Algorithmic Settlements | Modeling Informal Settlement Through Automated Generative Design

Ben Anderson-Nelson

Follow this and additional works at: https://surface.syr.edu/architecture_theses

Part of the Architectural Engineering Commons, Architectural Technology Commons, and the Urban, Community and Regional Planning Commons

Recommended Citation

This Thesis is brought to you for free and open access by the School of Architecture Dissertations and Theses at SURFACE. It has been accepted for inclusion in Architecture Senior Theses by an authorized administrator of SURFACE. For more information, please contact surface@syr.edu.
Algorithmic Settlements
"modeling informal settlements through the use of generative algorithmic design"

Ben Anderson-Nelson
Primary Advisor: Brian Lonsway
Secondary Advisor: Amber Bartosh
The position of the architect when designing is to arbitrate which information is relevant and which is not, and to do so across a broad spectrum of fields. Considering this, Christopher Alexander claimed as long ago as 1964, that “design problems are reaching insoluble levels of complexity.” This thesis focuses on informal settlement growth and how architects can investigate growth as a part of master-planning new housing. Drawing on case studies of settlements, video game logics, and existing architectural tools, a tool was developed to study the growth of settlements. This tool is based on cellular automata, a spatial and algorithmic method of computer modeling based on specific rules. The rules themselves have been developed to model settlement growth from both single house and neighborhood level as accurately as possible. The final result is a visual representation of hypothetical additions to homes over a period of time. The information produced can be used by architects to more holistically understand planning effects on settlement growth, and then to plan settlements with that informal growth in mind. [1]
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
</tr>
<tr>
<td>Case Studies</td>
</tr>
<tr>
<td>Informal Settlements</td>
</tr>
<tr>
<td>Game Environments</td>
</tr>
<tr>
<td>Algorithmic Models</td>
</tr>
<tr>
<td>Strategy</td>
</tr>
<tr>
<td>CA Introduction</td>
</tr>
<tr>
<td>Development/Methods</td>
</tr>
<tr>
<td>Testing and Results</td>
</tr>
<tr>
<td>Citations Reference Pages</td>
</tr>
</tbody>
</table>
As technical innovation occurs, the focus of those developing those fields becomes increasingly specific, allowing for increased productivity and rapid advancement. However, one challenge faced by architects in respect to this phenomena, is that in the design process, there are countless areas of expertise which might have effects on a particular project. The role of the architect is partly to choose which information is relevant and which is not, and to do so across a broad spectrum of fields. There is no way for an individual to be an expert in all fields which are relevant. Considering this, Christopher Alexander claimed as long ago as 1964, that ‘design problems are reaching insoluble levels of complexity.’ So, perhaps what designers need are specific tools at their disposal by which to approach design problems. Tools which aid in the organization, legibility and accessibility of information.

Research into various informal settlements uncovered intriguing instances where government housing and other master-planned projects acted as platforms for occupant driven growth. In one particular case, Cidade de Deus in Rio de Janeiro, a suburban plan for 10,000 people densified into an urban area for 30,000. And at the same time, the neighborhood transitioned from one of the most dangerous and ostracised neighborhoods into a growing lower-middle class community. The architectural formalism of the initial masterplan hybridized with the informal additions begs led to investigation of an architectural tool for designers to begin analyzing these conditions. Starting from an existing building plan, or a new master planning scheme, this thesis explores how informal growth can occur and the effects it might have on spatial planning.

The process for investigating this type of informal growth begins with determining how to translate conditions into more abstract simulation models. While investigating this topic, several potential simulation methods were studied, including cellular automata, shape grammars, genetic algorithms, agent based systems and self organized maps. The applicability of each to simulating informal settlements was not the same, and cellular automata was the option chosen. Cellular automata (CA) modeling is based on a grid of cells (typically but not necessarily square) which can have different states/appearances. The state of the cell in a grid is determined based on rule sets which take into account the neighboring cells to determine the selected cell state. There are one, two and three dimensional cellular automata, and in each the number of neighboring cells is different, two neighbors in 1D, eight neighbors in 2D and twenty-six neighbors in 3D. For this simulation 2D CA’s are used and rule sets are specific, meaning positions of neighbors are described and produce a specific described state change.

In order to use CA’s to model informal growth, Professor Brian Lonsway worked to develop a Processing script which could input specific rulesets for a 2D CA. His knowledge of programming allowed for what would have been time intensive and unrelated to the actual investigation, to be worked through rapidly, and at a higher level. Using this program, the objective became to create rules and criteria for the script to build informal settlements as accurately as possible. So, the first step was to take
Overview

existing buildings from Cidade de Deus and translate them into images legible by the CA. Each block becomes a pixel in a 100 by 100 pixel image.

The challenge becomes to write a set of rules which govern the state of each cell so that when the CA is run, the house model grows to have additional spaced added on incrementally. To control the growth of each wall, multiple rules are needed. Once walls have been produced which form room-like spaces, the program that they serve must be denoted based on a percentage likelihood. Then the final step is to change each wall so that the material delegated reflects as accurately as possible the existing conditions of informal growth. To attempt this, the rulesets are incrementally developed based on specific hypotheses pursued. As methods are discovered and improved upon the hypothesis changes to accommodate these as well as postulate further goals.

The result of this process is a two dimensional image in which a basic house model has grown to encompass generations of informal growth. The final output is an image which denotes the two dimensional plan form, the program for each space created as well as the material which is used to enclose that space. The percentage parameters controlling the process can be changed to fit differing settlement conditions, allowing for the CA to take on a role as a flexible design tool. This process of importing an image to the CA becomes more accurate with both iteration and user discretion. Because the output is based on percentage likelihood and is essentially random, the image produced may or may not reflect accurate building conditions. It may be up to the architect or designer to sift through the outcomes produced and determine if the result reflects the settlement typology being studied.

Examining results of a tool run over many iterations, modeling the aggregation of materials and volumes in an informal settlement as accurately as possible, architects will be able to better understand potential growth patterns in these settlements. Drawing from data and statistics on materials available and building typologies, an existing base model of a settlement can be used as a framework on which to begin modeling potential growth patterns. The program can generate permutations of massing or material usage in a two dimensional way throughout a settlement from which an architect can extrapolate. The output of such a tool would be a set of visual data which helps to postulate what forms might be generated by occupants over time given specific user defined conditions.
The three settlements which were studied by this project were Cidade de Deus in Rio de Janeiro, Brazil, Kowloon Walled City in Hong Kong, China, and Torre David in Caracas, Venezuela. Each of these three settlements have formal spatial constraints or origins which is in part why they were selected. The expansion of each settlement has been documented closely by architects in all three cases, thus providing a strong framework of research from which to draw from.
Cidade de Deus was created in an effort to move people from urban favela’s around Rio de Janeiro into a more suburban/rural setting outside the city. Its intention was to provide adequate housing, but at the same time it moved urban poor out of the city’s more valuable real estate. This suburbanization distanced the occupants from jobs in the city and the new and isolated community was not dense. The suburban planning methods allowed for residents to slowly infill the blocks until it achieved a density similar to that of urban Rio de Janeiro. [2]

The city today has filled in. This diagram shows the full extent of the Cidade de Deus today with the solid yellow line. The suburban housing units which consisted of two bedrooms, a kitchen/dining room and bathroom have been deconstructed and enveloped by the expansion. The city today has turned from a poor and dangerous settlement to a new place for the growing Brazilian middle class. The city was the target of a strong police crackdown on crime in the early 2000s and has since bounced back. A local currency called CDD has been created which promotes local business and is currently valued at 20% higher than the national Brazilian Real. In 55 years of growth a what was essentially a government initiative to displace poorer people has become a thriving community. [3]
Cidade de Deus

high-rise housing blocks
suburban style housing blocks

Rio de Janeiro
The city is composed of two original planning schemes, suburban and vertical mass housing. Over time, both housing typologies were informally expanded on to accommodate the residents needs. The suburban typology grew denser with added structures to buildings and eventually the removal of the single family houses almost entirely, in favor of multiple story residences. Other spaces became stores and other programs. The high rise housing was acted upon in similar ways to create space for stores, garages and other programs. These additions helped to fill in the open spaces between high rises. Open blocks planned into the suburban side of the settlement remained free of structures and are still main gathering spaces. [2] The suburban style houses were much easier to informally build on than the tower housing. The construction and planning of the towers was such that it was often too difficult to add on.

In the book *Cidade de Deus; working with informalized mass housing in Brazil*, the authors diagram the expansion of the both suburb and high rise housing over time. The addition of ad-hock pieces were added over many years and predominantly at the discretion of the individuals constructing them. The informality stems from the poverty of the community, the nature of construction is a product of the resources available. Each addition has different implications for the community, walls serve to protect but separate individual parcels of land, while porches and awnings can provide inviting spaces. The city is a product of formal master planing as well as informal development. The informal construction is essentially occupants of the city investing in their community in a way because it was one of the only ways real estate was added. This city is a case where a fairly depressed government development was converted into much more valuable space as a result of informal construction and community building. [2] [3]

Kowloon Walled City began as a Chinese fort which was constructed between 960 and 1297 BCE. In 1841 the British occupy Hong Kong and by 1898, jurisdiction over the fort and the small “city” it now surrounds is being debated between the Chinese and British. The British finally seize the city, but due to the fact it’s surrounded by slums, no effort is made to maintain the city. The walls surrounding the city are demolished in 1940 by the Japanese occupying Hong Kong, and from then on, the city becomes a haven for people who are escaping rural life for more successful prospects in the city. Due to its location and confusion surrounding government control, it becomes a haven for poorer people and even lower middle class who constructed business operations within the city. It also became home to drug users, criminals and other more darker aspects of society. On a basic level it grew from a need based desire for more housing space, coupled with a complete lack of governmental oversight or regulation. Developers saw an opportunity for profits unsupervised construction, and built high rises from 1973 onward, increasing the density of the city until it was the highest in the world. [4]
Kowloon Walled City

The city started with around 2000 occupants in 1940, grew to 10,000 by 1973 and topped out at around 50,000 by 1990 when it was torn down. The majority of the city was made up of housing for the residents of the city, but it also contained numerous businesses, factories, food processing facilities and even a temple. The city was also home to many more unlawful programs such as gambling facilities, prostitution houses, and drug dens. The illustrated section below shows the density of the city as well as some of the activities going on inside the block. The city also became a place for people in Hong Kong to buy goods at cheaper rates than they could downtown. By the end of its existence, it was providing a valuable market for goods within the city as well as homes for 50,000 people. However the sanitation within the settlement was very poor, over the years the Chinese government had made efforts to bring in clean water and electricity, but largely unsuccessfully. [4]
These are diagrams from the book City of Darkness and they help to illustrate the spatial aspects of Kowloon walled city. The diagrams on the left show how the scale of the buildings changed over the course of the city's development. The city began as smaller single or double story homes. Over time as developers from Hong Kong saw the city as a building opportunity, taller tower housing of up to 14 stories was constructed. The initial small parcels of land were consolidated into larger and taller buildings. [4]
As the city filled in and became a dense block, streets became increasingly compact. They carved through the block and had numerous alcoves for businesses opening up onto the corridors and small public spaces. There is a chaotic, maze-like quality to the circulation, but due to the consolidation of parcels the circulation became somewhat more straight forward. [4]
In the book City of Darkness, Ian Lambot categorizes apartments relationships with circulation into various typologies. These diagrams are self made reproductions of his typological analysis, and serve to show the different formal interpretations for each typology. Understanding the ways in which the high-rise towers shift form around staircases and access points is crucial to understanding how the city was spatially conceived. Buildings sometimes had public staircases which went through private apartments. And frequently buildings would borrow staircases from the neighboring building and simply punch through the wall of the stairwell in places to provide access to their own corridors. This was highly unsafe and challenging to navigate, but residents frequently took staircases directly to the roof and used that as a means of lateral circulation to get to their own apartment. [4]
The tower began as a development project that ran into financing issues when the owner died and the recession hit, and so was abandoned in 1994. The government absorbed the bankrupt company which had begun the project. The property was guarded for years, but many of the valuable materials on the site were taken by thieves and scavengers over the years. In 2007, there was a flood in Caracas, and some of the people in affected areas sought shelter in the tower. The guards let the people shelter in the tower, after which the new occupants decided to see how long they could make it last. After months and then years of being left alone in the building, residents began to make their homes more permanent. [6]

New residents converted the office tower floors into various configurations of apartment and retail spaces. The tower provided a relatively safe and environmentally protected area for people to live. As people moved into the building and constructed residences, others moved in and constructed stores. The tower functioned as a thriving community for a number of years beginning in 2007 when it was first occupied until 2014 when police evicted the residents. As of now, the usage of the building is currently up for debate. [6] [7]
The diagram to the bottom right by Urban-Think Tank shows the location of various residential and non-residential programs within the tower. The majority of stores which opened were grocery and convenience style stores. Other programs which were added were workshops and manufacturing spaces. Due to the unfinished nature of the building, there were no elevators for residents to use, making it increasingly difficult to live higher up. The residents were able to use motorcycles to get up to the first 10 floors due to an adjoining parking garage, but the remaining floors were only accessible by stair. Less able residents like the elderly and handicapped usually took up residences near the bottom, with able people living higher up. [7] The total population living informally in the building was 2,500 at its peak occupation. Its occupation began with the housing crisis in Caracas, where there is a shortage of an estimated 400,000 units. The building was only occupied up to the 28th floor due to issues with getting water high enough up and general accessibility. As can be seen in the section to the far right, there was a water tank on the 15th floor which provided residents with a small, but weekly resupplied reservoir of water. [6] [7]
In studying game environments, the principal focus was to look at how the computational model meets user generation, and methods of mass customization. In each case study, these attributes show up, but each one is being operated on at a different scale. Rust is limiting in some ways, but what is constructed is always unique and its completely player driven. In No Man's Sky, the player has limited choices among a mass customized set of parts or components, and the game engine itself draws the environment by blending a kit of parts to make unique content. In the final case study, Minecraft, the player is generating content, but there is a level of abstraction to what is created by players due to the rendering style, as well as an unlimited amount of resources which impacts what people end up building.
Rust is a game in Alpha release created by Facepunch Studios in the UK. It is a first person, multiplayer, survival game in which players collect and scavenge materials from their environment in order to create various structures and items. Players can attack one another and take each others possessions, making it crucial to build a structure which can protect what you’ve made and collected. When you log out, your body remains “sleeping” on the server, making it vulnerable to other players. Due to this hostile and resource scarce environment, players creations are interesting to study due to their ingenuity and informality. Often players work together to protect their common interests and form factions. These groups create large structures to protect their valuables and share them with each other. Other groups act as traders, pacifists, raiders, pretend religious orders and a variety of other creative roles. While the context is unlike any real-life scenario, the structures produced often resemble aspects of informal settlements. [7]
Once players have collected resources, they can craft a building plan. Once created, building is done through an interface which appears on the screen when a plan is accessed. It provides a series of selectable options of modular pieces for players to build with. The player decides where to begin building, and the ground must be sufficiently flat to place a foundation. There is no global grid system or module in the game world which constructions have to adhere to. So once a player has set down a piece, every subsequent piece is referring back to the module of that original piece. The main goal when constructing is to make sure that resources are protected by whatever is built. To this end, any initially built material can be upgraded four times, eventually reaching the limit of its defensive qualities at reinforced steel paneling. Upgrading requires incrementally more resources with each tier, and thus requires a great deal of effort by the player, making resources even more valuable. [7]
build foundation on stable ground  
construct walls  
windows optional  
build door and second room  
expand as needed
These are examples of player built towns/villages. These creations are the product of random individual players getting together into larger groups and building together. There are no larger governing rules by which these players are operating, the ability to steal or murder is the same. There is an interesting social aspect to this construction, in that the players benefit more from being packed closely together because of the ability to share materials and protect each other. It is the virtual game equivalent to early villages throughout the world. [7] [8]

Interestingly, due to the textures of the materials in the game, there is a similar aesthetic to that of many informal settlements. More interestingly though is the spaces both intentional and residual that are a result of the construction. Many independent operators come together with different goals, but similar access to materials and building methods and they have created a cohesive community. One large structure which housed all of the players involved might be less safe. This is due to the fact that none of the players living there would have anything of their own invested into the creation, it would simply be living inside someone’s created environment. In a sense players have something at stake and so they treat their neighbors as they want to be treated. [8]
Minecraft is an independently developed game that has acquired a lot of attention in the past 5 years. Due to its game-play concept, it has become an influential example within game design for survival and sandbox games. It abstracts the realize so many other games pursue in favor of 8 bit textures and simple character models. The biggest element to the game that sets it apart, is the fact that it breaks down the game world into modular voxel blocks which can be altered at will by the player. The game can be played in two different ways, one way is creative, where supplies are limitless and great creations can be built and the other is survival where supplies are collected, and there are hostile creatures. In both, players have managed to create impressive structures which take hundreds of hours to build. [9]

Due to the fact that the game was initially created by a single person, Notch, the generation of different worlds players build in, is computational. Every world is generated through a seed value, which is a numeric value that guides the output of the generation. Each value produces the same result every time that it is used to generate a world. [9]
The generation the world is done primarily through the use of Perlin noise and Perlin worms. Perlin noise, pictured across the top right, is a gradient noise which is often used in procedural design. It was developed in 1983 by Ken Perlin in order to improve aspects of repetitive computer graphics. Perlin noise appears to be somewhat random, but at the same time, all of the data (gradient values) are at the same unit scale, and thus can be applied evenly. In Minecraft, this method is used as a terrain generator in conjunction with Perlin worms and a variety of other generation methods focused at specific elements. Perlin worms are used to create cave systems in the worlds and can interconnect providing players with dense cave networks. [9] [10]

Minecraft also generates cliffs and overhangs, which do not rely on Perlin noise. Additionally, villages and other constructions meant for the player to explore are placed in the map based on a series of location based parameters. These elements will generate only if these parameters are met, and will vary in organization between cases. [10]


This project is a collaboration between Mojang, the parent company of Minecraft, and UN Habitat. The goal of this project is to get communities around the world to be more involved in local planning and development projects. In this project, Minecraft is being used as an interface between the reality of the site and the communities desires. Mojang builds each site in Minecraft on a 1-1 scale, modeling in similar textures and color palates. Then they work with UN Habitat to help residents realize their goals for the community projects. As a tool for expression, Minecraft’s easy interface and concept allow for residents to shows more specifically what they envision. [11]

In the left column are real world images compared to their Minecraft equivalents. The realism is surprising based on the simplified graphics in the game. The center column shows members of the communities, both young and old, participating in the design project. And finally, the right column shows the redesigned park based on the residents input. [11]
This much anticipated game is currently in development by Hello Games and is set to be released in June 2016. What makes this an interesting model to study is that the entire environment is procedurally generated, meaning that it is created from a series of algorithms. Each planet, spaceship, and animal is created through a series of dependent systems of code, each one building off the previous and depending on variables produced. For example, if a planet is the right distance from a star it might have liquid water, if it has liquid water it might have rivers, and so on, down to the level of aesthetics of the animals and plants on the surface. What is interesting about these systems, is that for the mass customization of planetary, animal and spaceship forms, mathematical algorithms are used. Using the algorithmic method allows for all of the data for each planet to be held within the equation rather than a specific and data intensive 3D model. The game engine can simply build the model of the planet, animals and ships based on the x,y,z location within the universe. Additionally, this allows the universe to be made up of 16 quintillion accessible planets, which makes the game environment infinite to a human player. [12]
The terrain of planets is essentially a series of plotted and densely overlapped equations. These equations are designed by the creators of the game to begin with a single numeric input which will produce a unique but specific outcome. This way, at any given moment you can compile the data for a specific x,y,z coordinate and generate that specific portion of the universe. The static equations allow for the world to be generated exactly the same way each time, for each planet to be unique in the universe. This method is also data efficient, so it loads rapidly and doesn’t require much disc space.

Developer Sean Murray explained that in order to debug and look for graphical errors, he has a series for simple equations that he plugs in for form planets. These equations generate very simple geography and make it easy to look for graphical errors. Pictured (top right) is a planet generated using the equation for a sine wave, \( y = \sin(x) \). This planet has sine wave form mountains extending over the entirety of its surface. So essentially, the way that the geography is generated for any more complex planet is by using more complex and chaotic equations, as well as overlapping them to create more densely articulated geometry. Then additional detail elements (which are also procedurally generated) are added to flush out the believability of the planet. [12]
The method of generation for animals and plants is different than that of planets, but uses as similar methodology. The proportions and variability of form and color are changeable on sliding numeric scales. By changing the numbers for each animal or plant, the aesthetics change. Utilizing similar equation based dependent systems allows the developers to input a single number and output the same animal form every time. This can be done with a near infinite amount of numbers, creating a huge amount of variation within the environment. The output from various equations within the generation system is dependent on the generated conditions of the planet the animal or plant is on, such as atmosphere, temperature, terrain etc. Space ships are generated in a similar way, (but without the dependence on the planet conditions) providing each player with their own unique spacecraft. [12]
The case studies which were investigated in this section were all produced by architects. It gives three examples of ways in which architects attempted to use shape grammars for mass customization, and various other methods to create architectural artifacts or proposals. The Housing Agency System is focused on generating potential forms through mass customization, while Kokkugia uses agent based systems to generate what become amorphous aggregations. Lastly MVRDV’s project Vertical Village, is studies for its attempts at creating a tool which architects can use to model out building form based on spatial and environmental parameters. In looking for a method to move forward with in this project, these three case studies provided examples on how architects have approached computational tool making in the past.

This is a generative design tool developed by the Architecture Research Lab (Alexandra and Michael Bergin) to produce mass customization of single family homes. They argue that according to the AIA when it comes to housing design only 28% have a licensed design professional involved. That leaves 72% to be designed without any major changes to the original drawings by the developer. The mass production of these forms leads to standardized communities without thinking about the implications on the client and above all eliminates the design process. The ARL is building off of Autodesk research into what they call a Housing Agency System. The Housing Agency System creates iterations from a specific input form, using a series of algorithms tied into site, planning and formal strategies, construction systems, detail systems, and building components. These algorithms generate custom designs based off of user input on what specifically they want. They can view a large number of output solutions (right) and pick one which they like. [13]
ARL created a model which superimposes various configurations of the basic housing unit. What is interesting about this is that it simultaneously showing 72 different possibilities for the basic model. It provides an overview of all of the spatial arrangements which have been analyzed for this model. The overall form does not change much in either volume or form, but provides numerous options for a person considering the model.

The spatial effect of this overlap below is similar aesthetically to the forms of informal additions in settlements. The image to the right shows the Mississippi river over a period of 300 years. The superimposition of the path of the river allows the viewer to better understand the nature and impact of the river on the surrounding land. Mapping the changes to an informal settlement over time, using a method which looks graphically similar to ARL’s could provide insight into the nature of settlement growth. Additionally, it could be used in a method similar to ARL’s to provide possible iterations of settlement growth based off of a hypothetical formal origin. [13]
The variations produced from the Housing Agency System start with a single base and operate on one aspect of the form at a time. When looking at Kowloon walled city, which was a high density informal settlement built outside Hong Kong, there are identifiable typologies of housing. These typologies are variable due to the constraints of the site, but have relatable origins. In the book City of Darkness, the author, Ian Lambot, categorized different apartments based on the relation of the stairwell to the building. These stair/apartment typologies differ based on the construction methods, and space available and even shift from floor to floor. When compared to the Housing Agency System variations produced, there is a clear relation between the formal operations. The possible versions of informal operations in Kowloon walled city could be modeled through a generative design program like the Housing Agency System. This could provide models of possible forms on a site or model potential for growth on an existing building. As a tool for development within existing informal settlements, it could a efficient way to look at specific sites and see what possible informal additions might be made in the future. [4] [13]
Kokkugia

The method deployed by Kokkugia in the majority of their projects is to generate form and organization through the use of autonomous agents. There is focus on decentralizing the decision making process allowing for an emergence of form through small actions. Their agents have specific and limited interactions with one another, but have similar goals. There is an insect-like quality to their methods and even the digital images created. Two projects of theirs that are relevant to this thesis are, the Swarm Urbanism, and Behavioral Urbanism projects.

In Swarm Urbanism, there are two autonomous agent systems working together. One agent set is deployed first and Kokkugia describes it as operating similarly to a termite or ant. It creates pathways and self organizes through stigmergic growth. The second set builds off this scaffold created by the first, but builds into more minimal and optimized structures. The overall goal of the program is to develop ideas about how to create ‘self-organizing urban structures.’ [14]
The image on the left is of the Swarm Urbanism project, where it has been superimposed on an urban site in order to attempt to generate infrastructure pathways. The product of this generation seems more chaotic, but produces a compelling set of paths and aggregations. The swarms however seem largely ignorant of the context, and seem to start from a planned point and spill out haphazardly. [14]
The second image is of the Behavioral Urbanism project. In this project the goal was to use ideas about cellular automata in order to produce aggregated mega structures within the city. There is no program associated with these large scale interventions, but housing seems like a likely scenario. There are different iterations of aggregation based on the various parameters which determine the agent's response. These mega forms also seem to have little to do with their context apart from avoiding the water and site edges. [14]
This project grew out of a semester long design studio led by MVRDV about the urbanization of different cities in Asia. What the studio noticed, was that there was a severe lack of affordable housing, with many people waiting years to buy a house. They analyzed the development of smaller scale pockets within the city and analyzed the qualities they maintained. The main argument being that the village mentality of interaction, smaller scale, local interest, community among other things, created an environment that would be positive to emulate or retain when building denser communities in emerging cities. [15] The models produced are their first analogue attempts to produce a vertical community with village like elements. Teams traded models and built in different colors with different goals, all relating back to the qualities of the urban village. Later, they produced models like the one on the bottom left, which consider the possibility of each individual’s home having a unique form. The idea stems from mass customizing of housing, and to understand it they built a Grasshopper script called ‘House Maker’ which allowed users to customize their own house models. [15]
Building off the ideas that were started in ‘House Maker’ the studio created another script called ‘Village Maker’ for aggregating the customized houses. This program took into account specific user defined conditions, such form, windows, space between houses and amount of sun. Using these parameters, the ‘Village Maker’ would generate an aggregated form based on the houses input. It considered a variety of conditions and modeled them in various colors to visualize them. Vertical circulation was a critical issue, it can be seen highlighted in red in the images. The sun cone is in yellow, and the daylight cones are in blue. The houses themselves are in white and are wrapped in purple structure. Each of the parameters could be adjusted and could be modeled with different intents. Structure for instance could be modeled to be like a scaffold, a tree, and a box truss system. The ability to customize the model to the site as well as iterate through design schemes quickly makes this an interesting case study when considering informal settlements. [15]
THE SUN VILLAGE

The sun village is based on the single daylight village, with an additional 20 minutes of sun in the morning or afternoon as an added quality. This results in this but wide sunlight cones that slice through large parts of the village. These sunlight cones could not be built, and so a lot of open space is created. The result is a mountain-like organization of houses, all clustered toward the northern boundaries, allowing sun to penetrate deep into the volume.
This is an array of outputs from the ‘Village Maker’ and their different compositional elements extracted. In comparing these schemes it becomes apparent that certain input models of aggregation cause more complex systems of structure, access, view or housing organization. Unlike the Kokkugia project this is a tool which is taking a set of parameters (house designs) and iterating through them to produce an optimized result form. Within this system however, the structure and circulation elements act somewhat like autonomous agents in order to create pathways to iterate and check. [15]
This project is designed to provide architects or consumers with a rapid way to iterate through project possibilities. As a concept for a tool it is very successful despite the fact these typologies of construction have inherent flaws. The spatial analysis capabilities within this program allow it to provide relatively specific architectural insight into specific form created by architects. [15]

Moving forward from the case studies, it was necessary to investigate informal settlement conditions and create a kit of parts with which to use later. Images of settlements in South America were selected and then modeled in 3D using Rhino. The parts created for these models were catalogued and then would be used later in visualizing the generative models produced. This also helped inform on typical construction methods and building practices. The materials used, as well as forms shown here became studies in how to represent the models that would eventually be generated by the simulation tool being created.
Informal Construction Methods

Case study of construction methods for a small timber frame building with corrugated steel roof and fabric/plastic tarp as walls.

Case study of residential and commercial building made of concrete frame with terra-cotta block infill.
Case study of partially finished building made from concrete frame construction with brick infill.

Case study of residential building made from cast in place concrete with steel sheet roofing covering the roof.
The next step in the process was to investigate various computational methods of approach to the question of informal growth. The five case studied generative programming methods were shape grammars, genetic algorithms, agent based systems, self organizing maps and finally cellular automata.
Shape Grammar

A type of generative system which create 2D and 3D geometric shapes by utilizing a series of specific “shape rules” in combination with a “generation engine.” The engine processes the rules which are transformative operations such as boolean unions, intersection, difference, as well as transformations such as reflection, rotation, scale, and translation this. The initial input is modified through these transformations in order to produce a new form. [16]
Genetic Algorithm

This is a method of programming which mimics the recombinant nature of genetic mutation in order to produce new combinations and variety from specific input models. Newly produced combinations are then assessed based on a best fit model (manually or parametric) and then recombined if they meet the requirements, similarly to the way in which mimics the process of natural selection. They can be assigned a "fitness value" based on their resemblance of the best fit model and ranked accordingly. The final output will be a set of "offspring" which are algorithmically generated, and sorted according to their correlation with the best fit model. [17] [18]

Genotypes that match best fit parameters

parents
generation 1

YES YES NO

"best fit" parameters

YES

NO
Agent Based

Modeling method based on the actions of individual objects or “agents” within the system which act independently based on a set of criteria. Usually when making agent based models, the criteria which determine behavior are simple in order to avoid conflict and chaotic results. The agents start from an origin and are allowed to make a specific number decisions, after which the model produced is assessed based on movement from their origins toward an emergent form. Typical modeling rules include cohesion and separation where agents steer towards or away from each other or another specific element, and alignment where agents derive their trajectory from the average of their neighbors. [14]
Self Organizing Maps

As a type of unsupervised learning, SOMs are classified as artificial neural networks in programming, they are typically used to take multi-dimensional data (x, y, z... R, G, B...) and represent it in lower dimensions. Usually colors are used for data because of the fact that they carry 3 values if assessed on an RGB scale. And the location is addressed by the x, y coordinate. The individual neuron’s within the system randomly select a sample, then move to the best matching coordinate, scale the neighboring neuron locations and increase the weight of the neuron (resistance to movement/scaling). This is repeated often more than 1000 times. [19]
Cellular automata (or CA) are a method of modeling generative growth through 1D, 2D or 3D environments. The growth of a CA is determined by a ruleset, which looks at the context and causes an action based on what is observed. CAs were first developed in the 1950’s by Stanislaw Ulam and John von Neumann as a method of understanding growth of crystals and possible systems of self replication respectively. Since then, models have expanded from bi-state to multi-state and from one dimensional to two and three dimensional. [20]

In order to implement the rules on the cells that are laid out, the state of each cell in the neighborhood has to be assessed. For the purposes of this project, a 2D CA was used due to its simulation capability as well as ease of writing rules for the CA. [20]

Code example for 2D CA:

```python
check(grid, c - 1, r - 1)
check(grid, c, r - 1)
check(grid, c + 1, r - 1)

check(grid, c - 1, r)
skip (c, r)
check(grid, c + 1, r)

check(grid, c -1, r + 1)
check(grid, c, r + 1)
check(grid, c + 1, r +1)
```
The 2D CA works by assessing the neighborhood and state of each cell in a grid. This CA was written to move left to right across the grid, row by row. Each time it checks the state of the cell and compares it to the dictated rules. Each time the CA reaches the end of the grid, it increments the generation count by 1. Also, as the cells are being changed, a cell that is changed to the left of one cell does not change how that cell is assessed. The state of each cell is being assessed based on a static image of all of the cells at the end of the previous generation. In effect, all cells are being assessed before any actions are being taken, only upon reaching the end of the generation are the cells changing. What this does, is prevent the CA from becoming trapped in loops or creating hierarchy for construction based on the order in which it is assessing cells. This provides a much better ability to control the CA’s modifications to the base image as the ruleset creator. [20]
The state of each cell will be assessed when assessed, it will be compared to a set of rules about how cells behave.
Finally, after choosing a CA to begin development, the focus for implementation shifts back to Cidade de Deus in a way. Currently the Brazilian government has a program called Minha Casa Minha Vida, translating to My House My Life. This program uses essentially the same model for housing that was used nearly 60 years ago in Cidade de Deus. The recurring implementation of such a simplified model for government housing is a result of both a lack of architect involvement as well as corruption within the construction industry. For the government, the houses are being provided cheaply, but with a change in masterplanning this type of development could be rethought in order to “maximize return on investment.” By saying maximizing return, what is implied is that with the understanding from the beginning that the houses will be informally expanded, the planning should change to reflect that. And, it that happens it can be expected that the growth which happens can be influenced as well as encouraged as a way of having residents add value to the development over time. In the long run, Cidade de Deus is envisioned as an example of successful development in that the city which began as a slum, expanded to hold three times the density of people, and the value of the properties became equivalent to a lower middle class neighborhood. [21]
The translation of the Cidade de Deus housing model into a CA was straightforward. The CA requires an input image to begin operating on, and so the house plan was abstracted into a png image and distances became pixel dimensions. It was translated at two different scales, the first was where 1 pixel in the image was equal to .2 meters, and the second translated to 1 pixel to .1 meters. This allowed for testing at two different scales as rules were developed. In some cases, the scale of the input image changed how the CA would operate, and the goal was to make rulesets which could be applied to a number of inputs. In these images, the walls became blocks filled with black, interior floor space became red, and windows became yellow. This provided an easy way to distinguish as well as denote locations within the input image. A third image (on the bottom left) became a very basic testing condition.
The first import image is a basic and scaleless model of a small 1 room building. It contains all of the components of the more detailed versions however.

The second import image is a scaled model of the Cidade de Deus government housing at 1 square equals .25 meters square.

The third import image is a scaled model of the Cidade de Deus government housing at 1 square equals .125 meters square.
To begin working with CA’s the first step was to create a ruleset for a Bi-State CA. This type of CA only had two cell types, dead/alive or white/black. This is the first ruleset created and essentially it identifies a wall, or the joining of two lines perpendicularly. From there it begins modifying the connection in order to produce a start of a new wall. This budding is based on creating patterns in the neighborhood that can be dictated in the CA ruleset. The rules themselves are a dependent system, meaning that one builds off the next and without any given rule, the CA would not progress past that point. The challenge with this is that it becomes difficult to change rules created early on, without breaking the intended effect of the CA. In this case, of a Bi-State CA, the rulset is very simple due to the limitations of control with only two states. So to advance from here, it became necessary to add additional cell states (different colors) as well as probability components to rules.
One of the early rule sets that was written for the model which incorporated probability and multiple states, was this example. Here, the generation of walls was based on the likelihood that a black vs red block would be placed. Initially a wall would start from the edge of an existing opening (window or door) and grow upward, with a 90% each time of placing a black block. There was also a 10% that a red block would be placed. In which case, the next generation the CA processed, would activate a rule which indicated a black block to be placed to the right of the red block. From there, subsequent generations would add black blocks to the right with again the same probability of 90% and 10% for red. Now, whenever red blocks were placed in this horizontal direction, the ruleset would indicate the beginning of growth down. Essentially, this ruleset was created a spiraling effect of walls growing out from the start and redirecting back towards the ‘house’ in an effort to create room-like conditions. The ruleset succeeded in creating partitions, however there was a lack of control regarding the number of rooms, and their frequency. And the final results were somewhat chaotic and arbitrary. However, it was a large step forward in thinking about the usage of probability and multiple cell states.

Rule 1: 111 111 003 -- 0 (100%)  
instigates growth from window edge

Rule 2: 111 111 101 -- 0 (90%) ; 2 (10%)  
stops wall growth at certain height by placing red block

Rule 3: 111 211 011 -- 0 (100%)  
adds black block to right of red to begin growth towards right

Rule 4: 111 011 111 -- 0 (90%) ; 2 (10%)  
instigates growth down based on probability

Rule 5: 021 111 111 -- 0 (100%)  
adds black block below red blocks on horizontal

Rule 6: 101 111 111 -- 0 (95%) ; 2 (5%)  
continues growth down from black blocks

Rule 7: 110 112 111 -- 0 (100%)  
place block to left of red blocks on downward string

Rule 8: 111 110 111 -- 0 (90%) ; 2 (10%)  
instigates end of horizontal with probability of placing red block
This CA begins to utilize more cell states in order to better control the CA's output. Each color can now be coordinated with a specific change in direction. So for instance in this case, vertical growth is still stopped by placing a red block. But when the CA begins building left, the placement of a yellow block dictates that the CA stops building left. But in each of these cases, both red and yellow, there was a 10% and 5% change respectively of each being placed. The effect is to begin creating different sized spaces which build off the input image. This was also the first attempt to write rules which would create a fill for a space created by the CA. In order to create a fill, 28 different rules needed to be written in specific order. But with this came the concept of probability dictating different color fills for walled spaces created with the CA rules. At this point, the CA also was limited to building only upward, to the left and down. This was very limited and growth was only occurring on the top edge of the input image. And so at this point the images produced were not reflecting accurate comparisons to real world conditions.

Rules to create wall with probability

- 11111003 -- 0 (100%)
- 11111101 -- 0 (90%), 2 (10%)
- 111111001 -- 0 (100%)
- 11112110 -- 0 (100%)
- 11110111 -- 0 (95%), 3 (5%)
- 13011111 -- 0 (100%)
- 10111111 -- 0 (100%)

Rules to create even blue fill

- 002110110 -- 4 (100%)
- 000114111 -- 4 (100%)
- 444111111 -- 4 (100%)
- 444414111 -- 4 (100%)
- 441411111 -- 4 (100%)
- 144114111 -- 4 (100%)
- 444414414 -- 4 (100%)
- 444114114 -- 4 (100%)
- 444141144 -- 4 (100%)
- 444114114 -- 4 (100%)
- 444114111 -- 4 (100%)
- 440410410 -- 4 (100%)
- 440410110 -- 4 (100%)
- 440110000 -- 4 (100%)
- 444114000 -- 4 (100%)
- 444114100 -- 4 (100%)
- 300014011 -- 4 (100%)
- 300014014 -- 4 (100%)
- 044014014 -- 4 (100%)
- 044014114 -- 4 (100%)
- 101111000 -- 4 (100%)
- 440110110 -- 4 (100%)
- 444114300 -- 4 (100%)
- 444114330 -- 4 (100%)
- 444114333 -- 4 (100%)
- 444114033 -- 4 (100%)
- 444114003 -- 4 (100%)
- 044014011 -- 4 (100%)
- 044014000 -- 4 (100%)
On of the later iterations became like this, where now once walled spaces are being made, they can be filled with three different possible cell states (colors). The walls are also able to build off more areas of the base image as well. Now corners and window/door openings are all points where a wall can begin. This is to reflect the likely points of growth according to observed architectural phenomena. The different colors that are being used to fill space can reflect different programs or materials for those spaces. Depending on how the CA is desired to be used, material could give a useful model of material usage throughout the settlement. The issue again, is that the CA is only building from the top of the input image. Which was predominantly an effort to make it easier to write the ruleset. By only building off the top, the ruleset for downward construction and construction towards the right were excluded. However this doesn't reflect observed architectural conditions.

Note: The rules to the right are written as they are interpreted in the Processing script. The first 9 digits are the neighborhood conditions, followed by a comma, the new cell state being dictated, and the probability (1 = 100% change rate, .9 = 90% chance, etc.)
This shows 66 generations of the CA running for a .2 meter sq scaled house. Each generation builds off the next, and the growth happens block by block.
10-State Probability CA

These are examples of the CA operating on a scaled version of the house where one pixel is equivalent to .2 sq meters. These examples show the variation possible with this method of generative design. Each iteration is different, but based on the same set of rules and the same set of data. There is clearly a level of abstraction with what is being produced. The images are meaningful individually as a means of checking the accuracy of the ruleset but as a whole, they start to tell more about the methods of aggregation. There is an emergent quality to the value of these, where the sum of each part is insignificant compared to impact of the whole.
These CA examples are iterations where the scale of the input image was one pixel is equal to .1 meters. Compared to the previous page, the images produced here show much smaller informal additions. This is caused by scaling the input image and leaving the rules unchanged. The goal was to test the effect of the CA at different scales and see if it could still be useful at scales other than .2 sqm per pixel. The result is that essentially it can be used and whatever scale relates best to the expected average size of informal additions. Meaning that as a user, data on average aggregation size must be either found by or collected before using the CA. Or conversely, the CA can be used on many scales of the same building therefore creating different permutations based on average informal addition sizes.
This shows the CA growing as a 3D model. A CA output was taken and then manually 3D modeled and rendered at different generations.
Once the CA returned useful images, it became possible to model the buildings and their additions using Rhino. The plan for each was based exactly on the CA output, and the scale depended on the scale of the input image. Materials were chosen based on the percentage likely hood of occurrence. This data was based on available data gathered on Cidade de Deus, but where information was unavailable, expected values were assigned. For example, data suggested that 39% of new construction was being done with CMU, however limited information was found regarding wood, but due to climate as well as observations of images of Cidade de Deus it was assumed that around 12% of new construction might be wood. These numbers may not be completely accurate to what is existing, but they provide placeholders which help to check the accuracy of the CA.

<table>
<thead>
<tr>
<th>Wall Materials</th>
<th>Likely hood</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMU</td>
<td>39%</td>
</tr>
<tr>
<td>Concrete</td>
<td>31%</td>
</tr>
<tr>
<td>Fabric/Plastic</td>
<td>18%</td>
</tr>
<tr>
<td>Wood</td>
<td>12%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Roof Materials</th>
<th>Likely hood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrugated Metal</td>
<td>61%</td>
</tr>
<tr>
<td>Concrete</td>
<td>24%</td>
</tr>
<tr>
<td>Fabric/Plastic</td>
<td>15%</td>
</tr>
</tbody>
</table>
As more CA outputs were modeled, inconsistencies with expected construction methods were observed. Examples of such occurrences are walls which double back on themselves, walls which seal off windows but provide minimal additional spaces, or walls which extend across a facade effectively blocking it in. These suggest that the accuracy of the CA can be improved and these specific instances are things that might require writing new rulesets to fix.
As a final step in visualizing the output produced by the CA, 30 separate iterations were overlaid and made semi-transparent in order to see which areas were being constructed on the most. The result is these 2D and 3D images which display areas of new construction in blue/greens and walls in black. The darker the color, or higher the elevation in 3D, the more construction has taken place in that location. What these images show, is that due to the ruleset, little to no building is going on in the bottom left corner of the houses. This suggests that another area of development is to write rules to ensure that all sides are being developed equally, or at least in accordance with what would be expected in a given settlement. Otherwise, these images are starting to give a sense of where the most construction is taking place and therefore which areas architects should look at designing.
CA at the Neighborhood Scale

After using the CA to produce images of single house aggregations, the next step was to test this concept on a neighborhood. The input image constructed for this is modeled after the plan of a single block in Cidade de Deus. 18 houses are arranged in two rows with equal spacing in between. The images on the left side show the CA as it builds out to its final static state. And the images to the right side show the starting condition in 3D as well as the final output modeled in 3D. The settlement that was generated is only as accurate as the single house modes were determined to be. Producing this is in some ways more of a proof of concept as it is not completely accurate, but it shows the potential for accuracy with a fully calibrated and fine tuned CA ruleset.
These are 2D and 3D images produced from overlaying 30 CA outputs of the neighborhood scheme. The goal here is to show that while acknowledging the flaws with the current state of the CA (particularly that with the density of growth from the bottom left side of the houses), the CA is beginning to produce an accurate model of how a neighborhood might expand informally over time. Similarly to the single house models produced by the same method, this model shows density of growth through opacity in 2D and height in 3D. The 3D image is thought of as being similar to the types of outputs that would be generated for architects to read in a completed version of this software. The architect could look at the map and identify where building was likely to take place and then respond accordingly.
The image on the top was created from 30 output images overlaid together in black and white. It is showing potential growth examples layered on top of each other. While not representing a concrete result in terms of settlement conditions or density, the density of the image is beginning to suggest that the program, with the right modifications could produce an output that is very similar to the existing conditions of the site (shown below). It also looks at the idea of layering the outputs to produce a different type of outcome where either multiple floors are being modeled or different densities of walls.
New Neighborhood Schemes

In this neighborhood scheme the plan of the block was changed to alternate the distance between houses and street edge (perimeter of image). The result was a different density than the exiting conditions, but the differences are hard to identify. The overall distance between houses remained the same, so the only change was that the informal aggregations were better able to fill in the spaces between the buildings. But what is interesting about this test is that even with minimal changes, the growth of the settlement is being impacted by the design of the neighborhood masterplan. This supports the hypothesis that a tool for architects to use for modeling informal growth when designing these settlements is relevant and useful.
New Neighborhood Schemes

For this iteration of the neighborhood scheme, the houses were clustered into pairs. The expectation was that this would provide areas for larger aggregations along between pairs while still allowing for small aggregations to happen between each house in the pair and in the front and back of houses. The results indicated that this expectation was accurate in that larger spaces were able to be generated between pairs, but due to the rules of the CA, it became much less likely to build between units. This again illustrated that by changing the masterplan scheme, the expected results of the informal aggregations generated reflected that. The success is that the program is responding to the input in different ways and the ways in which it responds are informing about assumptions that might be incorrectly made.
New Neighborhood Schemes

In this iteration of the neighborhood schemes, the clustering method was used but in addition, the buildings were placed along a spine. The expectation was that the buildings would again fill the spaces between with larger volume informal aggregations, and there would be limited construction between the buildings. The concept was that by leaving a spine of less dense construction along the centerline of the block, 20-30 years after construction of the neighborhood, the government could come back and add in or replace infrastructure elements. Creating a flexible environment both in terms of informal additions as well as potential for future civic improvements is important in this type of occupant driven design. If residents are expected and encouraged to expand their houses, it becomes necessary to provide needed utilities to support that increased density. But in all likelihood the money for providing those additional utilities won’t be provided until the need arises.
In this final iteration, the neighborhood scheme was changed to create an open interior to the block, with two openings placed along the edge. The expectation was for the informal aggregation to expand into the interior but leave openings where the houses were omitted. The actual result indicated that while construction was happening towards the middle of the block, the openings where buildings were omitted were not always left open. Sometimes large volumes were produced and other times they were simply walled off. In all three outcomes, open, walled or filled, a different program could be envisioned in that space. In cases where it is walled off it becomes a private space for the block, or a public space if left open. And when it is filled with a large volume, that volume could be a store or other type of programmed space which otherwise wouldn’t have been possible without demolishing a home in the original government masterplans.
New Neighborhood Schemes

Again, in these overlay images the space being left open is very visible, and along the bottom edge of the 2D image, examples of a large filled volume can be seen. Also, the 3D imaging technique by height mapping the overlay is very legible in this final neighborhood scheme. It is easy to see an architect plugging their own scheme into the CA and having it generate potential growth patterns. On top of that, once generated, methods of interpretation like overlay and heightmapping become visual and practical ways to make the CA produced data easy to use and more descriptive.


