Parametric Analysis for Building Efficiency

developing a tool for diagramming programmatic relationships using adjacency requirements

Christopher Boon. graduate student
Lyee Chong. graduate student
Sergio Palleroni, Huafen Hu. professors, school of architecture
Benjamin Deines. advisor
Nicholas Papaefthimiou, Jonah Ross. ZGF collaboration

ZGF
ZIMMER GUNSUL FRASCA ARCHITECTS LLP

_abstract + project goals

Emerging Parametric technologies are opening new opportunities in Arcitecture. Generally it is seen primarily as an engine to drive formal exploration and renderings. Its implications however are larger and it is possible to employ it at many stages in the design process. During the initial design stage, much of what is explored involves theoretical concept. The work is expressed diagrammatically. If the concept can be distilled to its parameters, then it is possible to begin including parametric analysis. This type of analysis will allow designers to develop a much wider range of options in a much shorter timeframe. This study intends to explore the ability of parametric modeling to be useful as a diagram-building tool.

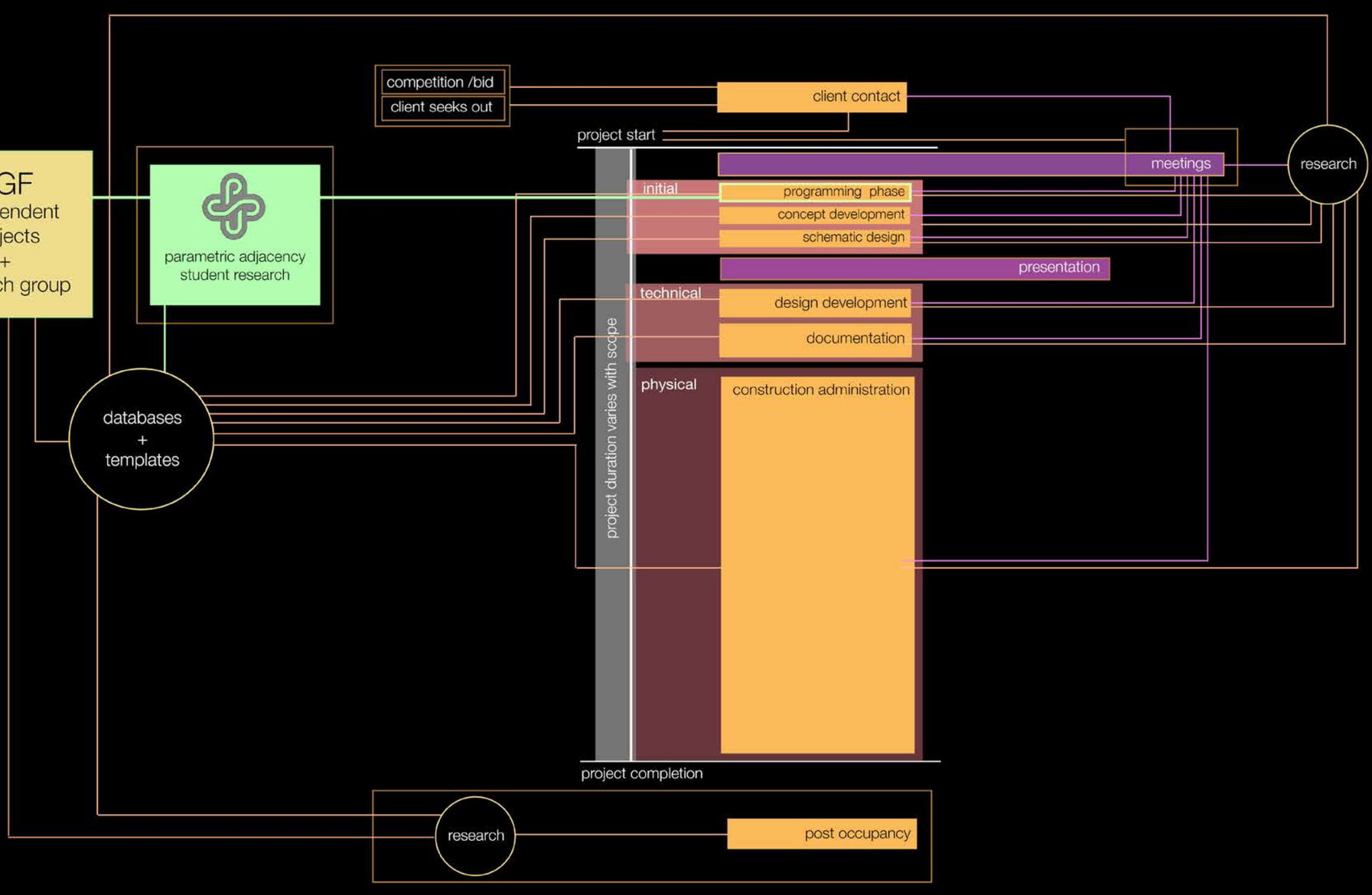
This project explores the utilization of Grasshopper (a Rhino Plugin) as an analytical and planning tool to graphically represent a 3-dimensional analysis of adjacency requirements in program and spaces. The purpose of this project is to explore and innovate new ways of further utilizing the technology in evolutionary algorithms or physics modelers to reduce inefficiencies in the design process as well as optimize floor-plan layouts.

The project goals are simply to reduce inefficiencies in the design process as well as optimize floor plan layouts. The efficiency of having a easily manipulatable algorithm allows for rapid prototyping, potentially saving the client and the firm time and money. The architect, the project manager and the grasshopper advisor form the main responsible project team in directing the project's depth and complexity.

_additional resourses

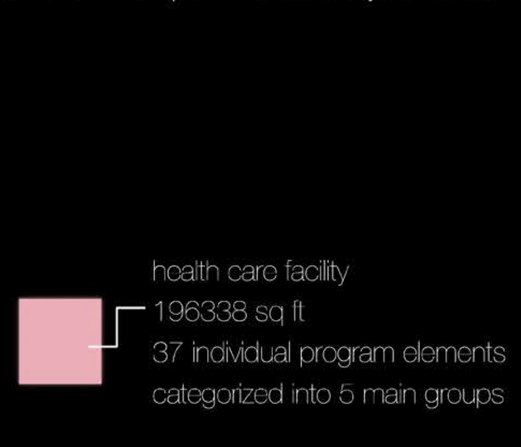
Issa, Rajaa. "Essential Mathematics For Computational Design." Grasshopper: Generative Modeling for Rhino. Robert McNeel & Associates, 29 May 2010. Web.
Khabazi, Zubin. Generative Algorithms (using Grasshopper). N.p.: Robert McNeel & Associates, Grasshopper Generative Modeling for Rhino, 15 Aug. 2012. Web.
Samuel, Flora. Le Corbusier and the Architectural Promenade. Basel: Birkhäuser, 2010. Print.
Syp, Marc. "Architecture: Realtime Physics for Space Planning." Vimeo. N.p., 3 Mar. 2010. Web.
Tedeschi, Arturo. "Parametric Architecture with Grasshopper." Edizioni Le Penseur. N.p., Apr. 2011. Web.

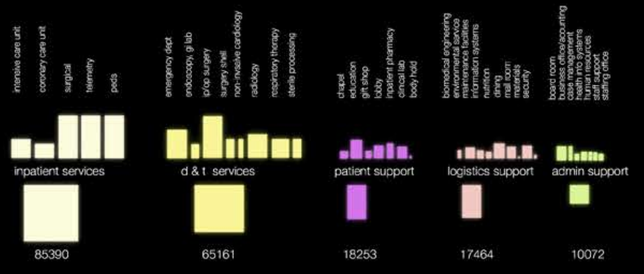
_firm methodology

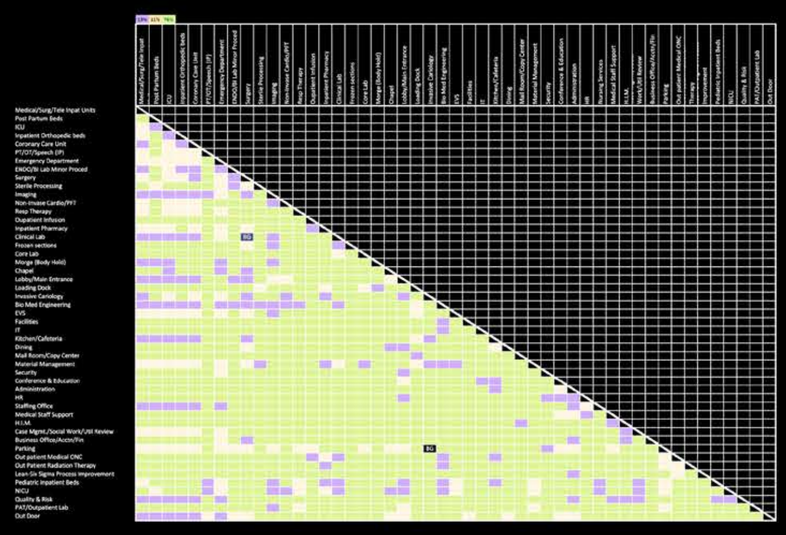


_programming complex relationships

For a building to be efficient it must meet the needs of it occupants. The needs of the occupants are determined by a complex relationship of programmatic and physical situations. These situations can be measured and used as metric data; which can become a parameter. This project will attempt to use generative tools to create spatial diagrams using adjacency requirements as the driving parameter to create a method for massing out buildings. Programming a health care facility is an especially difficult job. This is an ideal building program to test the grasshopper definition on because of the high level of complexity in spatial relationships and high number of programmatic elements. A hospital also requires a high level of efficiency due to the nature of its function. To get an idea of the variety and type of program we began by establishing categories of spaces, then assigning an order of importance to the space types. A space like the emergency room would take priority over the giftshop for example.
ZGF is a firm that has much experience with designing hospitals, they are experts and they use their own research and intuition to make the spaces work efficiently. When we at PSU were beginning this project, ZGF was in process of designing a hospital and we could observe some charts generated at their client meetings. They established the various needs of the client, then developed a matrix of all the discrete elements. Thhis matrix tells them which spaces want to be adjacent to each other in three degrees.



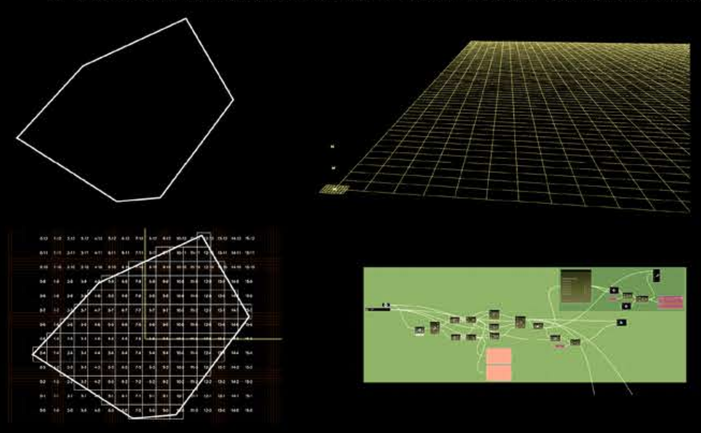




_logic : :

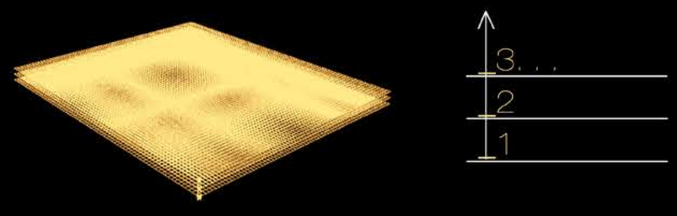
_establish base grid / make site specific

The starting parameters are customizable. the user sets up a grid that the spaces are built within. The number of cells and size of cells can be set to a specific coarsenes. The program can also build the grid inside of a two dimensional site outline to further refine the end results.



_offset multiple levels

To account for multiple storeys the grid is offset vertically. There can be any number of levels, the offst is equal to the grid size so that all directions are equidistant from point to point.

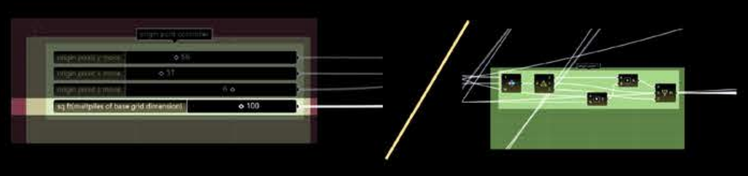


_assign hierarchy of spaces with list order

When the program activates, the order of the list of program elements is the order of importance. The first element dominates the second, which dominates the third and so on. This is applied when the spaces are divided and reassigned, the dominate one always takes priority.

_assign square footage

The volume of each space is set on number sliders.

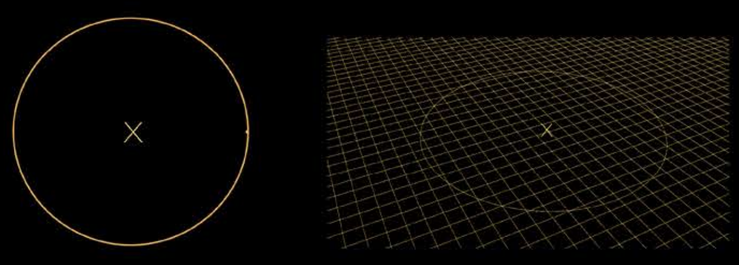


_set origin locations for individual program elements

The various program elements are located with points. Grid cell spaces are filled in using these points as centers. The points snap to cell corners on the pre-established grid. Their position can be moved on the grids in x, y , and z directions.

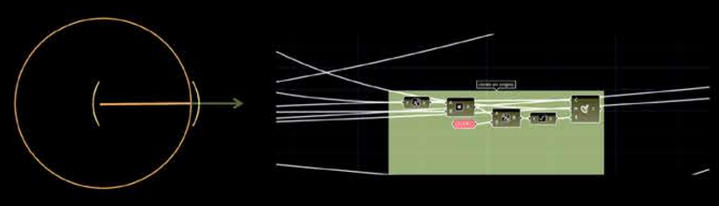
_build circular areas

The volumes are represented at first by circles, from their origin points at the centers. Curcles are chocen because they have equal distance in all directions. The areas are each confined to one level.



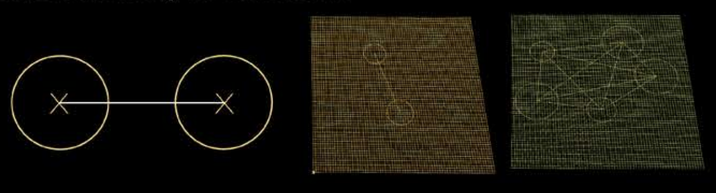
_extractract radius

Evaluate the radii as curves and use the length to determine each circle's domain on the grid. This will be the limiting parameter when they intersect.



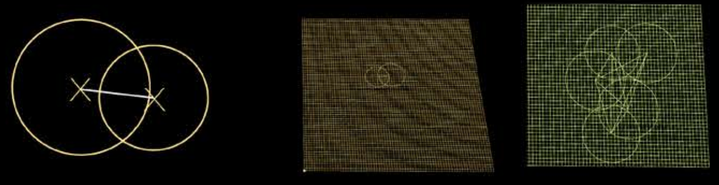
_create vectors between origins

A line connecting origin points is established for every possible combination. These lines represent the distance between programs. The lowest value being the fittest result.



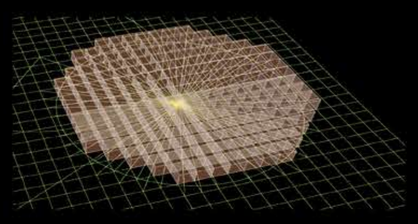
_circles converge on each other limited by radius

The elements wil not cross each others' center points. This prevents them from all collapsing in on themselves. By letting them overlap like this, potential dead spaces are filled in.



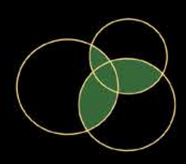
_test for edge points of areas

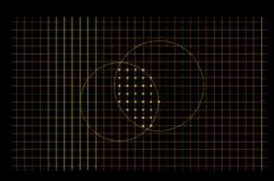
The elements are sensitive to the closest points along their perimeters. These points are referenced when reassigning volume. They represent adjacent, available space.



_test for area intersections

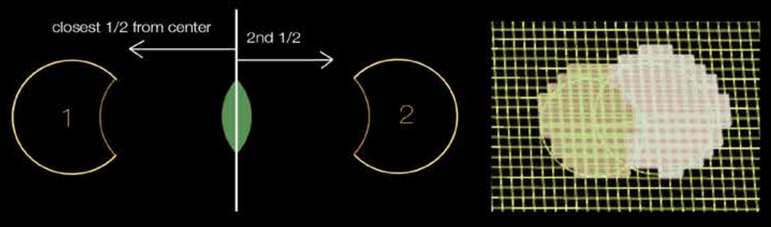
Isolate any intersecting areas. Test for all points inside these zones and remove them from each respective element.





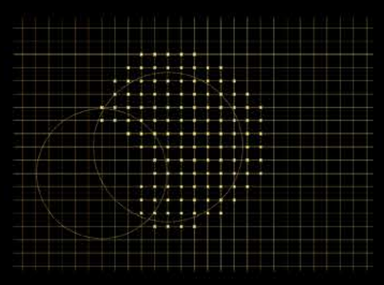
_divide intersections between programs

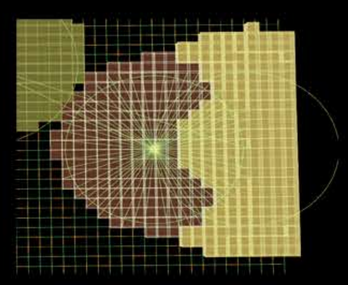
Isolate any intersecting areas. Test for all points inside these zones and remove them from each respective element.

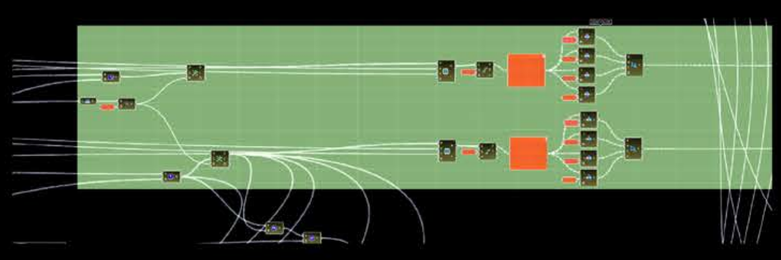


_reassign lost area to adjacent cells

For each intersection, half of the cells are assigned to each circle, the remaining half that was lost is replaced in the adjacent cells.

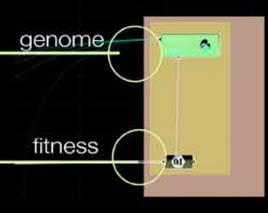






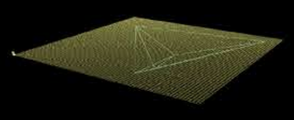
_use galapagos evolutionary problem solving : make iterative generations

The genome collection is made up of all the origin location sliders.



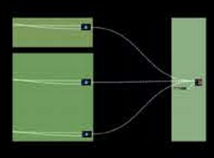
_add up total distance of all interconnecting vectors: fittest results = shortest overall distance

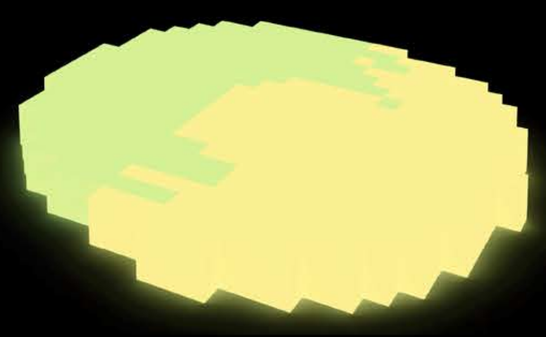
The lines that connect origins are added together and this number is the iteration's score. The use of galapagos allows for a variety of potential results, the lowest score may not necessarily be the most useful result.



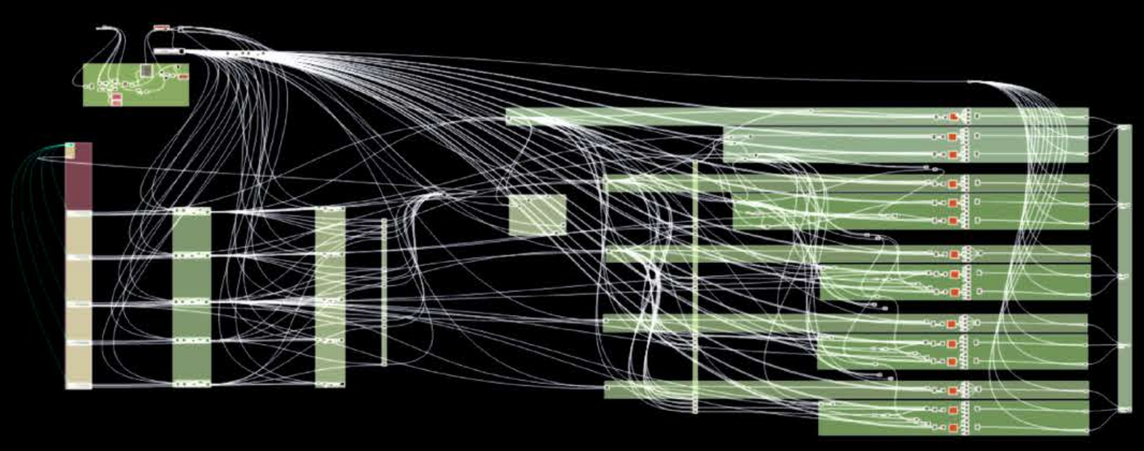
_visualize 3d spaces

The areas inside the circles are translated into squares on the grid and extruded into cubes that give 3D presence to the diagram.





_definition



_conclusions

The program holds potential to be applied at the early design phase. There are clearly many ways to achieve similar results within the stucture of the parametric tools. Better definitions are ones that can accomplish complex moves within a few commands, this makes the load on the computer exponentially lighter and therefore produce results more quickly. Evolutionary algorithms are a way to let the computer do creative work. By employing a combination of parametric operations with evolutionary solving, a vast array of results can be achieved in a short time. The trick of course is building up a working logic flow and refining it. It always helps to have an idea of what you want the results to be, this way you can narrow the variables and explore more pertinent results. The end results of this definition are diagrammatic, so they serve as a source of information that can be observed as a metric then incorporated into design. It could be expanded into a more literal space, with doors and windows etc., and the results could become more directly applicable to building design in the later stages.