Proceedings of the Ninth International Space Syntax Symposium
Edited by Y O Kim, H T Park and K W Seo, Seoul: Sejong University, 2013

A SYNTACTIC ARCHITECTURAL DESIGN METHODOLOGY:

Integrating real-time space syntax analysis in a configurative architectural design process

048

Pirouz Nourian
Delft University of Technology
e-mail : p.nourian@tudelft.nl

Samaneh Rezvani
Delft University of Technology
e-mail : samaneh.rezvani@mail.polimi.it

Sevil Sariyildiz
Delft University of Technology
e-mail : i.s.sariyildiz@tudelft.nl

Abstract

This paper presents a computational toolkit developed for configurative architectural design, i.e. a computational design process, equipped with real-time space syntax analyses in a parametric CAD environment, which begins with defining the desired spatial configuration in form of a bubble diagram. The syntactic design methodology put forward by this toolkit is aimed at bridging the gap between space syntax as an analytic theory of architecture and architectural design practice. The toolkit has been made in an attempt to investigate the possibility of deriving at plan layout patterns through sketching spatial configuration using an 'interactive bubble diagram' that represents a spatial connectivity graph. In other words, we have worked on a way of reaching at concrete plan layouts from an abstract connectivity pattern as a graph. Beginning the design process with a graph allows for real-time feedback of Space Syntax measures such as integration, choice and difference factor. Besides, by choosing every space as a 'root', designers can immediately view their configurative ideas, literally from different points of views in automatically drawn justified graphs. In this paper, we give an overview of a syntactic design process as a graph theoretical approach to architectural design and report our preliminary results.

Keywords: *Configurative Design, Real-Time Space Syntax Analysis, Architectural Design, Computational Design, Graph Theory, Plan Layout*

Theme: *Modelling and Methodological Developments*

Background

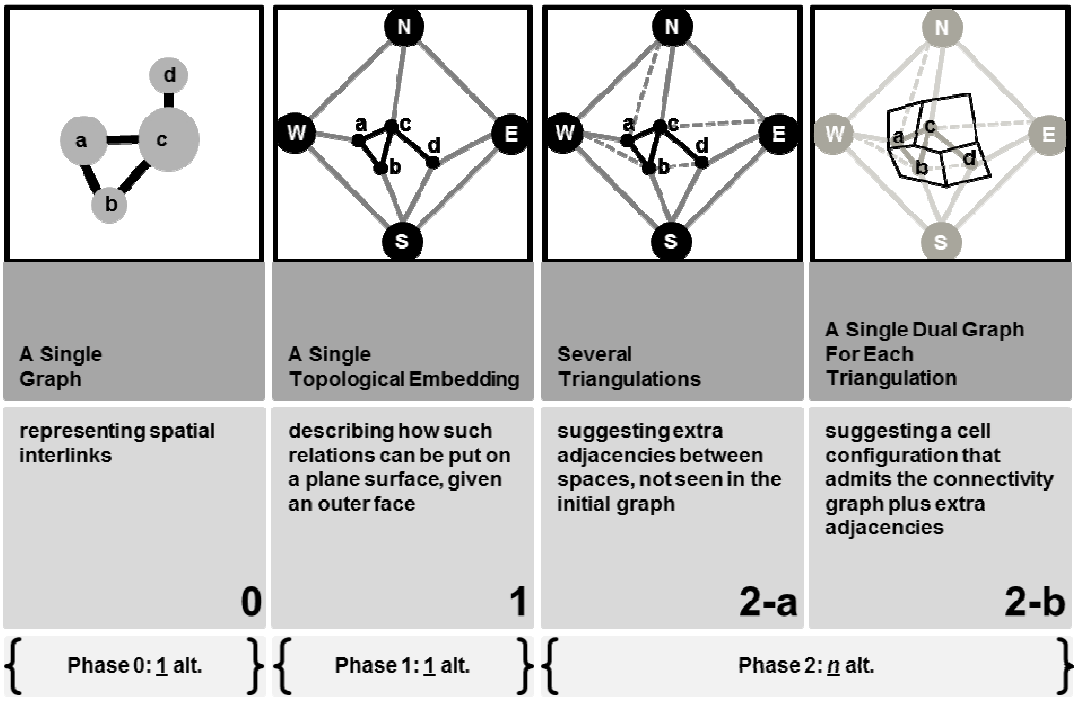
If we consider spatial arrangement as a fundamental aspect of architecture that affects the likely social performance of buildings, then some sort of spatial analysis should be an integral part of architectural design process. Space Syntax as an analytic theory of architecture, has established a body of knowledge about spatial qualities of architecture, their likely social effects, and a methodology for analysing them. Besides, Space Syntax inspires the idea of syntactic design process as it suggests architectural design is configurative per se (Hillier 2007, 1). There are quite a few analytic tools for Space Syntax analyses, but to the best of our knowledge, there has been no direct application of Space Syntax in generative design in (recently parametric) CAD platforms[1]. This is partly because of technical difficulties in merging design and analysis platforms, mathematical complicacy of configurative design in itself, and perhaps, partly because of methodological neglect. Several researchers have worked on computational plan layout, most of which following evolutionary form-finding methodologies. A survey of such methodologies can be found in (Lobos, D, Donath, D 2010). We criticize viewing architectural design as an automated evolutionary process for two main reasons: First, because it implies that configuration is an order that can be 'found' through a course of automated trials and errors. We find this approach in fundamental contradiction to viewing architectural design as an 'intellectual' process aimed at 'proposing' configurations as to their desirable social implications. Second, we question the comprehensiveness of automated performance 'evaluation' in the mentioned approaches; especially because we question the soundness of 'automated evaluation' of social qualities pertained to configuration. Based on the work of March and Steadman (March, L, Steadman, P 1974), (Steadman 1983), we propose an alternative approach rooted in consideration of "how designers think" (Lawson 1980). We deem design process as an intellectual process of progressing ideas, which develop over time as "problem & solution" definitions (Dorst and Cross 2007). More specifically, we see architecture as being prominently about spatial configuration, i.e. interrelating functional spaces in a particular way to serve a higher order purpose. Whether we think about such matters explicitly or not; the outcome of an architectural design process is essentially a configuration; we therefore decided to make it an explicit exploration process. Our alternative is an interactive syntactic design process powered by real-time analytic feedback on spatial properties, in which designers have full intellectual control over spatial configuration and evaluation benefiting from computation in systematic analysis and synthesis.

Introduction

Our initial idea of an applied syntactic design methodology was based on bubble diagrams conventionally used by designers. The subtle fact about a bubble diagram is that it is a comprehensible configurative tool for designers and at the same time for computer programs and that it does not suggest a single geometric form. Bubble diagrams convey very important meanings that may not be seen easily by bare eyes; for instance they implicate which spaces are to be relatively private and which ones are to be communal and much more. We find it very important to reveal such meanings from the very begging of a design process, and report these meanings to the designer so that they can see whether the bubble diagram corresponds to their initial ideas about matters such as privacy and community of spaces. Our proposed system reads a bubble diagram in an intuitive way and translates it into a connectivity graph; it later provides the designer with Space Syntax measures; and eventually explores a particular class of plan layout patterns, which have the same connectivity pattern represented in the initial bubble

[1] Design usually is carried out within CAD platforms and available analytic software applications do not support CAD functionalities in general; therefore, we decided to bring analytic functionalities into a popular CAD platform. Our toolkit is developed in VB.NET that is installed as an add-on for Rhinoceros® CAD environment and its parametric modelling platform Grasshopper© .

diagram. These patterns can be used later as starting points by designers to elaborate their plan layouts. We need to give an overview of the last phase very carefully before we go further. The idea of exploring concrete (say geometric) plan layout patterns, which share an abstract connectivity pattern (a graph), brings about many questions: most fundamental of which is how many alternatives could be out there? In fact, a theoretical answer is that, in general, there could be countless number of plans sharing the same connectivity pattern. However, if we confine the search into a specific class of geometric shapes, such as rectangles, there can be a way to enumerate several alternatives systematically, according to the design inputs. A workable idea is to first reach at a topological embedding of a connectivity graph to bring it closer to become a geometric pattern. Throughout the process, in a few steps, the number of possibilities grows rapidly so that we need to select certain paths, methodically, to explore ranges of these possibilities (technically referred to as a design space). **Figure 1** depicts a schema of such a design space and the challenge of exploring it systematically.



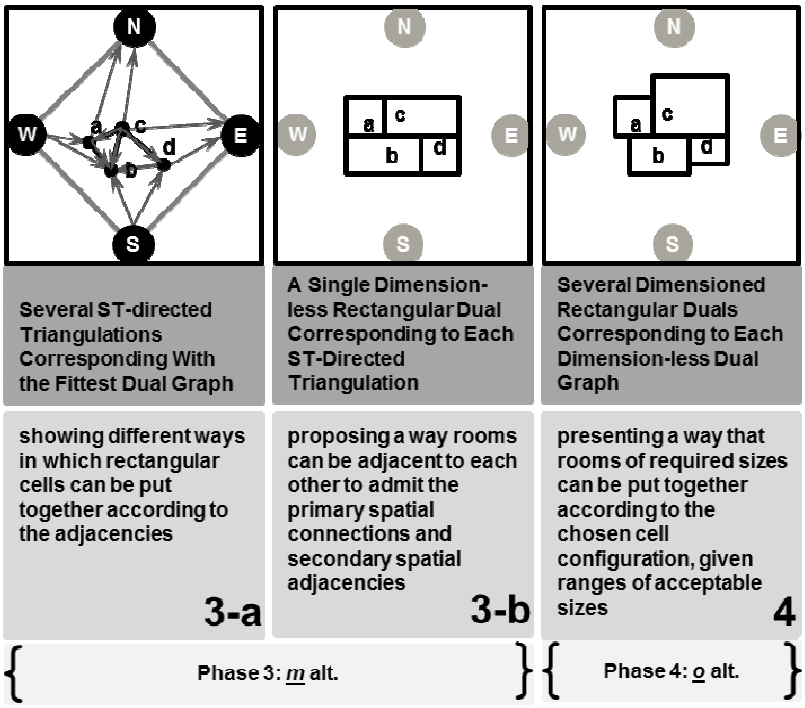


Figure 1: is a phase model schema of the proposed design methodology. Note that at phase 1, there is only a single alternative at hand, and then at the end of phase 2, we will have several alternatives, among which we can choose the one that has the best aptitude in terms of the size correspondence of its cells to the initial bubble diagram. We do this by choosing the one that its cells have almost the same size distribution as the given list of areas, using a 'minimum squared error' fitness criterion. This way, we reduce the amount of alternatives to one at this level. Later, we will potentially have m different dimensionless rectangular cell configurations, each of which permits several dimensioned plan layout patterns. Therefore, we could imagine that the total number of alternatives would be of orderm × o. So far, we have developed the algorithms of phase 0, 1 and 2.

The Design Methodology Put Forward by the Toolkit

Our design methodology is actually a fusion of what was proposed by Steadman (Steadman 1983, 69-75), Tutte's (Tutte 1963) convex drawing algorithm, an innovative force directed algorithm, several minor algorithms, and real-time Space syntax analyses. Following our design methodology, designer is free to insert a configurative idea and change it as they think is best, both at the beginning and during the process[2]. In a manner of speaking, our tools are meant to reveal meanings and implications of such inputs. The whole methodology can be comprehended looking at Figure 2.

[2] Our process does not automate the design process in any sense.

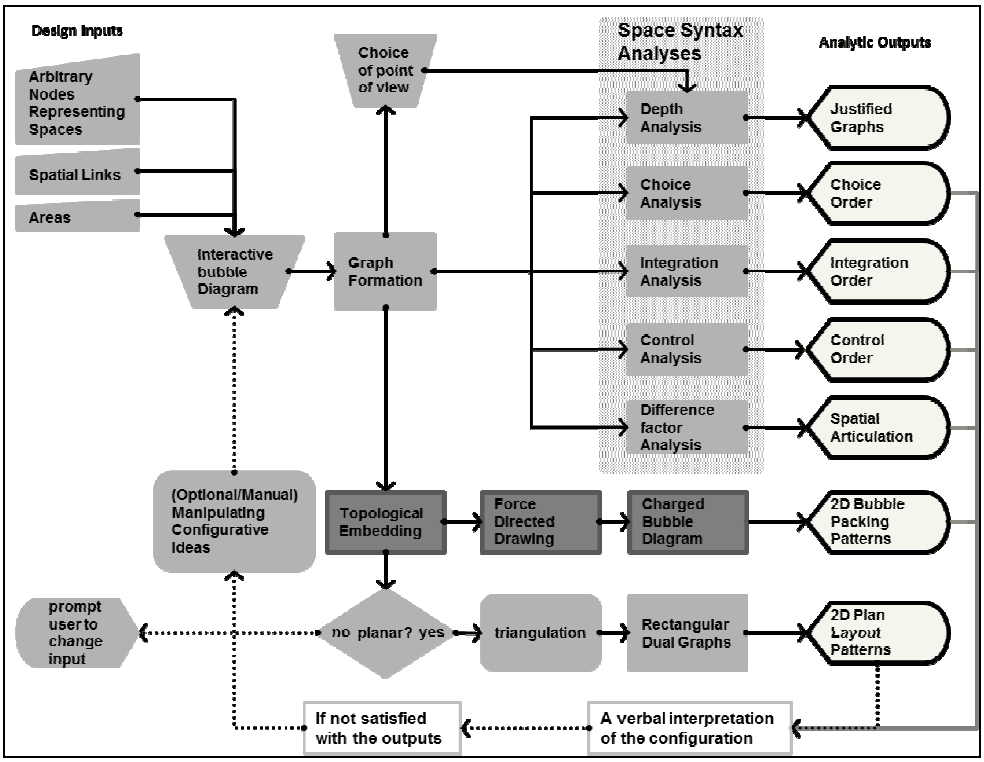


Figure 2: shows a flowchart diagram of the design methodology suggested by the toolkit. Without using a topological embedding like the one we have developed based on Tutte algorithm and further a 'high resolution' graph drawing method like the physical drawing algorithm we have developed, it is impossible to untangle messy bubble diagrams automatically. We believe that users should be able to sketch freely the inter-relations, without worrying about geometry at all. We leave the necessary changes in inputs for the users for we believe the interpretation of configurational qualities is very complex and dependant on contextual particularities, so that it can only be performed by a human intellectual thinker. This is why we have shown the feedback arrow differently: to stress the fact that this is not an automated feedback. Our tools do not perform any form finding or form optimization whatsoever; they merely reveal what is implicated by the user's configurative inputs.

The course of actions suggested by our proposed design methodology and supported by this tool suite is as technically described below (we have marked what is automated by the tools with bullets; the numbered items are what expected from the user to do with the tools). We could potentially combine most of the tools in a single box; however, we chose to let the curious user be able to explore different possibilities for combining the tools and provided them as a toolkit. The first sets of tools that a user needs are those shown in Figure 3. These tools cover the first six stages of our proposed design methodology introduced before.

1. Start with putting a number of arbitrary points as for defining the centre of functional spaces
2. Provide a list of (rough or exact)[3] area values for all functional spaces
3. Provide a list of spatial labels (names) for the functional spaces
   - A tool assigns rainbow colours to the functional spaces to make them more recognizable (see Figure 3).
   - A graph reader puts circles of sizes specified by the area values around all centre points (see Figure 3 and Figure 4).
   - The graph reader tool provides a sketchpad with nominal "North, South, East and West" sides for the user to draw the connections in (see Figure 3 and Figure 4).
4. According to your configurative ideas, draw a line between every pair of points (circles representing functional spaces) that you think should be directly linked. Add a few links to relate some of the spaces to the Northern, Southern, Eastern, or Western

[3] If the user wants to achieve a needed total area, it is enough to provide rough values and the tool puts out a set of area values that exactly sum up to the required area.

frontiers of the plan. These links are vital as they further guarantee that certain spaces be naturally lit in a desired way.

- The graph reader interprets the input links and points and their "label & area & colour" attributes as a graph (see Figure 3 and Figure 4).
- It provides the user with a verbal interpretation of links between spaces.
- It tells the user[4] if there *can* ever be a plan in one floor with such connections (corresponding to a planar graph)[5].

5. Provide the convex drawing tool (Shown in **Figure 3**, based on Tutte algorithm (Tutte 1963)) with the connectivity graph obtained in the previous stage, graph vertices, original centre points and NEWS (North, East, West, South) points derived from the graph reader.

    - This tool untangles the proposed bubble diagram and delivers a planar convex drawing of the connectivity graph[6]; and tells the user if a floor plan is admissible for the set of connectivity requirements; provides an ordering for automated justified graph drawing; and distinguishes a sub graph of the whole connectivity graph (excluding NEWS vertices). This sub graph, its vertices and its attributes will be used further on (See Figure 4).
    - This tool also generates error messages when the connectivity graph is not planar.

[4] According to the famous Euler formula (F + V - E = 2), this tool also tells the user the number of faces for a topological embedding of the plan graph on a plane for a potential planar embedding.

[5] In a later stage, we use the Tutte algorithm and theorem (Tutte 1963) for our definitive planarity test. If the input graph is planar and triconnected, then the drawing output by the barycentre algorithm is planar, and every face is convex.

[6] This is possible only if the graph is planar per se; in other words, if a planar embedding of the graph exists, it will be the output of this algorithm, meaning that if the output is not planar, the input graph is not planar.

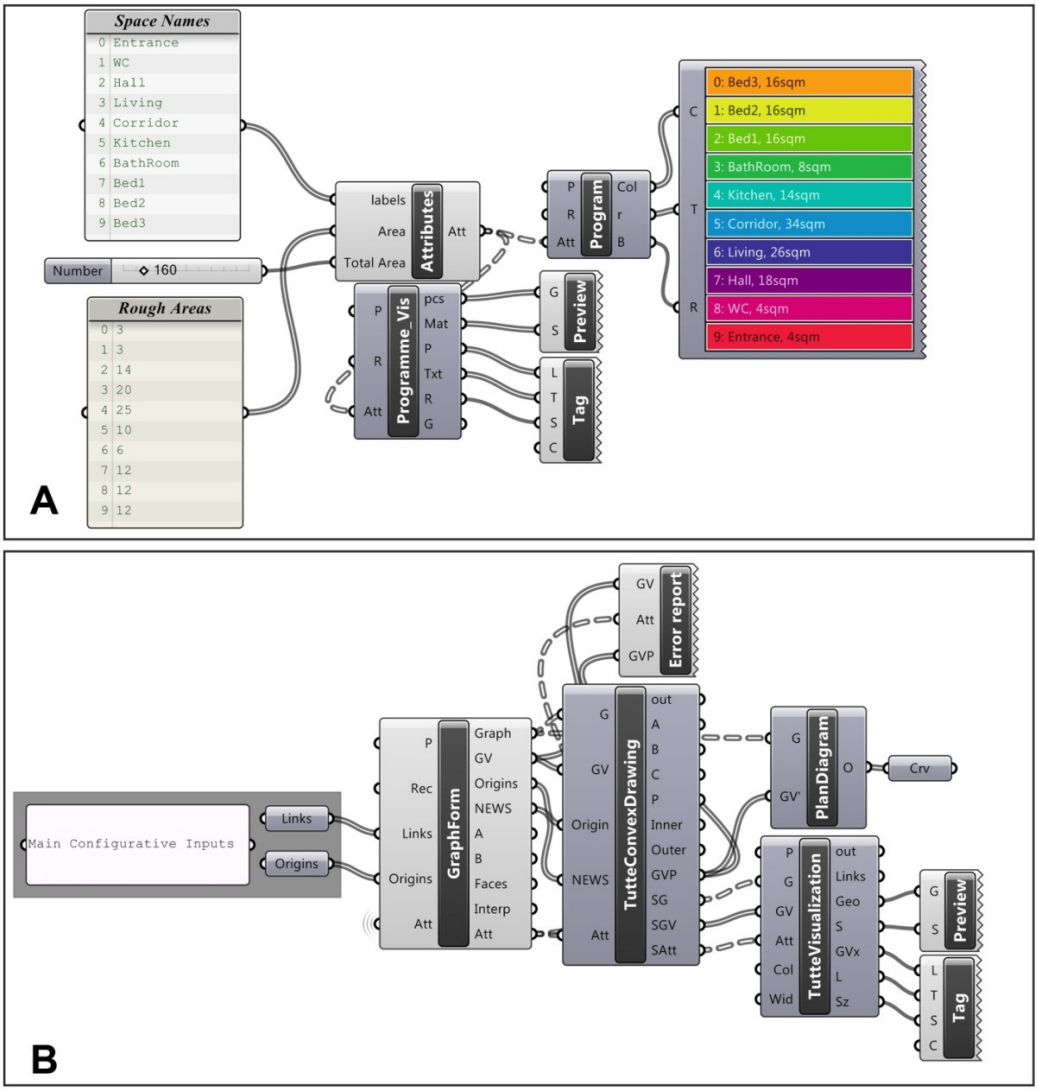


Figure 3: The set of tools in A are for formalizing a programme of requirements. One of the tools assigns arbitrary colours to the functional spaces to make them better recognizable. The set of tools in B, from left to right: Graph Form reads a graph of the connectivity and adjacency requirements, from a set of points that user puts in to represent the functional spaces. It immediately draws a sketchpad where designer draws a rectangle, and asks for necessary adjacencies with NEWS frontiers (as shown in Figure 4); then it asks for the links between functional spaces. The main output here is the graph, which is not in the right shape for further operations. This graph does not have a topological embedding until it is embedded with our modified Tutte algorithm for convex drawing. After this key operation, the algorithm delivers a convex planar embedding, a sub graph, a set of vertices and a subset of attributes that are vital inputs for all other operations. Our preliminary plan-layout component is also shown at the right top.

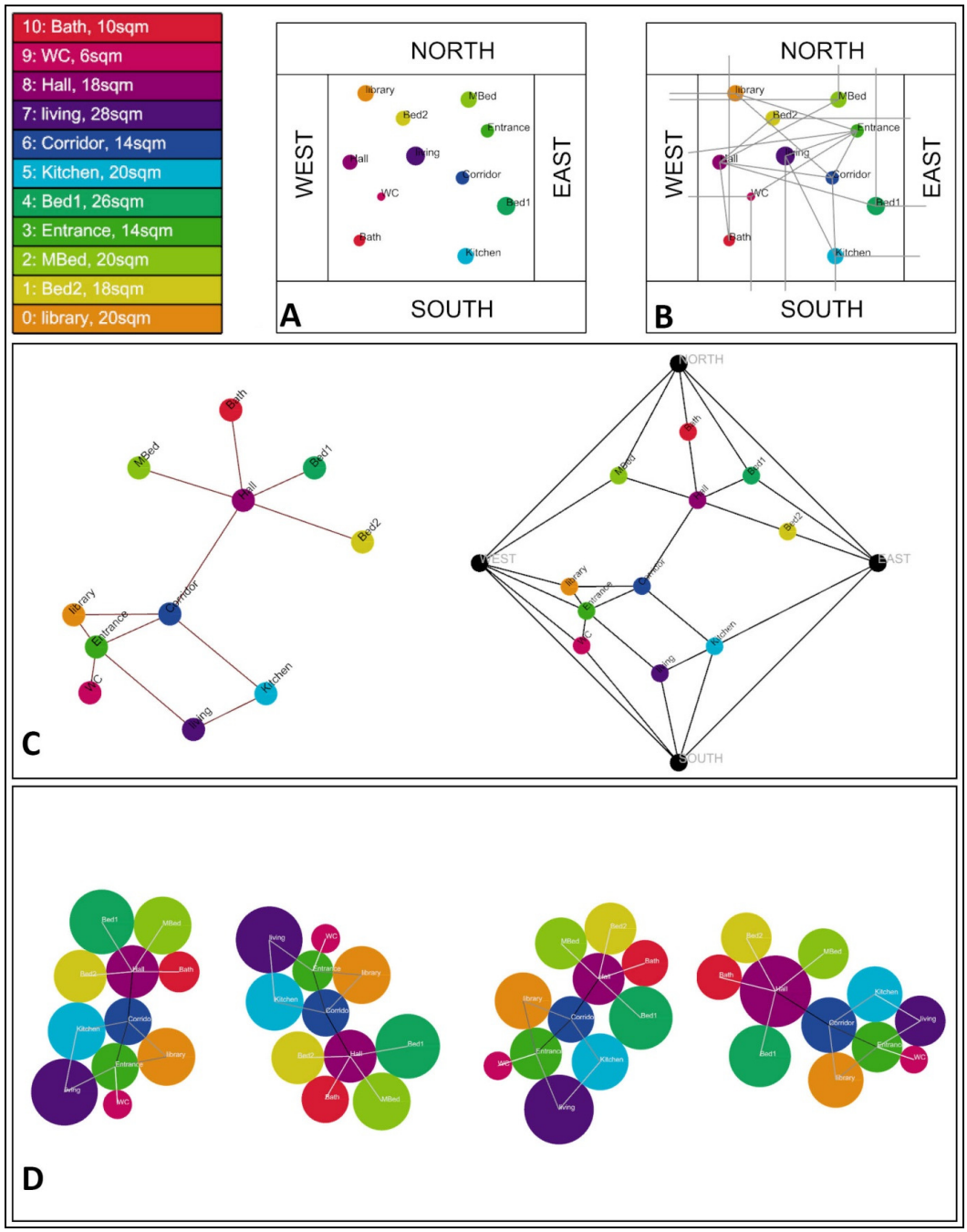


Figure 4: shows respectively: the sketchpad, a sample configurative input, an untangled convex drawing obtained from a customized Tutte algorithm, the sub graph containing exclusive information about the functional spaces, a kissing disk drawing obtained from our Force-Directed graph drawing algorithm (in this diagram the links between disks are coloured according to their segment choice values.)

6. Provide the space syntax components (see **Figure 5**, group D) with the sub graph found at the fifth stage, its vertices and its attributes.

   - The Space Syntax components perform the following analyses:
     a. It computes a list of depth-maps that is a set of graph-theoretical distances of all nodes (spaces) from one another. This is later used for automated justified graph drawing.
     b. It does the ***integration*** analysis according to the method introduced in (Hillier, B. Hanson, J. 1984); it also delivers a verbal report saying which

space is the most integrated (potentially communal) and which one is the most segregated (potentially the most private space) in the whole configuration. It also delivers a comparative table of integration values.

c. It calculates a list of ***control values*** as introduced in (Ibid).)
d. It computes the ***choice*** values as introduce in (Hillier, B, Shinichi, I 2007) for all spaces and for the links between them.
e. It finds and reports the ***difference factor*** as introduced in (Hanson 1998).

7. Provide the ***justified graph*** component with the depth-maps found in the previous stage, the sub graph found at the fifth stage, its vertices and its attributes; then you may choose a point of view to look at your proposed configuration literally from different points of views (see **Figure 6**).

   - This tool automatically draws justified graphs (Ibid) from whichever point of view that the user chooses. It is meant to make it easy for the user to feel the meaning of depth and integration by means of interaction. "This is the simple fact that a pattern of space not only looks different but actually is different when justified from the point of view of its different constituent elements. It is through the creation and distribution of such differences that space becomes such a powerful raw material for the transmission of culture through buildings and settlement forms, and also a potent means of architectural discovery." (Hillier 2007, 22). The algorithm embedded in this tool benefits from an ordering found out of Tutte convex drawing, that certifies 'good' graph drawings (with the fewest possible crossing edges[7]).

8. Provide the physical drawing tools with the sub graph and its vertices obtained from previous stage[8].

   This tool contains our force-directed graph-drawing algorithm and makes a "kissing disk" drawing of the bubble diagram[9]. Our force-directed component is quite intuitive and shows in real-time feasible planar spatial arrangements according to the specified areas and the connectivity graph (**Figure 4**).

[7] It is important to note that even for a planar graph, there is no guarantee that all justified graphs be possibly drawn without crossings.

[8] This algorithm and some other details are better explained in a forthcoming paper of us in eCAADe 2013 proceedings, named Designing with Space Syntax.

[9] This tools is quite efficient and makes such a drawing usually in about a second or less than that for a few small configuration as tested on a Intel® Core™ i7-2640M Dual Core (2.80GHz,4M cache) machine, usually in less than 500 iterations.

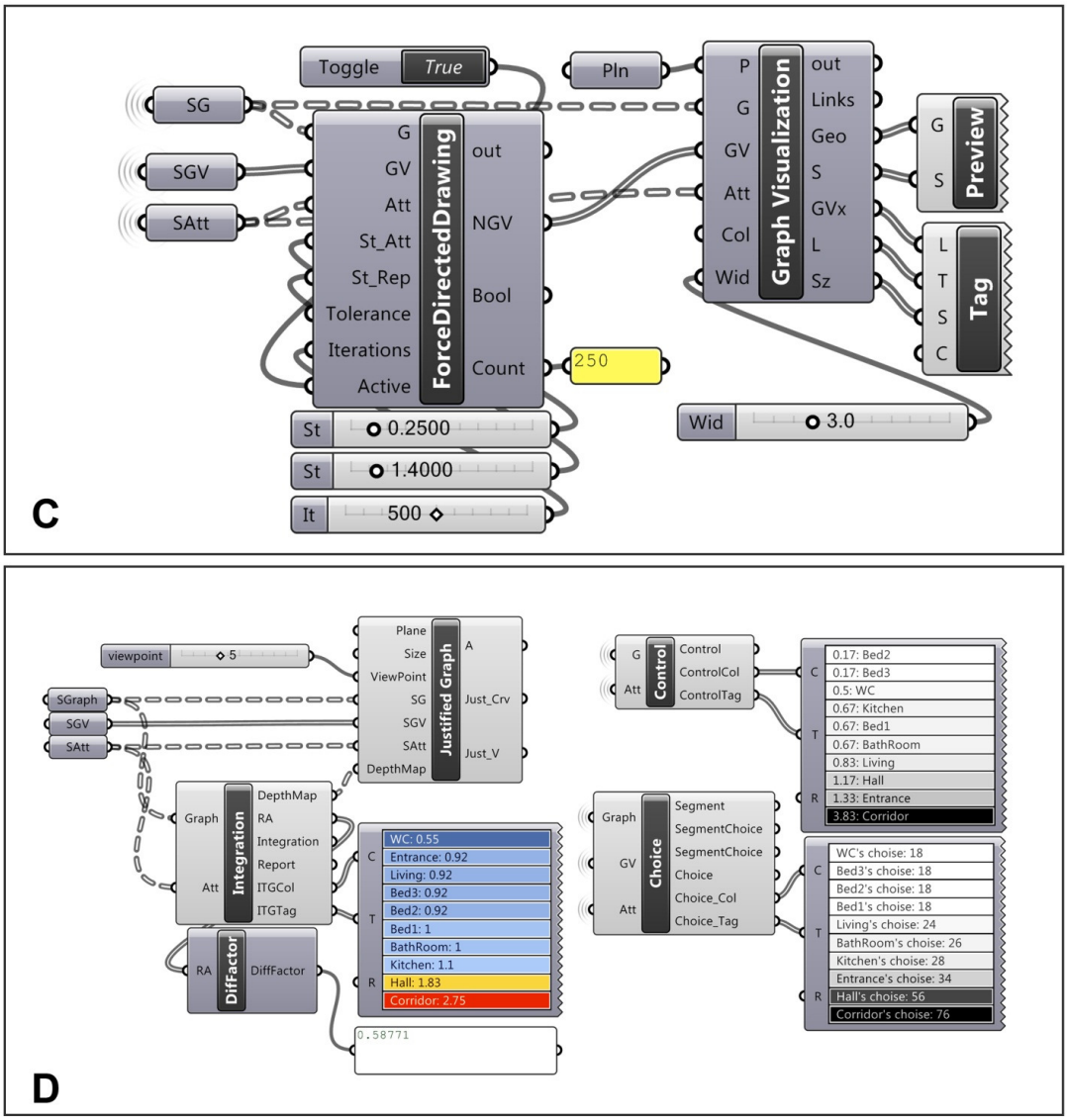


Figure 5: Group C shows our Force-Directed graph-drawing algorithm, which delivers a kissing disk drawing of a bubble diagram. This algorithm works by a set of attractive and repulsive forces acting recursively on graph vertices. Group D shows a set of tools performing Space Syntax analyses and reporting orders of Space Syntax measure on a sample architectural design assignment. Our Justified Graph component provides a unique opportunity for designers to draw justified graphs automatically in real-time.

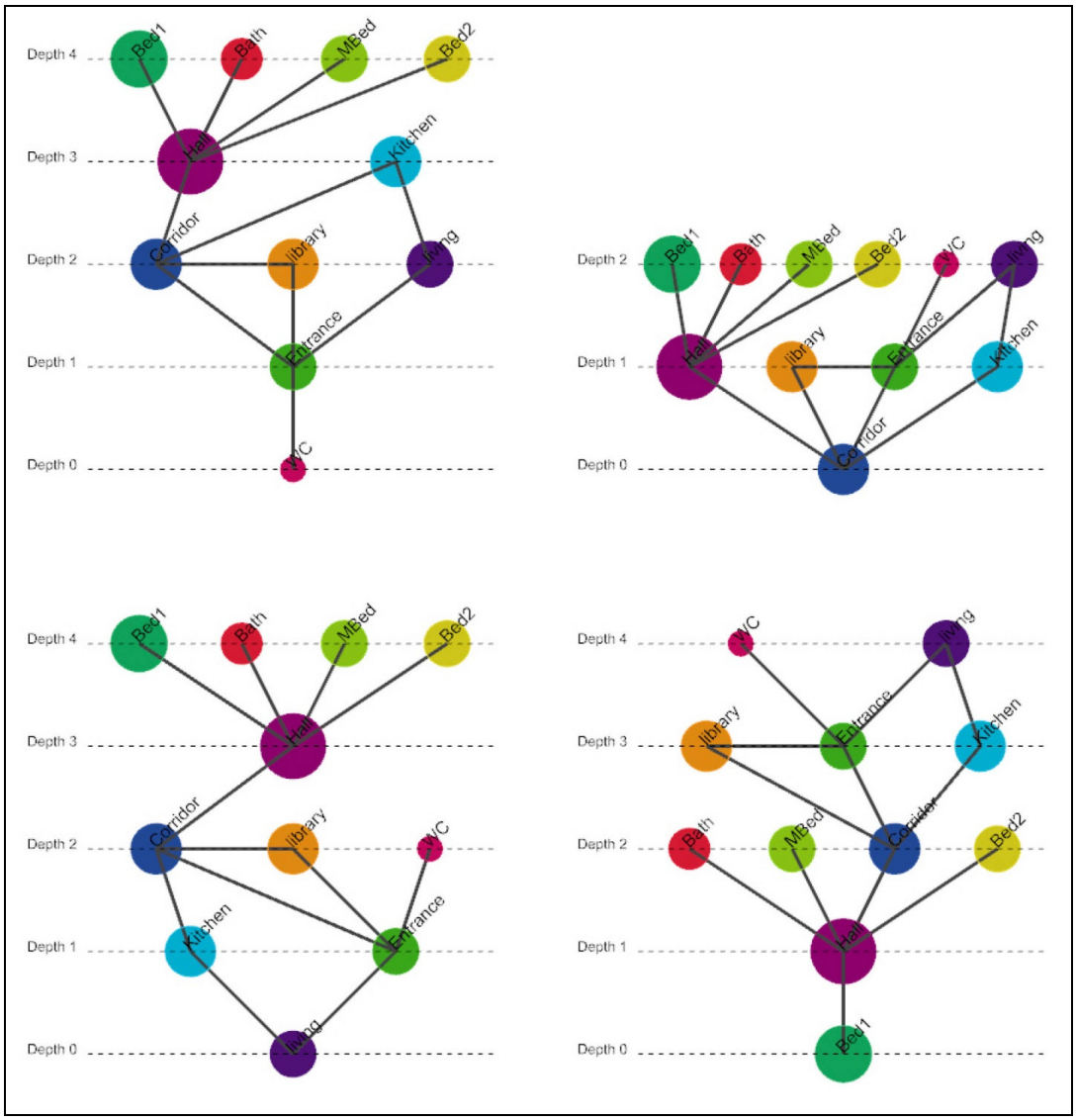


Figure 6: shows a few justified graphs drawing for the sample input discussed above. Our automated drawing algorithm uses an ordering that certifies a 'good' drawing with fewest possible crossings.

9. Deliver the sub graph and its vertices to a dual graph component (results shown in **Figure 7**)

    - This tool finds a dual graph for every admissible triangulation of the connectivity graph. A triangulation here implies that certain adjacencies, which had not been seen in the bubble diagram are being added to the input to make the whole configuration as compact as possible (we are still developing an automated triangulation algorithm) (see **Figure 8**).
    - After a catalogue of triangulations and their dual graph is made, the one with the minimum squared error (SSR) in terms of difference of its cells' area to the areas given by designer is chosen automatically as a root for further explorations. One of the triangulations in **Figure 8** is chosen as to its aptitude in terms of area distribution and worked out manually in **Figure 7**.

10. Hand in the chosen triangulation to the rectangular drawing component (this component is still under development. This tool is to reveal all possible dimensionless

plan layout diagrams sharing the same connectivity graph. It is important to mention that these diagrams just show possible spatial arrangements without suggesting any actual size or dimension. (Like the one shown in Figure 7).

11. Provide the dimensioning algorithm (still under development, introduced in (March, L, Steadman, P 1974), (Steadman 1983) and (Roth, J & Hashimshony, R 1988)) with the initial list of areas to generate dimensioned plan layout diagrams (Like the one shown in **Figure 7**). These diagrams can be starting points for the user in order to elaborate plan layouts.

12. You may choose to visualize graphs wherever you want by providing graph visualization components with a plane for location and orientation of the picture of the graph you want to see.

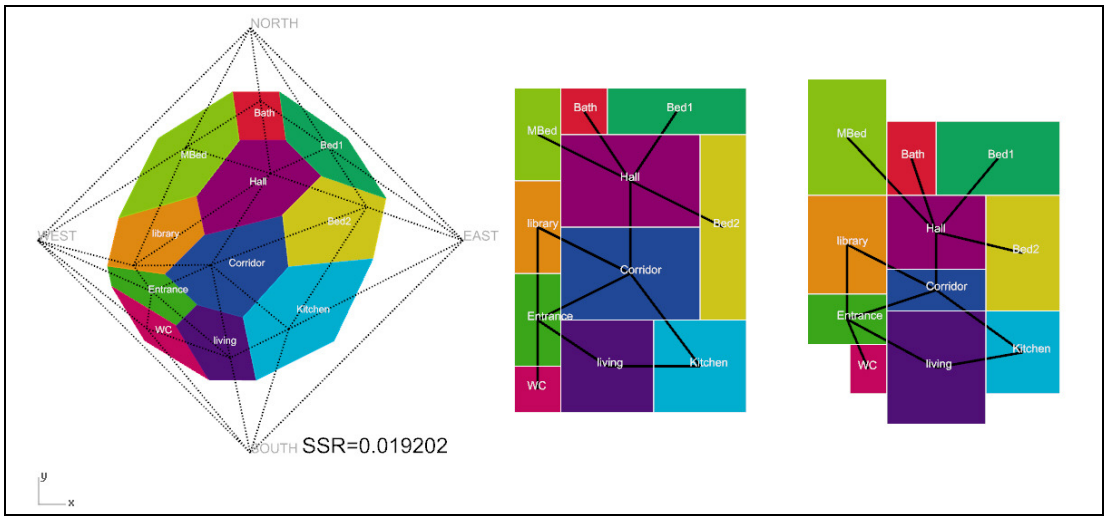


Figure 7: (Left) shows a triangulation of the augmented connectivity graph containing NEWS vertices, a dual graph corresponding to a fully triangulated connectivity graph, and its rectangular dimension-less drawing (middle); meaning that the present sizes and proportions of the cells do not indicate any actual sense yet. This rectangular drawing can be dimensioned later (like in Right), according to the initial requirements, by means of another algorithm. This is not the ultimate design outcome, but simply one of possible rectangular drawing of the proposed bubble diagram. The point is that when triangulating a connectivity graph, we need to introduce adjacencies, which are not necessarily required by the user, these adjacencies usually happen in actual plans to make them compact, in any case a connectivity graph is a spanning sub graph of an adjacency graph **(Steadman 1983, 71-72).**

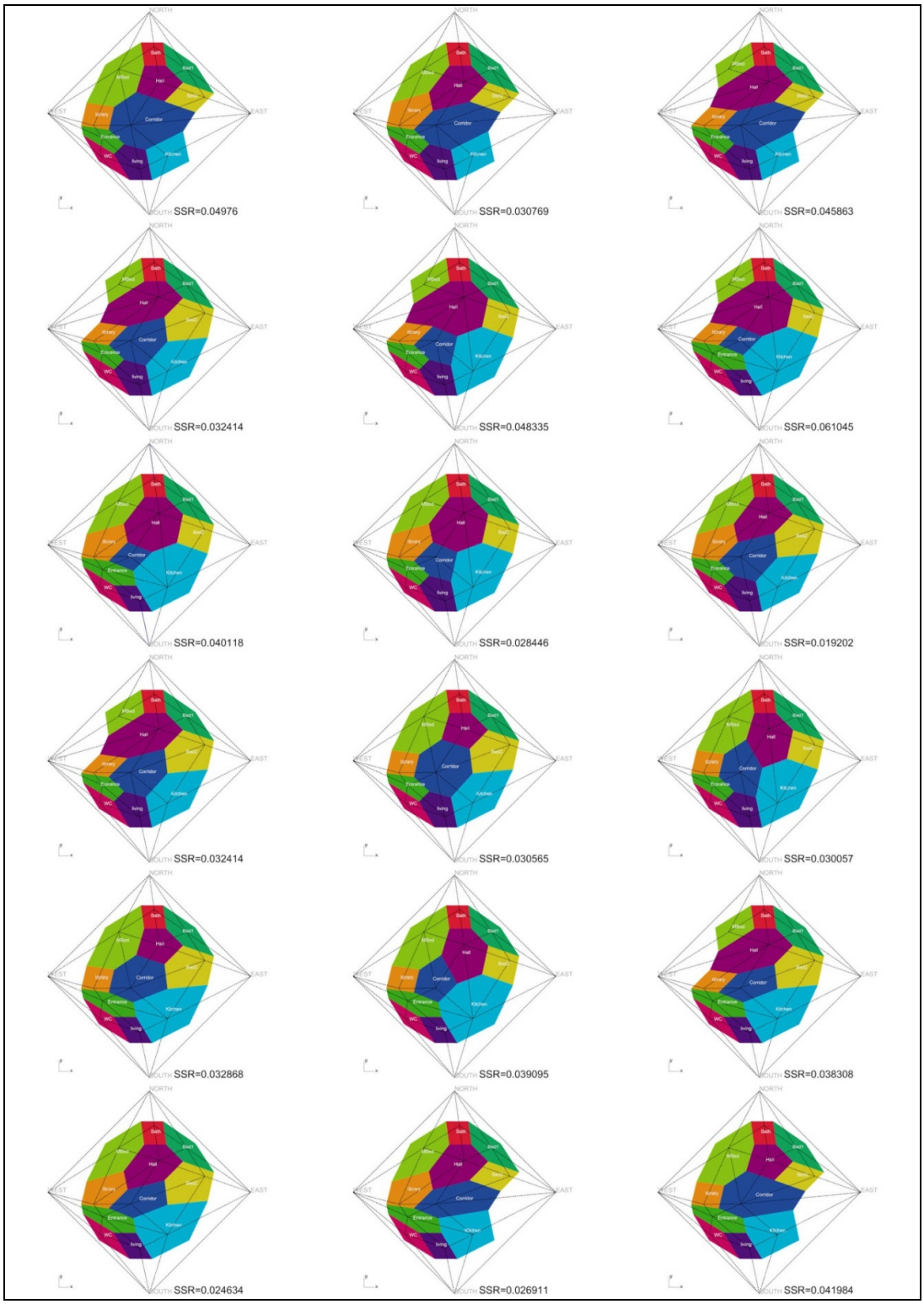


Figure 8: Figure 8 shows 18 admissible triangulations of the convex embedding of the connectivity graph, each of which ranked with a SSR (Sum of Squared Errors) score indicating the deviation of that alternative from the given list of areas.

Discussion

A single bubble diagram, with the help of some additional geometric constraints, can eventually be interpreted into a catalogue of feasible plan layout diagrams, all of which share the same configurational features (those pertained to topological properties only). In other words, a graph as a topological entity may correspond to a set of plan-layouts. A graph-theoretical design process, not only does not limit the creativity of a designer, but it actually enhances the designer's view and reveals multiple possible interpretations of a single idea put into a bubble diagram. We have succeeded in bringing the analytic power of space syntax theory into a CAD platform and initiated an integrated configurative design methodology; except that the last phase (the automated rectangular graph drawing) is still at a preliminary level and needs more developments. Regarding the drawbacks and limitations of our methodology, we should clarify one important issue: when sketching the functional interrelations, it is necessary for our convex drawing algorithm to be provided with connections to geographic out-sides. However, this is more a theoretical limitation than a technical problem, because the Tutte algorithm needs a set of fixed vertices to find a convex embedding of the inner vertices. It could also be that the convex drawing algorithm delivers results with extremely poor resolution for poorly connected graphs. This is again a theoretical limitation. In any case, our physical drawing algorithm is capable of untangling complex configurations and deriving at a 'good' drawing of them. However, we cannot theoretically prove its convergence, as there are no such proofs for other force-directed algorithms. Nevertheless, this method is practically quite effective and intuitive for graph drawing, and in our case proved very efficient as well.

As mentioned in introduction and explained in **Figure 1**, this methodology is only workable under certain limit conditions. It is not capable of exploring 'all' possible geometric counterparts of a single connectivity graph, not even for a particular class of cell configurations as rectangular dissections. We are still critically investigating the soundness of the selection method we use to choose a 'promising' triangulation. However, we have almost reached to the conclusion that without a selection at this stage the number of possibilities would be 'too many' in relatively large problems. Our main prospect is to overcome a few minor technical issues and publish the tools as a freeware package[10] to benefit from critiques of a large group of interested designers for improving the tools in action. We hereby acknowledge that our methodology still needs to be further tested in real-world design problems.

The main innovative aspect of our design methodology, apart from developing a HCI (human computer Interaction) interface for a sophisticated course of actions, is the unique combination of a set of formerly unrelated methods and techniques, e.g. using a Tutte convex drawing for untangling bubble diagrams.

References

Dorst, K., and N Cross. 2007. "Co-evolution of Problem and Solution Spaces in Creative Design." In *Computational Models of Creative Design*, edited by J. Maher, and M.L. Gero. Sydney: Key Centre of Design Computing and Cognition.

Hanson. J. 1998. *Decoding Homes and Houses*. Cambridge: Cambridge University Press.

Hillier, B. 2007. *Space is the machine*. London: UCL.

Hillier, B. and I. Shinichi. 2007. *Network and Psychological Effects in Urban Movement*. Melbourne: Springer.

[10] http://www.grasshopper3d.com/group/space-syntax

Hillier, B. and J. Hanson. 1984. *The Social logic of space*. Cambridge University Press.

Lawson, Brayan. 1980. *How designers think*. 4th, 2005. Burlington: Architectural Press, Elsevier.

Lobos, D, Donath, D. 2010. "The problem of space layout in architecture." *arquiteturarevista* 6(2): 136-161.

March, L. and P. Steadman. 1974. *The Geometry of Environment: An Introduction to Spatial Organization in Design*. M.I.T. Press.

Roth, J and R. Hashimshony. 1988. "Algorithms in graph theory and their use for solving problems in architectural design." *computer-aided design (Butterworth & Co (Publishers) Ltd)* 20(7): 373-381.

Steadman, P. 1983. "Architectural Morphology: An Introduction to the Geometry of Building Plans." *Taylor & Francis* 276.

Tutte, W T. 1963. "How to draw a graph." *Proceedings of the London Mathematical Society* 13: 743–767.