Amini Behbahani *et al. Visualization in Engineering* (2017) 5:17
DOI 10.1186/s40327-017-0056-z

RESEARCH Open Access

Viraph: exploring the potentials of visibility graphs and their analysis

Peiman Amini Behbahani[1*], Ning Gu[1] and Michael Ostwald[2]

Abstract

Background: Visibility Graph Analysis (VGA) is a space syntax method for quantifying some socio-spatial properties of the built environment by mapping the floor plan into a grid. Presently, VGA mainly relies on mean values of the measures while the individual room-to-room ("interspatial") relations are not considered or achievable by the common VGA software – depthMapX.

Methods: A new software package, Viraph, is developed using a weighed Dijkstra algorithm to calculate depths. The floor plan is mapped into convex areas to increase the speed of calculation. The interspatial relations are calculated by averaging the point-to-point depths in-between.

Results: The effectiveness of Viraph is demonstrated through a case study, which compares the houses from the late Victorian era with Wright's Prairie style with a detailed analysis of Francis Little House (designed in 1902). In summary, Viraph has showed a significantly higher calculation speed for VGA, and capability of capturing some socio-spatial relations of the styles by measuring the interspatial depth.

Conclusions: Viraph's improved speed for measurement has made VGA analysis more accessible for general design researchers and practitioners. Further this paper shows that the added feature of measuring the interspatial depth opens a new perspective for enhancing traditional space syntax analysis.

Keywords: Space syntax, Visibility Graph Analysis, Viraph, Francis Little House

Background

Space syntax theory offers several techniques to analyze various aspects of built environment regarding the usage, visual and programmatic configuration of space. One of the most elaborate approaches of the space syntax theory is Visibility Graph Analysis (VGA). This approach features a quantitative analysis of visual properties in the built environment, offering modelling and understanding of how the space may be used and perceived by its occupants (Ostwald 2011). Like other techniques of space syntax, VGA is based on a graph representation of the gross geometry of built environment. To create the representative graph, the space is articulated into a fine grid (usually with units in the size of a human shoulder or foot step). The grid is converted into a graph whose vertices represent the grid units (Turner et al. 2001). The edges of the graph are made between any two vertices (grid units) if the units are mutually visible (though there can be a metric limit for a visible distance). An advantage of this visibility graph is that the vertices retain the precise geographic properties of their corresponding points.

Following the creation of the graph, it is possible to calculate the common measures of space syntax. Among the measures, connectivity and depth are the most basic ones. The connectivity value for a grid cell is the number of visible grid cells from it, or the number of vertices directly connected to the corresponding vertex. This measure also represents the area visible from a point in the space. Depth (*D*) is the shortest distance between to two grid cells or graph vertices. In VGA, usually two non-metric definitions of depth are used: step and angular. Step depth is the minimum number of edges between two vertices. In the space, this number indicates the number of different straight paths a person must take to reach from one point to another. On the other

* Correspondence: Peiman.aminibehbahani@unisa.edu.au
[1]University of South Australia, Adelaide, SA 5001, Australia
Full list of author information is available at the end of the article

hand, the angular depth is the smallest degree of turns a person must take for the same task. The angular sum is represented by units of 90° (e.g., D = 0.5 indicates a sum of 45° turns) (Turner 2001a). Either of the depth values is usually represented as Mean Depth (*MD*), the average of all depths from one grid cell to the rest of the grid. Unlike graphs of other space syntax approaches (convex and axial maps), there is usually no further normalization of *MD* into relative asymmetry, integration value or intelligibility.

A visibility graph will usually be an extensive system which can only be handled by computer, even for small buildings. Therefore, visibility graphs and their analysis results are usually visualized by assigning colors to grid cells. The colors represent the respective values of a measure (be it *MD*, connectivity, etc.). Presently, the mainstream software for performing VGA is depthMap. The latest version is called depthMapX (Varoudis 2015). This open-source software was developed in 2001 at University College London (Turner 2001b) to produce and analyze convex and axial mappings and VGA results. While depthMap provides a common platform for VGA on a low-end computer system, the measurement may take a relatively long time (up to hours) on such systems depending on the geometry of the floor plan (or street plan). In such cases, the prolonged analysis may slow down the research process or tempt the researcher to reduce the grid resolution to get a faster but less detailed result. These two issues became the motivation for the authors to develop an alternative and faster tool which was ultimately named Viraph (standing for Visibility graph).

While Viraph's initial goal was to accelerate VGA procedure, several possible ways of approaching the raw depth data were emerged during the process of making and testing Viraph. Eventually, some of these possibilities were proved to be useful and relevant to the research and therefore were incorporated into the software. Hence, this paper introduces the Viraph on two grounds: its faster calculation algorithm in comparison to depthMap and the additional features originated from the emerged possibilities. It should be noted that the presented version of Viraph slightly differs (for the better) from the one reported in the first author's PhD thesis (Amini Behbahani, 2016) and is visually improved from the version reported in the ConVR 2016 conference (Amini Behbahani, Gu & Ostwald, 2016). These features are further demonstrated in the present paper through a case study – Francis Little House, designed by Frank Lloyd Wright in 1902, in addition to a comparative study including dozens of other Victorian-era and Prairie houses.

The paper is structured in three further sections. The first section presents the development of Viraph, its user interface and the comparison of its algorithm and results with depthMap's in three respective subsections. The second section discusses the new VGA measures and their possible use in future research. Finally, the last section summarizes and concludes the paper.

Technical features of Viraph

In this section, the technical features of Viraph are explained. The section is devised into three subsections. The first subsection presents the geometrical and algorithmic basis of the pathfinding operations. This subsection only focuses on the angular shortest path as the main issue regarding its time taken for calculation. The focus of the section is only on Viraph. To the knowledge of the authors, full details of the algorithm behind depthMap's calculation has not been formally documented. The second subsection compares the duration of calculation and accuracy of results of Viraph with those of depthMap and the third subsection presents the user interface of the software.

Calculation of angular path in Viraph

The basis of the calculation of the angular depth in Viraph is to divide the space into arbitrary convex areas (not to be confused with convex mapping of space syntax). Considering that all points inside a convex area are visible to each other, the border between two convex areas is also mutually visible to both areas. This axiom leads to two other obvious axioms:

- The shortest angular path between any two points in the respective convex areas always passes the borders between them (if there is no other convex area in the system).
- Therefore, the shortest path between any two points in the space will pass through a number of borders between convex areas, as long as the points are not located in the same convex area (Fig. 1).

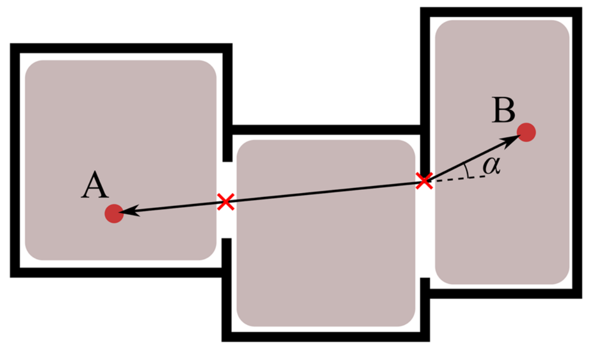


Fig. 1 The shortest path between two points (A and B) always passes through the borders (dashed lines) between the convex areas

Another axiom is the fact that in a concave quadrilateral, ABCD, (Fig. 2), the angle δ at the concave vertex D is always larger than the angle β on its opposing convex vertex B. This indicates that the supplementary angle of δ will be smaller (δ'). This smaller supplementary angle (δ') is the angular depth between vertexes A and C, and therefore, the path (*ADC*) is the shortest path between the two points.

It is possible to draw similar concave quadrilaterals for any mutually invisible points in two convex areas (Fig. 3a). In any case, the shortest path always passes through D (*ADC*) in Fig. 3a. In other words, the end points of the borders between convex areas are crucial in forming the shortest angular paths. In a larger set of convex areas where the shortest path will pass through multiple borders, this premise will still be applicable (Fig. 3b, the shortest path is *ADD'C* not *ABB'C*). Even if a segment of the shortest path passes through the middle of one or more area borders (Fig. 3c, shortest path passes through P on the border of convex areas 1 and 2) which simply means that the two sides of that segment are inside a same virtual convex area (because they are mutually visible). Therefore, the crossed area borders are irrelevant and unnecessary to be considered.

In Viraph, the convex areas are devised based on orthogonal sets of grid cells (unless they border angled boundaries). Hence, convex areas border each other along with horizontal or vertical strips of cells. To adjust the border end points on the grid network, the border is considered as a couplet of neighboring strips of the two adjacent convex areas (Fig. 4a). Thus, the connecting lines connect the center of the grid cells on the border strips not the vertices of the boundary corners.

In order to find the shortest angular path between any two points, it is only necessary to search the set of border end points and the lines connecting mutually visible end points (calling them *E-lines* hereafter) (Fig. 4b). For this purpose, Viraph creates a graph representing all E-lines (except for the lines connecting end points of the same border). Each E-line is

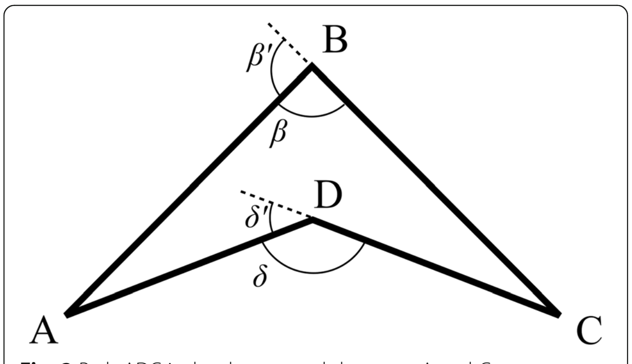


Fig. 2 Path ADC is the shortest path between A and C

recorded as two vectors of opposite directions (thus the graph has *2n* vertices, with *n* as the number of E-lines). Then, two databases are made: Table α that is a $k \times 2n$ table (*k* is the number of grid cells) containing the supplementary angle made by any $\mathrm{E-line}$ vector and any grid cell, and Table β, a $2n \times 2n$ table which contains the angular depth between all the E-line vectors. For Table β, the angles between the adjacent continuing vectors are calculated directly without pathfinding.

For finding the angular depths between the rest of the E-lines vectors in Table β, a variant of Dijkstra's algorithm (1959) for weighed graphs is used. The original Dijkstra algorithm (DA) is a step-by-step navigation through the graph to find the depth of other vertices from a specified vertex (V_0). The core concept of DA is that the passed vertices in each step always have a shorter depth to V_0 than vertices in the next step. Therefore, in each step (S_i), DA finds all the unpassed vertices which are neighbors of the vertices in the set V_i (i.e. vertices passed in step *i*) and records them in the set V_{i+1} and continues to the next step (S_{i+1}). DA concludes when no unpassed vertex is left. The depth of each vertex from V_0 is *i* (the index of V_i in which it is recorded). This makes sense because in un-weighed graphs the *number* of steps or connections between two vertices is the same as their distance. However, in a weighed graph, it is possible that two directly connected vertices have a larger distance than two vertices separated by several other vertices. Hence, in each step, DA should search for all neighbors of all recorded vertices in all previous steps to find the next navigable vertex with the smallest distance to V_0.

To address this issue in Viraph, the passed vertices (i.e. E-lines) are recorded in a *queue*. Each passed vertex also contains their angular distance (A) from the initial E-line (V_0). The queue is initially filled with only V_0 (A = 0). Then in each turn, Viraph looks for each neighbors (V_N) of each vertex in the queue (V_q) to find a neighbor with the shortest accumulative angular distance to V_0 (i.e. the minimum A_{Vq} + β[V_q, V_N]). Once a V_N with the shortest distance is found, it is added to the queue with its angular distance recorded in Table β. Meanwhile, V_q would be removed from the queue and deemed unpassable if all its neighbors are already included in the queue. Viraph proceeds to the next turn. This search continues until there is no vertex left in the queue (i.e. all their neighbors are already passed). In this case, the queue, the passed status and the A values are reset and Viraph restarts the pathfinding for the next E-line as V_0. Pseudo-code 1 shows the general outline of this pathfinding algorithm.

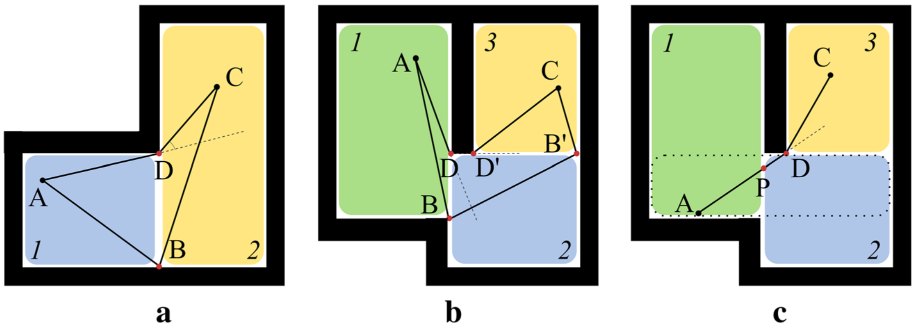


Fig. 3 Border end points on **a**) two adjacent convex areas, **b**) three convex areas, **c**) three convex areas when the shortest path passes through the middle of the border

Pseudo-code 1. The pathfinding algorithm for depth between the connecting lines of the borders (symbol // indicates beginning of comments).

```
for each V₀ in E-line graph {
    reset // queue, angles (A), passed status
    queue.add(V₀)
    mark(V₀) // as passed and added to queue
    min = 0 // minimum angle
    while queue is not empty {
      for each Vq in queue {
        for each unmarked V_N in Vq.neighborhood {
          if β[Vq, V_N] + Vq.angle <= min {
            min = β[Vq, V_N] + Vq.angle
            Q = Vq
            N = V_N
          }
        }
     }
     mark(N)
     β[N , V₀] = min
     N.angle = min
     if not Q.contains_unmarked_neighbor() {
        queue.remove(Q)
     }
  }
```

After obtaining the angular depth between all vectors and between vectors and grid cells, the depth between any two grid cells (C_i and C_j) is calculated by checking all connected vectors (V_i and V_j, respectively) to those grid cells as in the following Pseudo-code 2.

Pseudo-code 2. The algorithm for finding the shortest path between grid cells.

```
for each Ci in grid {
   for each Cj in grid {
      if visible(Ci, Cj) {
        min = 0
      } else{
        min = -1
        for each Vi in Ci.neighborhood {
          for each Vj in Cj.neighborhood {
            if ( β(Vi, Vj)+ α(Ci, Vi)+ α(Cj,Vj) <  min)
                or (min < 0) {
              min = β (Vi,Vj)+ α (Ci, Vi)+ α (Cj, Vj)
            }
          }
        }
        depth(Ci, Cj) = min
      }
      MDi += min / (k - 1)
   }
}
```

While Viragh supported by the above algorithms has been proven to be faster (in our practice) than depthMap, in some situations it still can take a considerable amount of time and memory. To address this issue, instead of the end points on border strips, the end points on the actual border line between the convex areas are considered (i.e. the line where borders face each other as marked in Fig. 4a). Geometrically, this end point is between the two adjacent

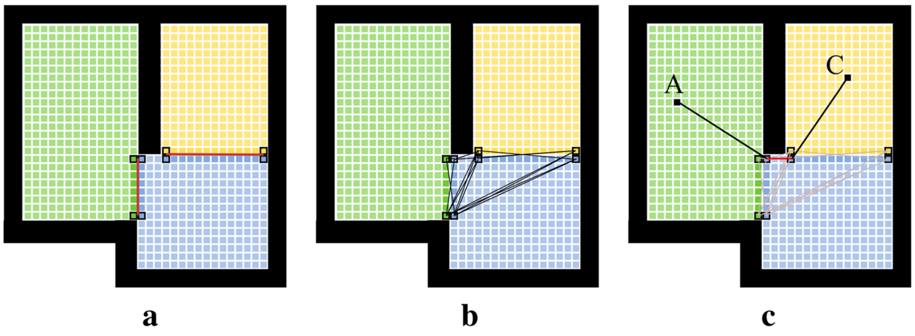


Fig. 4 **a**) border lines and end points between convex areas, **b**) E-lines connecting border end points, **c**) the E-line mediating the shortest path between A and C

end points as if those end points are combined into one (thus we called this approach "combined"). The end points of this line inherit the visual properties of the combined end points. This means that the new end point is considered visible from all the grid cells in both convex area despite the possibility that it is not actually visible from some cells. This treatment significantly reduces the number of E-lines and hence increases the calculation speed. However, it also reduces the accuracy of the results in the sense that "accuracy" is measured by how closely it abides by the strict grid articulation. The speed and accuracy of the results of both treatments are discussed in the next subsection.

Comparison with depthMap

In this subsection, Viraph is compared with depthMap (version X) in regard to the speed of calculation, accuracy, and memory usage. Firstly a summary of depthMap's algorithm is provided and then the results of a case study is presented.

To the knowledge of the author, depthMap's algorithm has not been formally documented. While the C++ source code (Varoudis 2015) is available but is too complex to be mapped completely. Therefore, we refer to various relevant sources that discuss some aspects and features of that algorithm. Firstly, the original developer of depthMap, Turner (2001b), stated that the pathfinding is done separately for each point (on the grid). He stated that the reason was not to consume a large memory considering the limits of memory size in early 2000s. Although his notion pertained to before the inclusion of angular analysis in depthMap, it still seems to be the used basis. This indicates that depthMap may have not been optimized through a table like β in Viraph and thus it recalculates several paths. Secondly, regarding the source code, depthMap apparently uses a "bin" system for pathfinding. In this system, the real-number angles are rounded and represented by integer steps. It seems that depthMap uses a 32-bin system (i.e. 8 "octant" bins for every exact 45 degrees, and 24 bins – 8 × 3 – for every 15-degree portions between two octants) to record angles and directions of movement. The bin system would significantly increase the calculation speed (as opposed to actual angles), as it is clearly stated in depthMap's interface for axial analysis and also in agent-based modelling in VGA (Koutsolampros & Varoudis, 2017). If this speculation is correct, it may reduce the accuracy of the measurement considering the maximum error margin of <15°. In any case, it seems that depthMap uses a variant of Dijkstra algorithm from this point to navigate through the graph. The bin system is only used for navigation. Once a path is found, the accumulative angle is calculated accurately based on the actual points on the path.

In order to compare the performance of Viraph to that of depthMap, a case study for the calculation of the angular depth is carried out for the first floor of the Francis Little House (floor plan adopted from Futugawa 1987) on a Dell Latitude 6420 with an Intel i5-2430 CPU, operating on Microsoft Windows 7 Home 64bit. Table 1 shows different features of the calculation. The results are shown for both Viraph conditions, with accurate border ends (A) and combined border ends (C). The comparison of the results is based on matching the geometrically closest points on either grid to each other. Once corresponding points are found, three comparisons are conducted for their *MD* values. First, the absolute difference between the values are measured. Second, the relative difference is measured (as percentile difference between corresponding values). Third, the ranking of the values is considered (i.e., to see what rank the nth visually significant point in depthMap has in Viraph's results).

The results show that Viraph is significantly faster than depthMapX (3.6 times in the condition A and 18 times in the condition C). However, it also consumes more memory (3 to 4 times) although a large portion of this memory is because Java's Garbage Collector does not return the used memory to the operating system. In addition, a part of the used memory is also related to retaining the point-to-point depth data for the interactive features as explained in Interactive visualization subsection. Regarding the correspondence of the results with those of depthMapX, the combined-border condition (C) scored better in both absolute and relative comparisons (3.6% or 2.2° difference compared to 6.6% or 4.0°). Majority of

Table 1 Comparison between Viraph's and depthMapX's calculation

Item	depthMapX	Viraph (A)	Viraph (C)
Grid resolution (*k*)	2150	2099	2099
Unit size (AutoCAD units)	40	40.7	40.7
Average unit point diff. (δ)	-	42%	42%
Convex spaces	-	27	27
Border end points	-	70	38
E-lines	-	1356	468
Average absolute difference	-	0.045 turn (4.0°)	0.025 turn (2.2°)
Average relative difference	-	6.6%	3.6%
Percentage of +5% differing results	-	64%	22%
Percentage of +10% differing results	-	14.8%	3.5%
Ranking difference (%)	-	2.3% (50 ranks)	2.5% (55 ranks)
Calculation duration	28:10 min	7:42 min (×3.6 faster)	1:32 min (×18 faster)
Memory used	72 MB	273 MB	230 MB

results (64%) for the condition A had more than 5% relative difference. This does not necessarily indicate the inaccuracy of the results in the condition A. Despite the potential inaccuracy regarding the possible use of the bin system in depthMapX, we think the main reason for the discrepancy between the results is the difference in grid positioning. There was a 42% difference (δ, out of maximum 71%, or half the diagonal length of a grid unit) in defining the location of grid unit points between the two applications. This difference is partially related to the rounding of coordinates by 10 and beginning the grid articulation from a round-number coordinates (Viraph begins the articulation relative to the top-left point of the plan). To verify this, a secondary depthMapX analysis was done with the floor plan being shifted by (+12, +10) CAD units (around 40% of a grid unit diameter). The results were averagely 5.1% different from the first depthMapX's result. This supports the notion that the difference in results is mainly due to difference in the positioning of grid cells. Regarding the difference in ranking of the grid cells, conditions A and C had 2.3% (50 ranks) and 2.5% (55 ranks) ranking differences from depthMapX results, respectively (i.e. if the point V_i has the 100th largest *MD* in depthMapX results, it would have the 50th or 150th largest *MD* in condition A). However, this difference is not necessarily significant as the *MD* values are very close to each other in depthMapX results (an average point has 64 other points with *MD* values of only ±0.01 or ±0.9° differences).

Nevertheless, the above results (and in general the faster calculation shown in our case study) are limited to certain settings which featured mainly orthogonal boundaries. This type of setting would produce relatively smaller number of convex spaces used to extract E-lines. Hence there will also be less E-lines and simpler graphs made by connecting them. Curves and recesses may increase the number of convex areas and E-lines however they will not fundamentally challenge the speed of Viraph. On the other hand, if there are numerous connections in the spaces (e.g., street maps or bazaars), the E-line graph would be intensive and time-consuming to calculate. Therefore, Viraph is not recommended for this type of plans.

On the other hand, because Viraph's core algorithm is vector-based, the calculation time is less affected by the resolution of the grid. This seems to be contrasting the factors affecting depthMap's calculation duration. The calculation time in depthMap seems to mainly increase, somewhat geometrically, by the resolution of the grid (i.e. in a *k*-unit grid, most of the calculation time correlates with k^2 while in Viraph, only a part of calculation follows this correlation).

User interface and file management

Viraph is developed in Java language (JDK 1.7+) with Eclipse Neon IDE. Theoretically, it can be operated on any device which supports JRE version 7 or higher. Figure 5 shows Viraph's user interface.

Similar to depthMap, Viraph also imports floor plans in DXF drawing formats (version R12). It exports results in two formats including the JPEG image from the display panel (with some options to control the exported data) and CSV spreadsheets of the numerical results. An important feature of Viraph (as explained in Interspatial depth and visibility subsection) is to select polygonal portions of the space and

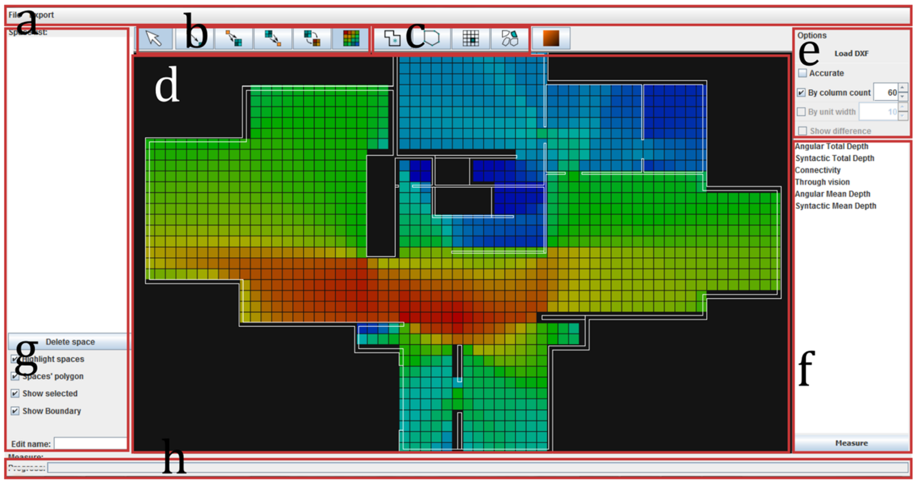


Fig. 5 Viraph's user interface. **a**) main menu including file management and export, **b**) display modes which control the user interaction with the display panel, **c**) operations (selecting and drawing points and spaces), **d**) main display panel, **e**) floor plan and grid options, **f**) measures and calculations, **g**) lists and options of spaces (name of spaces and their visualization options), **h**) status and progress bars

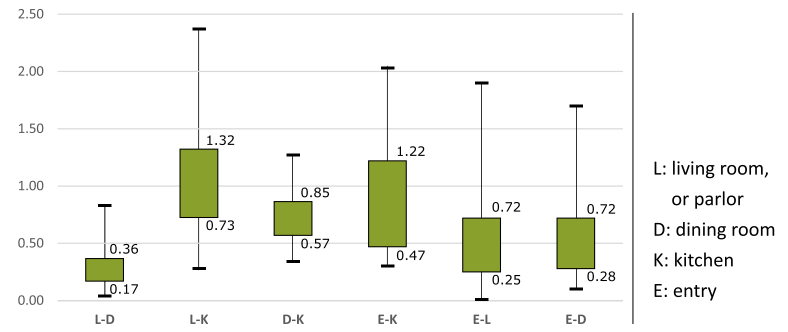


Fig. 6 The average angular *ID* between major spaces in houses of the late Victorian era and Wright's Prairie style

calculate their visual depths from each other. This type of results can also be exported as CSV files while it is also possible to be saved as a JPEG visualization of the portions.

Visibility graphs and their analysis

While the initial reason of creating Viraph was to have a faster angular VGA calculation, a number of potentially useful features have emerged during its development. In this section, two of them including interspatial relations and interactive visualization are discussed.

Interspatial depth and visibility

VGA measures in depthMap software are mainly represented as a collective value (mean or total) for a single spot. The importance of the one-spot *MD* is usually for identifying the least and most visually integrated spots. The collective depth values are obviously obtained from the values of individual point-to-point depths. However, the individual point-to-point depths are not usually a matter of research interest because there is rarely a semantic importance to the visual relationships between such single points in the space.

On the other hand, at least in theory, the visual depth between two rooms or portions of space reveals the degree of visual connection and possible flow of visual information between them. This type of visual depth may be useful to analyze the visual interaction between rooms. In this paper, this visual relation is called *interspatial depth* (*ID*). The challenge is that despite its theoretical prominence, to the authors' knowledge this measure is not explored by the literature and is not provided by depthMap results.

To address this issue, a feature has been added to Viraph to calculate the visual depths between the portions of a visibility grid which are defined by the user as polygons. The formulation for this measure is simply the measurement of the average of step and angular depth values between all grid units of two portions of the visibility grid. Equation 1 shows the formula for calculating *ID* between two spaces (*a* and *b*). In this equation, V_a and V_b are points (grid units) in spaces *a* and *b* respectively, and *D(x, y)* is the visual depth between any *x* and *y*; *k* represents the number of grid units in respective spaces. These depth values are recorded within the process of finding *MD*. This equation is valid also for step depths and any other measure like connectivity (where *V* values would be either 0 or 1). Notwithstanding the theoretical validity of *ID*, the significance of this measure is not yet verified thoroughly.

Equation 1: The formula for calculating ID.

$$ID_{a,b} = \sum_{i=1}^{k_a}\sum_{j=1}^{k_b} D(V_{a,i}, V_{a,j})/(k_a k_b) \tag{1}$$

In order to demonstrate the possible use of this measure, a case study was carried out featuring forty-two houses including those from the late Victorian era (fifteen houses, c. 1880-1890) and of the Prairie style (twenty-seven houses, designed by Frank Lloyd Wright between 1900 and 1913). The selection criteria, floor plans and preparation of cases are extensively introduced in the authors' earlier publications (Amini Behbahani, 2016; Amini Behbahani, Ostwald, & Gu, 2016). In summary, they were houses of simpler layouts for middle-class suburban clients in the mid-west United States. For visualization purposes, only one house - Francis Little House of the Prairie Style – was illustrated in this paper.

A relevance of the interspatial relationships and the domestic architecture in that era and style is that one of the key architectural values was the segregation of functions, people's genders and classes (Volz 1992; Grier 1992) which comes generally under the concept of *domesticity*, the separation of public and private life (Rosner 2005). A typical house was generally divided into three carefully segregated zones including social,

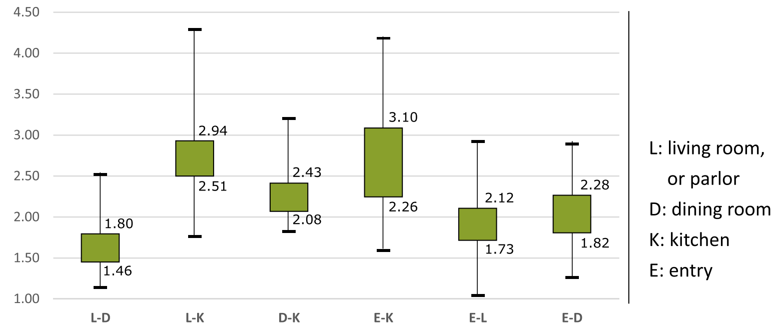


Fig. 7 The average step *ID* between major spaces in houses of the late Victorian era and Wright's Prairie style

service and private zones (Cromley 1996). The social zone included all spaces which were accessible by visitors, such as entry, dining and living rooms and hall. This zone was also where the most of social interactions between family members occurred. The service zone included kitchen and supplementary spaces (e.g., pantry, laundry, etc.) whose main role was to provide food as well as accommodate servants. Finally, the private zone hosted bedrooms and similarly personal spaces. This zone was usually placed on the second level of the house (as in the selected cases for this paper). Regarding segregation, the service zone was considered undesirable because of odor and mess, and was occupied by the lower-class servants (Whelan 2011; Cromley 2012). This particularly mattered for living room (or parlor) and entry rather than dining room (which was functionally related to kitchen). Since the last quarter of the nineteenth century and especially in Frank Lloyd Wright's opinion there should have been a closer interaction between family members and hence the relevant social spaces have been introduced in the first level of the house (Maddex 2002). Regarding the visual aspect of this relationships, one should expect that the undesirable service zone was hidden away from the eyes of the visitors and daily social life of the owner-habitants. In the space syntax context, this "hiding" is represented by higher visual depths between the service and social spaces (in comparison to the rest of the house).

In the case study, the angular and step visual depths between four major functional spaces (living room, dining room, kitchen and entry) were obtained. Figures 6 and 7 illustrate the results for the respective measures. In these figures, the spaces are represented by their initial letters (in Victorian era, there was a space called "parlor" which later refashioned as "living room". We also used letter "L" to represent this space). The angular *IDs* (Fig. 6) from kitchen to living room and entry were the highest (1.06 and 0.90, respectively) which indicate that the lowest degree of visual interaction in the house was between the "messy" kitchen and these two social spaces, as expected. The angular *IDs* between any two of entry, living and dining rooms were much lower (0.29 to 0.58) which show more interactions between the social spaces. The same ranking of depths is observed for step *IDs* (Fig. 7). The depths from the kitchen to the living room and entry are the highest (2.76 and 2.72, respectively) while living and dining room are visually the closest to each other (1.68). Overall, this measure fulfilled the expectation and successfully demonstrated a cultural characteristic of the era in the selected cases: the kitchen was visually distant from the living room and entry while the three social spaces were closely interacting.

A further analysis reveals the usefulness of this measure for understanding differences between the

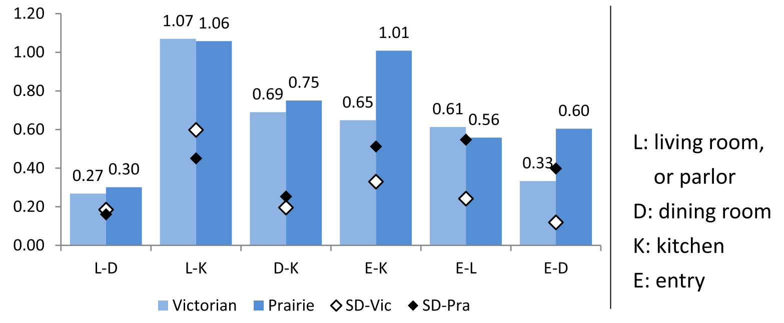
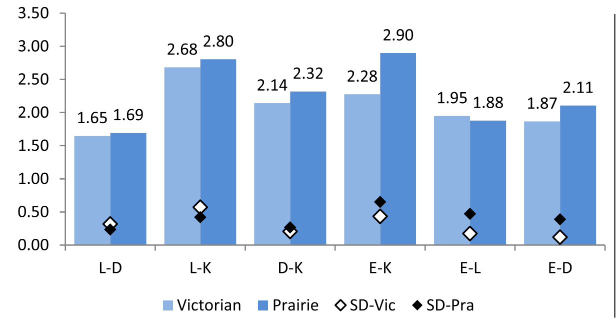


Fig. 8 Angular *IDs* between major spaces in houses of the late Victorian era and Wright's Prairie style



L: living room,
or parlor
D: dining room
K: kitchen
E: entry

Fig. 9 Step *IDs* between major spaces in houses of the late Victorian era and Wright's Prairie style

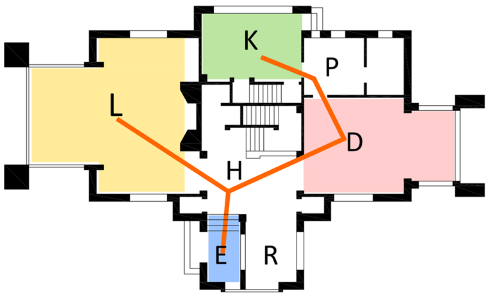
houses of the respective styles. Figures 8 and 9 show the angular and step *IDs* between major spaces in Victorian and Prairie houses (in these figures, SD stands for standard deviation). While in four interspatial connections there is minimal differences between the two designs, the Prairie houses feature significantly higher angular depths from the entry to the kitchen (1.01) and dining room (0.60). The former depth is also higher in the step measures (2.90). These results suggest that the Prairie houses were fulfilling the virtue of segregation more efficiently. This may partially explain the anecdotal and qualitative observations that the Prairie houses were more appealing to the conservative clients (Twombly 1975) and that they resolved issues of servant circulation (Wright 1960).

Regardless of style, these visual distances are generally achieved by putting the kitchen at the rear of the house and breaking the access path to it through buffer zones (Cromley 2012). For an example of this design approach, Fig. 10 displays the position of these four spaces in the Francis Little house, accompanied by the results of the measurements. As seen in Fig. 10, the L-K and E-K depths are significantly higher than other depths in the house for both measures (over 4 for step depth and over 1.3 for angular depth). The hierarchy and breaking of the access path is illustrated by the orange lines.

Interactive visualization

MD of a point in the space reveals an overall degree of visual relation between that point and the rest of the space. However, it does not provide any information on how the rest of the space is viewed from that point. The measure of *ID* addresses this issue for any two portions of the space. However, the exported results (as a spreadsheet) could only be useful when the number of portions are small enough to comprehend them without using a data analysis software. When the number of portions are higher (to the maximum of *k*), it is possible to use a visual representation of the results the same way the colors on the grid are used in depthMap.

For this purpose, Viraph provides an interactive visualization of the results based on the selected *display modes*. A display mode defines the types of two portions of the space for which the results are displayed. In the present version, there are four display modes: point-to-point, point-to-space, space-to-space and mean value. The point-to-point display mode enables the user to see the visual relation between any point and the rest of points in the space, individually, by simply hovering mouse cursor on that point. The point-to-space display mode shows the average of a measure for a portion of space (as defined polygons) from the viewpoint of the cursor's position. The space-to-space mode is similar to the point-to-space



Relation	Angular	Step
L-D	0.52	1.94
L-K	2.07	4.29
D-K	0.60	2.57
E-L	0.25	2.15
E-D	0.49	1.87
E-K	1.31	4.18

Fig. 10 Angular and step *IDs* between major spaces on the first level of Francis Little house

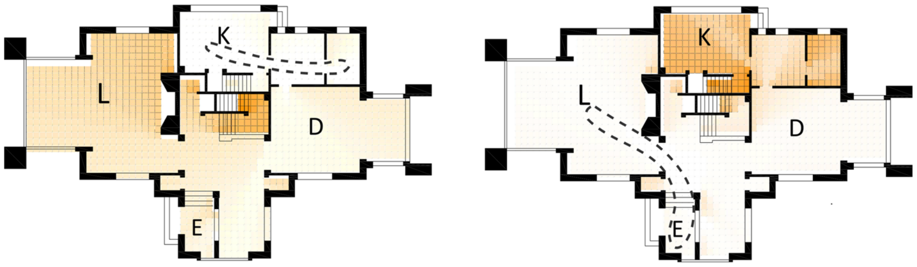


Fig. 11 The space-to-point visualizations of angular depth for the servant's view (*left*) and the visitor's view (*right*)

mode except that it visualizes the interspatial value if the cursor position is part of a defined portion. Finally, the mean value mode is the same as the existing visualization in depthMap: only showing the mean value for each point.

The potential advantage of the diverse display modes is to understand the visual relations from different points in the space in a very rapid way (all the interactive calculations for the point-to-point depths are already included in the *MD* calculation explained in Technical features of Viraph section). One other potential use is to analyze visual properties of space from the viewpoints of different people. For example, Figs. 11 and 12 show the angular depth and connectivity from the visitor's and servant's viewpoints of the house, respectively. Both figures show the space-to-point display mode in which the respective paths are outlined by dashed boundaries. The boundaries show an estimated approximate movement path of the servants or short-term visitors (excluding those who may stay for dinner). In both figures the darker shades represent the higher value of the respective measures (i.e. in Fig. 11, they are less visually connected to the selected part while in Fig. 12 the lighter shades indicate lower degree of visual connection).

The figures show a clear difference between the visual experiences of servants and visitors. In both measures, the service area is the most distant or least visually connected to the likely movement path of a visitor. Meanwhile, the service area does not visually interact with the social area of the house except the dining room which is functionally related to it. Overall, the results suggest the usefulness of this feature of Viraph in the context of analyzing Francis Little house.

Conclusion and future directions

This paper has presented Viraph, a new computational tool based on space syntax theories for visibility graph analysis that is used to model and understand certain visual properties of the built environment. Viraph features three major capabilities including faster calculation of the angular measures, different approaches to clustering and averaging of the results, and interactive visualizations of the results through its interface. These improvements have the potential to contribute to space syntax research and building analysis. Firstly, the analysis would be more time efficient especially on low-end systems commonly used by students and researchers. The higher calculation speed in the combined-border mode could even be suitable for testing a larger number of designs (e.g., outcomes of generative design procedures) or even during design process.

Another contribution of Viraph is to introduce the new measure of *ID*. This measure offers new perspectives to understanding visual relationship between different portions or points in the space. As briefly demonstrated in the case study of this paper, such measures can be used for studying aspects of space which have not been directly quantifiable by the existing

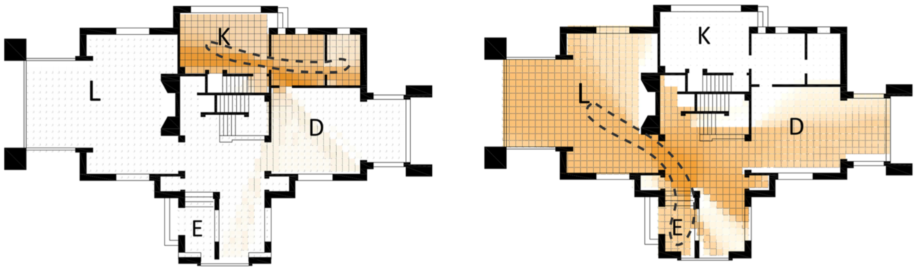


Fig. 12 The space-to-point visualizations of connectivity for the servant's view (*left*) and the visitor's view (*right*)

methods. Finally, the interactive interface of Viraph enables a quick analysis of the visual features without the need to refer to the extensive exported results in a spreadsheet file.

Nevertheless, Viraph can also be limited by its higher memory usage. This issue is partly due to the non-optimized garbage collection options of Java, but can be addressed by more suitable memory management in future versions. The current version of the software also uses arbitrary convex boundaries for creating the E-line graphs. It is possible to use a genuine space syntax convex mapping (whether drawn manually or automatically like the medial axial skeleton algorithm used by Miranda Carranza and Koch 2013). In this way, it will be possible to combine the usefulness of VGA and convex mapping to analyze programmatic configuration of the built environment (Ostwald 2011). Another main limitation of Viraph is for analyzing spatially and geometrically fluid spaces which may lead to an intensive E-line graph. Such a graph may significantly reduce the speed of calculations. This possible limitation will also be further studied and addressed in future versions.

Currently, the authors are developing an alternative version in C# language which will be more suitable to adapt for scripting in Revit and AutoCAD to facilitate integrated analysis within design process. This version will also be useful in more platforms (Windows, Linux and iOS). In parallel, the authors are exploring another approach to VGA analysis by considering transparency of barriers. This option will also be included in the next version to allow researchers to explore visibility separately from access.

The software, a brief guide and its source code (in Java) are uploaded on SourceForge (Viraph [Computer software] 2017) for reference and critique.

Abbreviations
A: Angle (in algorithms), Accurate (in Viraph approaches); C: Grid cell (in algorithms), Combined (in Viraph approaches); D: Depth (measure) as *D*, Dining room (in spaces in the house); DA: Dijkstra's algorithm; E: Entry; E-line: Endpoint line; ID: Interspatial depth; K: Kitchen; L: Living room or parlor; MD: Mean Depth; N: Neighboring; Q: (of) Queue; V: Vertex; VGA: Visibility Graph Analysis

Acknowledgements
Not applicable.

Funding
Not applicable.

Authors' contributions
All authors contributed to the conceptualization and development of research and also the manuscript. The first author (PAB) was in charge of the draft development and version management of refinement and revision. The second (NG) and third (MO) authors proposed the paper's structure and edited the paper and contributed to the revision of the manuscript. All authors read and approved the final manuscript.

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]University of South Australia, Adelaide, SA 5001, Australia. [2]University of Newcastle, Callaghan, NSW 2308, Australia.

Received: 9 March 2017 Accepted: 14 August 2017
Published online: 11 September 2017

References
Amini Behbahani, P. (2016). *Spatial Properties of Frank Lloyd Wright's Prairie Style: A Topological Analysis* [PhD thesis]. Australia: University of Newcastle.

Amini Behbahani, P., Gu, N., Ostwald, M. (2016). Viraph: enhancing architectural design visualization with results of visibility graph analysis. Paper presented at ConVR 2016 Conference. Hong Kong.

Amini Behbahani, P., Ostwald, M., Gu, N. (2016). A syntactical comparative analysis of the spatial properties of Prairie style and Victorian domestic architecture. *The Journal of Architecture, 21*(3), 348–374.

Cromley, E. C. (1996). Transforming the food axis: houses, tools, modes of analysis. *Material Culture Review, 44*(1), 8–22.

Cromley, E. C. (2012). Frank Lloyd Wright in the Kitchen. *Buildings & Landscapes: Journal of the Vernacular Architecture Forum, 19*(1), 18–42.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.

Futugawa, Y. (Ed.). (1987). *Frank Lloyd Wright monograph 1901–1906*. Tokyo: A. D. A EDITA.

Grier, K. C. (1992). The decline of memory palace: the Parlor after 1890. In American Home Life: 1880–1930 (pp. 49–74). Knoxville: University of Tennessee Press.

Koutsolampros, P. & Varoudis, T. (2017). Assisted agent-based simulations: fusing non-player character movement with space syntax. In T. Heitor, M. Serra, J. P. Silva, M. Bacharel & L. C. da Silva (eds). *Proceedings of the 11th International Space Syntax Symposium* (pp. 164.1–164.13). Portugal: Instituto Superior Técnico.

Maddex, D. (2002). *Frank Lloyd Wright: inside and out*. London: Pavilion.

Miranda Carranza, P., & Koch, D. (2013). A computational method for generating convex maps using the medial axial transform. In Y. O. Kim, H. T. Park & K. W. Seo (eds). *Proceedings of Paper presented in the Ninth International Space Syntax Symposium* (pp. 64.1–64.11), Seoul, South Korea.

Ostwald, M. J. (2011). The mathematics of spatial configuration: revisiting, revising, and critiquing justified plan graph theory. *Nexus Network Journal, 13*(2), 445–470.

Rosner, V. (2005). *Modernism and the architecture of private life*. New York: Columbia University Press.

Turner, A. (2001a) Angular analysis. *In Proceedings of the 3rd International Space Syntax Symposium* (pp. 30.1–30.11). Atlanta, GA.

Turner, A. (2001b) A program to perform visibility graph analysis. *In Proceedings of the 3rd International Space Syntax Symposium* (pp. 31.1–31.9). Atlanta, GA.

Turner, A., Doxa, M., O'Sullivan, D., & Penn, A. (2001). From isovists to visibility graphs: a methodology for the analysis of architectural space. *Environment and Planning B: Planning and Design, 28*(1), 103–121.

Twombly, R. C. (1975). Saving the Family: Middle Class Attraction to Wright's Prairie House, 1901-1909. *American Quarterly, 27*(1), 57–72.

Varoudis, T. (2015). *depthmapX is a* Multi-platform spatial network analysis software. From https://varoudis.github.io/depthmapX/. Accessed 14 Mar 2015.

Viraph [Computer software] (2017). From https://sourceforge.net/projects/viraph/. Accessed 31 Aug 2017.

Volz, C. M. (1992). The modern look of early twentieth-century house. In J. H. Foy & T. H. Schlereth (Eds.), *American Home Life: 1880–1930* (pp. 25–48). Knoxville: University of Tennessee Press.

Whelan, L. B. (2011). *Class, Culture and Suburban Anxieties in the Victorian Era*. New York: Routledge.

Wright, F. L. (1960). *Frank Lloyd Wright: writings and buildings*. Cleveland: World Publishing Co..