`https://pypi.org/project/lbt-ladybug/`

3.Multi-dimensional Array objects

Numpy:

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Features used:

Data types
Array creation
I/O with NumPy
Indexing
Broadcasting
Byte-swapping
Structured arrays
`https://numpy.org/doc/stable/user/index.html`

4.Topological structures for generative design

Topogenesis:

topoGenesis is an open-source python package that provides topological structures and functions for Generative Systems and Sciences for various application areas such as: generative design in architecture and built environment,generative spatial simulations,3D image processing,topological data analysis,machine learning.

Features used:

Geometry
Data structures
`https://topogenesis.readthedocs.io/`

5.Data analysis

Pandas:

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

Features used:

Data frame creation
csv , xlxs conversion.
`https://pandas.pydata.org/`

6.Multi-Criteria Decision Analysis

Sci-kit Criteria:

Scikit-Criteria is a collection of Multiple-criteria decision analysis (MCDA) methods integrated into scientific python stack. Is Open source and commercially usable..

Features used:

TOPSIS module
Weighted sum module
Weighted product module
`https://scikit-criteria.readthedocs.io/en/latest/index.html`

7.Network Analysis
Network X:
NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.
Features used:
Graph Creation
Connected components
Shortest distance algortihms like Flyod Warshall , Dijkstra's algorithm.
`https://networkx.org/`

8.Jupyter notebook Widgets
Ipywidgets :
ipywidgets, also known as jupyter-widgets or simply widgets, are interactive HTML widgets for Jupyter notebooks and the IPython kernel.Notebooks come alive when interactive widgets are used. Users gain control of their data and can visualize changes in the data.
Features used:
Widgets
`https://ipywidgets.readthedocs.io/en/latest/l`

9.Data visualisation
Dash and Plotly :
Dash is a productive Python framework for building web analytic applications.
Written on top of Flask, Plotly.js, and React.js, Dash is ideal for building data visualization apps with highly custom user interfaces in pure Python.
`https://dash.plotly.com/introduction`

10.Machine learning
Sci-kit Criteria:
Machine learning library built on numpy, Scipy and matplotlib
Features used:
K-means clustering algorithm
`https://scikit-learn.org/stable/index.html`

11.Operations research
Google OR Tools:
OR-Tools is an open source software suite for optimization, tuned for tackling the world's toughest problems in vehicle routing, flows, integer and linear programming, and constraint programming.
Features used:
MIP solver
SCIP Solver
Constraint solver.
`https://developers.google.com/optimization`

Code developed in the project classified according to the chapters:

Link to the project Website : `https://computationaldesignworks.com/`

GitHub Repository Link for P5 Release: `https://github.com/adityasoman/GEN-ARCH.git`

GitHub Repository Link for WIP Enviornment: `https://github.com/adityasoman/Aditya_Graduation_Project_BT`

GitHub Repository Link for Latex Report: `https://github.com/adityasoman/GEN_ARCH_Report`

1.Massing problem
MCDA jupyter notebooks, Rhino+ Grasshopper models for generating simulation values:
`https://github.com/adityasoman/GEN-ARCH/tree/main/01.Massing_problem`

2.Zoning problem
Agent Behaviours, Enviornment Lattices, Desirability Lattices, Toy Problems, Buiksloterham Case solutions:
`https://github.com/adityasoman/GEN-ARCH/tree/main/02.Zoning_problem`

3.Unit assignment Problem
Enviornment Lattices, Desirability Lattices, ABM simulations, Rhino+ Grasshopper models for unit assignment:
`https://github.com/adityasoman/GEN-ARCH/tree/main/03.Unit_assignment_problem`

4.Unit Layout Problem
Rhino + GH models for floor plan enumeration, Jupyter notebook for voxel based approach:
`https://github.com/adityasoman/GEN-ARCH/tree/main/04.Unit_Layout_Problem`

5.Unit Detailing Problem
Rhino + GH models for Unit Detail 3d block assignment:
`https://github.com/adityasoman/GEN-ARCH/tree/main/05.Unit_detailing_problem`

References

architecten, A. (n.d.). *Flexibel casco met maatwerkappartementen in het schetsblok.* Retrieved from https://www.ana.nl/portfolio-item/het-schetsblok/

Arvin, S. A., & House, D. H. (2002). Modeling architectural design objectives in physically based space planning. *Automation in Construction, 11*(2), 213-225. Retrieved from https://www.sciencedirect.com/science/article/pii/S0926580500000996 (ACADIA '99) doi: https://doi.org/10.1016/S0926-5805(00)00099-6

Azadi, S., & Nourian, P. (2020, August). *shervinazadi/topogenesis: Initial pre-release.* Zenodo. Retrieved from https://doi.org/10.5281/zenodo.4006514 doi: 10.5281/zenodo.4006514

Bilbao-Terol, A., Arenas-Parra, M., Cañal-Fernández, V., & Antomil-Ibias, J. (2014, dec). Using TOPSIS for assessing the sustainability of government bond funds. *Omega (United Kingdom), 49,* 1–17. doi: 10.1016/j.omega.2014.04.005

blockbyblock. (n.d.). *Rebuilding tradition in the kathmandu valley.* Retrieved from https://www.blockbyblock.org/projects/kathmandu

Block'hood. (n.d.). Retrieved from https://www.plethora-project.com/blockhood

Cabral, J. B., Luczywo, N. A., & Zanazzi, J. L. (2016). Scikit-criteria: Colección de métodos de análisis multi-criterio integrado al stack científico de Python. In *Xlv jornadas argentinas de informática e investigación operativa (45jaiio)- xiv simposio argentino de investigación operativa (sio) (buenos aires, 2016)* (pp. 59–66). Retrieved from http://45jaiio.sadio.org.ar/sites/default/files/Sio-23.pdf

Ching, F. D. K. (1979). *Architecture : form, space order / francis d. k. ching* [Book]. Van Nostrand Reinhold New York.

Cities skylines - cities: Skylines. (2020, Dec). Retrieved from https://www.epicgames.com/store/en-US/product/cities-skylines/home

Dawson-Haggerty et al. (n.d.). *trimesh.* Retrieved from https://trimsh.org/

Daylight analysis in the design process. (n.d.). Retrieved from https://www.spacemakerai.com/blog/daylight-analysis-in-the-design-process

Dijkstra's shortest path algorithm — Greedy Algo-7. (2020). Retrieved from https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/

Dino, I. G. (2016). An evolutionary approach for 3D architectural space layout design exploration. *Automation in Construction, 69,* 131–150. Retrieved from http://www.sciencedirect.com/science/article/pii/S0926580516301005 doi: https://doi.org/10.1016/j.autcon.2016.05.020

Du, T., Turrin, M., Jansen, S., van den Dobbelsteen, A., & Fang, J. (2020). Gaps and requirements for automatic generation of space layouts with optimised energy performance. *Automation in Construction, 116,* 103132. Retrieved from http://www.sciencedirect.com/science/article/pii/S0926580519307496 doi: https://doi.org/10.1016/j.autcon.2020.103132

García-Cascales, M. S., & Sánchez-Lozano, J. (2013, 05). Geographical information systems (gis) and multi-criteria decision making (mcdm) methods for the evaluation of solar farms locations: Case study in southeastern spain. *Renewable and Sustainable Energy Reviews, 24,* 544. doi: 10.1016/j.rser.2013.03.019

Ghaffarian, M., Fallah, R., & Jacob, C. (2018). Organic architectural spatial design driven by agent-based crowd simulation. In *Proceedings of the symposium on simulation for architecture and urban design.* San Diego, CA, USA: Society for Computer Simulation International.

Göbel, R. D. ·. S., & Wiemeyer, W. E. ·. J. (2016). *Serious Games Foundations, Concepts and Practice.*

Granados-López, D., Díez-Mediavilla, M., Dieste-Velasco, M. I., Suárez-García, A., & Alonso-Tristán, C. (2020, apr). Evaluation of the Vertical Sky Component without Obstructions for Daylighting in Burgos, Spain. *Applied Sciences, 10*(9), 3095. Retrieved from https://www.mdpi.com/2076-3417/10/9/3095 doi: 10.3390/app10093095

Guo, Z., & Li, B. (2017). Evolutionary approach for spatial architecture layout design enhanced by an agent-based topology finding system. *Frontiers of Architectural Research, 6*(1), 53–62. Retrieved from http://www.sciencedirect.com/science/article/pii/S2095263516300565 doi: https://doi.org/10.1016/j.foar.2016.11.003

Habraken, N. J. (1961). *De dragers en*

de mensen : het einde van de massawoningbouw. Amsterdam : Scheltema en Holkema. Retrieved from http://lib.ugent.be/catalog/rug01:001496451

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, & J. Millman (Eds.), *Proceedings of the 7th python in science conference* (p. 11 - 15). Pasadena, CA USA.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020, September). Array programming with NumPy. *Nature*, *585*(7825), 357–362. Retrieved from https://doi.org/10.1038/s41586-020-2649-2 doi: 10.1038/s41586-020-2649-2

(n.d.). Gemante Amsterdam. Retrieved from https://www.amsterdam.nl/projecten/buiksloterham/

Ibarra, D., & Reinhart, C. (2011, jan). Solar availability: A comparison study of six irradiation distribution methods. *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association*, 2627–2634.

IESVE. (n.d.). Retrieved from https://help.iesve.com/ve2018/sky_component_vertical_sky_component.htm

ipywidgets. (n.d.). Retrieved from https://ipywidgets.readthedocs.io/en/latest/index.html

Jati, N. P., Rahayu, A. D. I., Salsabila, S. E., & 'Azzam, A. (2020, dec). Facility layout design with corelap algorithm for educational tour. *IOP Conference Series: Materials Science and Engineering*, *982*, 012060. Retrieved from https://doi.org/10.1088/1757-899x/982/1/012060 doi: 10.1088/1757-899x/982/1/012060

Kleineberg, M. (2019, Jan). *Infinite procedurally generated city with the wave function collapse algorithm.* Retrieved from https://marian42.de/article/wfc/

Koltsova, A., Tuncer, B., & Schmitt, G. (2013). *Visibility Analysis for 3D Urban Environments* (Vol. 2).

Li, D. H. W., Cheung, G. H. W., Cheung, K. L., & Lam, J. C. (2009). Simple method for determining daylight illuminance in a heavily obstructed environment. *Building and Environment*, *44*(5), 1074–1080. Retrieved from http://www.sciencedirect.com/science/article/pii/S0360132308001893 doi: https://doi.org/10.1016/j.buildenv.2008.07.011

Li, S., Frazer, J., & Tang, M. (2000). A constraint based generative system for floor layouts..

Lobos, D., DannyAU . Donath. (2010, 2019/7/26). The problem of space layout in architecture: A survey and reflections. *Arquiteturarevista*(2 LA - Español). Retrieved from https://www.redalyc.org/articulo.oa?id=193617358005

Macal, C. M. (2016). Everything you need to know about agent-based modelling and simulation. *Journal of Simulation*, *10*(2), 144–156. Retrieved from https://doi.org/10.1057/jos.2016.7 doi: 10.1057/jos.2016.7

Michalek, J., Choudhary, R., & Papalambros, P. (2002). Architectural layout design optimization. *Engineering Optimization*, *34*(5), 461-484. Retrieved from https://doi.org/10.1080/03052150214016 doi: 10.1080/03052150214016

Molenvilet project. (n.d.). Frans van der Werf. Retrieved from http://www.vdwerf.nl/molenvliet.html

Np hard and np complete problem. (2021, Feb). Retrieved from https://www.geeksforgeeks.org/difference-between-np-hard-and-np-complete-problem/

Paul Rodgers, M. G. (2020). *No Title.* Retrieved from https://www.spacemakerai.com/blog/daylight-analysis-in-the-design-process

PR, T. (1999). The sensitivity of room daylight to sky brightness. *Architectural Science Review, 42(2):129–.*

Puspitasari, A. W., & Kwon, J. (2020, jul). A reliable method for visibility analysis of tall buildings and skyline: a case study of tall buildings cluster in Jakarta. *Journal of Asian Architecture and Building Engineering*, 1–12. Retrieved from https://doi.org/10.1080/13467581.2020.1787839 doi: 10.1080/13467581.2020.1787839

Rem Koolhaas. (2004). *Content*. Taschen.

Robinson, D., & Stone, A. (2004). Irradiation modelling made simple: the cumulative sky approach and its applications..

Rocha, J. (2017). Introductory Chapter: Multi-Agent Systems. In I. Boavida-Portugal (Ed.), (p. Ch. 1). Rijeka: IntechOpen. Retrieved from https://doi.org/10.5772/intechopen.70241 doi: 10.5772/intechopen.70241

Roudsari, M. S., & Pak, M. (n.d.). Proceedings of the 13th international ibpsa conference.. Retrieved from http://www.ibpsa.org/proceedings/BS2013/p_2499.pdf

Sanchez, J. .-T. A. T. T. (2021). *Architecture for the commons : participatory systems in the age of platforms LK - https://tudelft.on.worldcat.org/oclc/1138996774* (NV - 1 o ed.). abingdon, Oxon ;: Routledge. Retrieved from https://www.taylorfrancis.com/books/9780429432118

S. Das, A. H. J. H. D. D., C. Day. (n.d.). Space plan generator: rapid generation evaluation of floor plan design options to inform decision making journal =.

Sicart, M. (2008). Defining game mechanics.

Stuart Russell, P. N. (2010). *Artificial Intelligence: A Modern Approach, 3rd Edition*. Pearson. Retrieved from https://www.pearson.com/us/higher-education/program/Russell-Artificial-Intelligence-A-Modern-Approach-3rd-Edition/PGM156683.html

Sullivan, C. B., & Kaszynski, A. (2019, may). PyVista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (VTK). *Journal of Open Source Software*, *4*(37), 1450. Retrieved from https://doi.org/10.21105/joss.01450 doi: 10.21105/joss.01450

Superlofts. (n.d.). Retrieved from https://superlofts.co/

Sweetser, P., & Wyeth, P. (2005, jul). GameFlow: A Model for Evaluating Player Enjoyment in Games. *Computers in Entertainment*, *3*, 3. doi: 10.1145/1077246.1077253

Takizawa, A., Miyata, Y., & Katoh, N. (2014). *ENUMERATION OF FLOOR PLANS BASED ON ZERO-SUPPRESSED BINARY DECISION DIAGRAM.*

Takizawa, A., Miyata, Y., & Katoh, N. (2015, mar). Enumeration of Floor Plans Based on a Zero-Suppressed Binary Decision Diagram. *International Journal of Architectural Computing*, *13*, 25–44. doi: 10.1260/1478-0771.13.1.25

Thibaut Abergel, B. D., & Dulac, J. (2017). *Towards a zero-emission, efficient, and resilient buildings and construction sector* (Tech. Rep.). Retrieved from https://www.worldgbc.org/sites/default/files/UNEP188{_}GABC{_}en{%}28web{%}29.pdf

Ur Rehman, A., Awuah-Offei, K., Baker, D., & Bristow, D. (2019). *EMERGENCY EVACUATION GUIDANCE SYSTEM FOR UNDERGROUND MINERS.*

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi: 10.1038/s41592-019-0686-2

Wątróbski, J., Jankowski, J., Ziemba, P., Karczmarczyk, A., & Zioło, M. (2019, jul). Generalised framework for multi-criteria method selection. *Omega (United Kingdom)*, *86*, 107–124. doi: 10.1016/j.omega.2018.07.004

W.Huang, H. (n.d.). Architectural drawings recognition and generation through machine learning.. (English) journal =.

Www.remaudiology.com. (n.d.). *No Title.* Retrieved from https://www.remaudiology.com/surprising-levels-everyday-sounds/