Towards Quantifying Multiple View Layouts in Visualisation as Seen from Research Publications

Al-Maneea, Hayder Mahdi Abdullah; Roberts, Jonathan C.

Published: 01/10/2019

Publisher's PDF, also known as Version of record

Towards Quantifying Multiple View Layouts in Visualisation as Seen from Research Publications

Al-maneea, Hayder M.; Roberts, Jonathan. C.

IEEE VIS 2019 Conference: InfoVis

DOI:

Published: 01/09/2019

Peer reviewed version

Al-maneea, Hayder M.; Roberts, Jonathan. C., "Towards Quantifying Multiple View Layouts in Visualisation as Seen from Research Publications," IEEE VIS 2019 Conference: InfoVis
http://

Abstract:
We present initial results of a quantitative analysis of how developers layout the visualisations in their multiple view systems. Many devel-opers create multiple view systems and the technique is commonly used by the visualisation community. Each visualisation shows data in a different way, and often user interaction is coordinated between the views. But it is not always clear to know how many views a developer should use, or what would be the best layout. We extract images of visualisation tools, across TVCG journal, conference, posters and workshop papers 2012-2018 to analyse the quantity and layout of the views in these visualisation systems. Focusing on view juxtaposition, we code the layout of 491 images and analyse view topology in juxtaposed views. Our analysis acts as a starting point to help designers create better visualisations, acts as a taxonomy of visualisation layouts, and provides a quantitative analysis of how many views developers have used in their visualisation systems.

Definitive Version at:
URL:

@article{Al-maneea-etal-IEEE VIS Conference-2019,
author = {Al-maneea, Hayder M.; Roberts, Jonathan. C.},
Conference = {IEEE VIS 2019 Conference-2019,
title = { Towards Quantifying Multiple View Layouts in Visualisation as Seen from Research Publications},
year = {2019},
month = Oct,
url = {},
DOI ={}
}

Towards Quantifying Multiple View Layouts in Visualisation as Seen from Research Publications

Hayder M. Al-maneea*
Bangor University and University of Basrah

Jonathan C. Roberts†, *Member, IEEE*
Bangor University

Abstract

We present initial results of a quantitative analysis of how developers layout the visualisations in their multiple view systems. Many developers create multiple view systems and the technique is commonly used by the visualisation community. Each visualisation shows data in a different way, and often user interaction is coordinated between the views. But it is not always clear to know how many views a developer should use, or what would be the best layout. We extract images of visualisation tools, across TVCG journal, conference, posters and workshop papers 2012-2018 to analyse the quantity and layout of the views in these visualisation systems. Focusing on view juxtaposition, we code the layout of 491 images and analyse view topology in juxtaposed views. Our analysis acts as a starting point to help designers create better visualisations, acts as a taxonomy of visualisation layouts, and provides a quantitative analysis of how many views developers have used in their visualisation systems.

Keywords: Information visualization, multiple view layouts

1 Introduction

Multiple view systems are often used by visualisation developers. But it is not easy for a developer to know how to layout and position the views in their systems, or how many views they should use, or what design attributes work best. We believe that developers and learners should have guidelines and frameworks to help them make good design decisions. Subsequently, we are keen to develop theories for visualisation, and specifically develop guidelines on best practices of view layout. But to achieve these goals, researchers need to perform basic research to understand best practices of what we currently do. The results of this paper helps us move forward towards our goals.

We present initial results of a quantitative analysis of the quantity of views used in *multiple view* systems as reported in the visualisation literature. In this paper we focus on two questions: (Q1) how many views, and (Q2) what are their arrangements? To answer these questions, we **(1) prepare and extract images** from papers, **(2) code and classify each layout** through visual inspection and discussion, we considered each visualisation in turn, judging the topological makeup of each visualisation, coded them such that we can classify them, and recorded a sketch of their topology and **(3) analyse the results.** Our three-stage methodology is shown in Figure 1, and we use this process to structure the rest of the paper. First the related work in Section 2. Second, we describe how we collected the images for our analysis (Section 3). Third, present how we codify the layouts (Section 4) and fourth we present the results for each question (Q1 in Section 5 and Q2 in Section 6) and discuss how we organised the sketches on a tabletop. Finally we discuss the results and their application, and conclude.

We extract images from research publications that were published at the IEEE Visualisation conference between 2012 and 2018. This seven year period provides a convenient and reproducible set of images of modern visualisation tools that have been designed and presented by community experts. In particular, because these works have gone through peer review, we assume that the authors have spent careful thought over how they present their tools, and consequently they have been attentive to the selection of their views and the presentation of their multiple view systems. We considered many sources, including using a general Internet search for visualisation images, video sources such as Vimeo or YouTube, or other online image repositories. Adding these sources might give us a rich data set of different images and this is certainly a limitation of what we did here. But, they also bring challenges, where image searches change over time, results change per user or geographic location, which would make it more difficult for others to confirm our studies, and to add more images for future years.

In this work we concentrate on view juxtaposition, where each views sits alongside each other, and on the topology of each design layout (e.g., a 2-view system can have one view above another, or left/right of each other). This short-paper extends our poster paper, that was presented at the 2018 IEEE Visualisation conference, where we introduced our methodology and highlighted initial results [1]. The feedback and discussions with the community at this event was invaluable; consequently we extended our quantitative analysis, widened our input data to include 2018 papers and included statistics on single-view systems. There are many research questions that we have investigated in our broader research project, including the quality of views, layout organisation, symmetry and design attributes, to the type of visualisation used. With space limitations of this format, therefore we summarise our main findings and focus on two main questions: Q1 How many views are used in multiple view systems? and Q2 What layout arrangements are popular in multiple view systems?

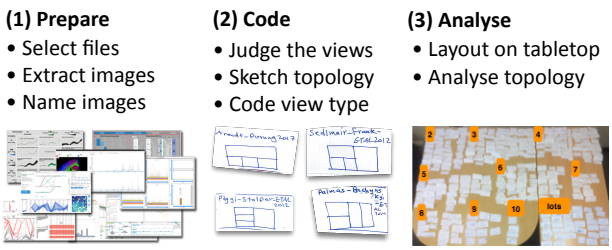


Figure 1: (1) We extracted 491 images from IEEE VIS 2012–2018 conference publication. (2) We coded the images by their topology (making sketches of the layout, totalling 22 sheets of paper); discussing cases to confirm their layouts. (3) We cut the 22 sheets of sketches into individual tiles, and organised them on a tabletop, to analyse and tally the quantities.

2 Related Work

We acknowledge the huge amount of well-cited research that has been achieved in the area of multiple views. Twenty five papers from the conference on Coordinated and Multiple Views conferences (2003 to 2007), rules and principles for the use of multiple views [24], state of the art in Coordinated Multiple Views (CMV) [20], juxtaposition, superposition and explicit designs for multiple view sys-

*h.m.almaneea@bangor.ac.uk
†j.c.roberts@bangor.ac.uk

tems [8,9], keeping multiple views consistent [17], coordinating multiple views across large displays devices [12], and the many systems that create and use multiple coordinated views (e.g., ComVis [13], Snap-together [15], Waltz [19], Improvise [27], Jigsaw [22]), and the recent published work on the phraseology of multiple-views [21]. We acknowledge the huge quantity of research in the area of coordinated multiple view systems; where user interaction in one view is linked to another view (such as linked highlighting). It is through this linking that a user can better explore and discover interesting facts about the data. In this regard researchers have created rudiments of coordination [3], researched linked highlighting and linked navigation [15] and linked brushing [2]. In particular, developers have created many types of brushing including: *compound brushing* [5], *multiple brushes* [25], and complex filtering operations such as through *angular brushing* [10] or *cross-filtered views* [27]. But to date, there has been no systematic study investigating view-layout strategies for multiple views.

3 Preparation and Image Selection

To study the layout of multiple-view tools we needed a set of images to judge. Our goal was to capture images that demonstrate a visualisation tool or technique, that clearly originated from applications (whether on a desktop or website), that were created by through a snapshot/screen-grab operation or directly output from the tool.

Our **preparation process** had four stages. We started by considering all papers presented at the IEEE Visualisation conference between 2012 and 2018. But this creates a very large corpus of information that we decided to reduce the quantity, as follows. (1) We removed all files of supplementary materials. (2) We removed papers that did not have visualisations. (3) We removed papers that only had images that were clearly put-together or had been edited (by an image processing tool). We took this decision because it is difficult to quantify how much editing had been achieved by the authors. (4) We kept files that had at least one candidate image.

We considered how we would code the images, whether manual or automatic encoding. We decided to use a manual coding process. We could have coded the layouts using an image processing algorithm, but decided that we would start with the manual encoding process. Therefore, we extracted the images from the files, and stored them in year-based folders, and labelled each image with a unique abbreviation (that we also use in LaTeXto cite the paper), as follows: $Author1[-Author2[-Author3][-ETAL]]Year.png$. This meant that we could easily reference the image, and locate the associated publication. If there were several suitable images which were different then we collected both, and added a F (for Figure) followed by the number, to the file name ($-F1.png$, $-F2.png$, etc.).

4 Coding the Layouts

To address our two questions (Q1 and Q2) we needed to first be able to **identify** individual facets of a multiple-view display, and then **code** them effectively. In this section we define our view identification and coding strategies.

4.1 View identification

It is not necessarily easy to identify, or count, views on a visualisation. Sometimes it is clear, that there are separate dividing parts to the view, that (say) one visualisation is a scatterplot and another a bar chart. But other times it is less clear how many sub component parts the visualisation contains. Designers overlay visualisations, or even place visualisations in an irregular way which can complicate deciphering the layout. But, because developers want users to understand their data display, they are deliberate in how they position their views. In addition, they will engineer the display such that the graphical marks standout from the background, and contain clearly perceptible parts. In perceptual terms, this is *figure and ground* [26]. The graphical marks that encode the data are the *figure*, which stands out from the background colour.

Most visualisation tools have interface components, and may include menus, buttons, slider bars, legends, colourmaps, legends, etc. Sometimes these are integrated with a view, sometimes they are shown in their own window. In most cases it is possible to ignore these menus. However sometimes the menus take up a significant space. We could code these views as "menu", however this would not allow us to be consistent in our coding, because other views have this interaction/menu integrated into the views. We could code them as "null" views (or information panels), but again this may skew the results. Consequently, we make a multi-criteria solution. If the menu is on the side, or along the top, and can be easily ignored without changing the topology of the view layout we ignore it! If a menu-window is enclosed between other views then we treat this as "null" space and merge it into the closest neighbouring view. This allows us to treat all menus as part of (at least) one view, and every visualisation is treated consistently.

To perform meaningful and consistent manual coding of view quantity we needed to develop a set of rules (that we name the **view identification** process). We identified five rules to determine 'what is a view': (1) **Views are usually visually separate from another view.** Count the views that are clearly separated by spacing, a gap that is coloured in the background colour, rendered in a rectangle, or placed within a window. (2) **Views have different tasks**. Count the views separately if they afford a different task. (3) **If we can name them, we have different views.** For example, you could say "scatterplot, line graph and bar chart" and you would have three views. (4) **Consider how a programmer would code it.** If they cannot be separated visually, they may be able to be separated functionally. This is separation of concerns at the *functional* level, e.g., `draw.scatterplot()`. (5) **Ignore interface components**. Ignore menu windows if it is sensible to do so (such as a menu along the top of all views). We printed these rules, and kept them close, and especially used them when we had a dilemma of how to judge the layout.

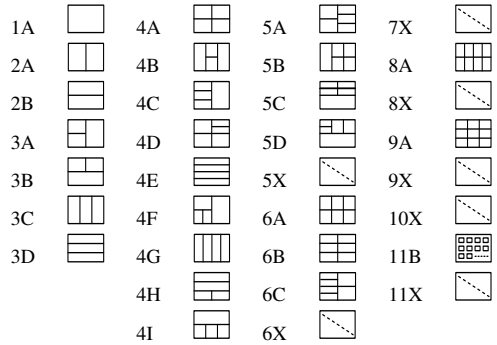
4.2 Coding the layout arrangements

Now, being able to identify a "view" we needed a way to record the views and their configuration. We chose an inductive strategy to develop codes to answer Q1 and Q2. We developed these codes through refinement and critical thought. We chose a visual coding scheme where we displayed each image, made a simple sketch of the layout topology and labelled the sketch with the file name. If we had a dilemma on how to sketch the topology we drew all possible arrangements, which were later discussed. This meant that the 'codes' were developed as part of the judging process. Our **coding-layout process** had eight stages: (1) Every image was displayed on a computer screen. (2) Judge topology (using the view rules in view-identification process). (3) Code the layout in sketches, write file name alongside. (4) With dilemmas, sketch all possible layout configurations. (5) Discuss each dilemma, agree on one topology, and keep agreed topology. (6) Cut the sketches into individual *tiles*. (7) Arrange tiles on tabletop, discuss ideas, and categorise layouts. (8) Record quantities.

When we evaluated all 491 images, we generated over 22 sheets of paper (with on average over 22 sketches per page). We discussed 124 cases and agreed their topological structure, and randomly checked another 10 to make sure we agreed with a selection of the other cases. The agreed sketched images were then cut up into individual *tiles*, keeping only the agreed topologies.

There are different potential ways to name the view layouts. We could use real names, such as "one view", "dual view" or "three view", but decided that this would not make a convenient shorthand version, because ordering of them would be difficult. Our solution was to label them with a number (the view quantity) followed by

Table 1: Numlacature for the layouts. Those that are not similar to others are binned together into the category X.



a letter (indexing a view layout). For example, "1A" is a one view (there is only one possible layout), and "2A" is a vertical 2-view layout, while "2B" the horizontal layout. While the number is logical, it is unclear how to allocate letters to view configurations. Our solution is arbitrary. It was decided on a first-come basis. When counting the views, the first new type was allocated an A, the next unique type a B, and so on. Our naming scheme is listed in Table 1. Additionally, when the views quantities become large, there are many more topological arrangements. We could name every configuration, but decided that it would not be useful for our analysis, therefore we named these as being irregular and gave them the letter "X".

5 Results and Discussion: Q1, How Many Views?

We used a tabletop strategy to confirm the quantity of layouts. We arranged the cut tiles on the table organised by (first) the quantity of views, and then by their topological arrangement. We did this by arranging the cut tiles into groups, organised by their quantity; shown in Figure 1(right). By physically moving these tiles, it helped us better understand the frequency of each layout. We grouped tiles together by quantity of views, and then by their topological structure. E.g., putting all dual-views, three-views, four-views together, and so on. This tabletop view gives a physical area chart of the quantities in each strategy and from this tabletop collection of tiles we were able to quickly record the quantities in a spreadsheet for further analysis. We recorded the quantity of views used per year, and tallied each of the different view topologies up to 20 views. We decided that the space allocated to individual views, when the view quantity exceeds 20, views is very small, and therefore we bin all this information into a bin 20+. In the Tabletop view (Figure 1) we put them together under a label of "lots". We tallied the scores and charted the data, shown in Figure 2.

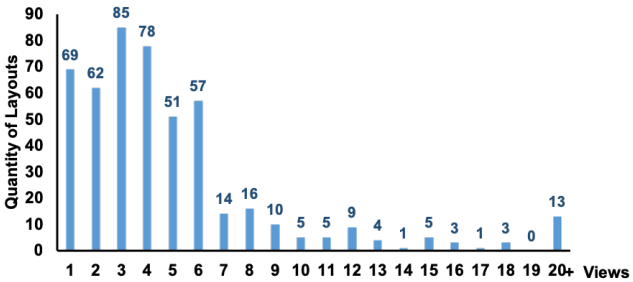


Figure 2: Histogram showing the frequency distribution of the views. From 491 multiple view systems, in our study we find that a 3-view system is most frequent.

From our analysis, we are now able to start to answer Q1: "how many views do people use", and because there are different ways to interpret the statistics, we break this question into several further questions.

What is the most common layout? To answer this question we rank the view quantities (Figure 2). The most common layout is a 3-view system. Four-view systems are next, followed by one-view systems and dual-views. Six view systems are slightly more frequent than five view systems. There is a clear division between 6-view and 7-view (and more) systems. In fact 84.68% of the systems are 6-view or less. These results are important. They say that the majority of developers use 6-views or less, and most of them choose a 3-view system. The also suggest that (in general) fewer views are used more often, which supports the rule of parsimony by Baldonado et al. [24].

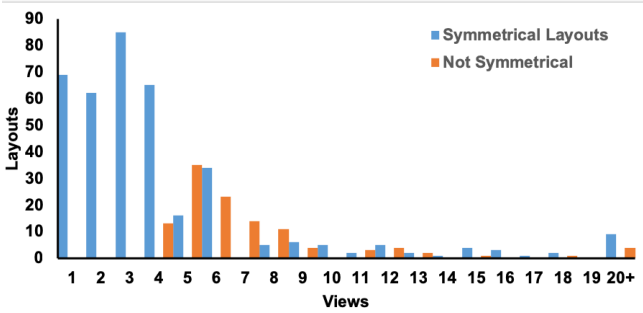


Figure 3: Histogram of symmetrical versus non-symmetrical views.

What is the average quantity of views used? The naive arithmetic mean calculates to 4.9, but this is misleading. We have a positive skew in the distribution of the view count (skew is 1.049), and it is clear from Figure 2 that we have a very long tail. Such a positive skew is understandable; when we count views it is impossible to get a value less than a 1-view system, and it is far less likely to see systems with huge quantities of views (it is just impractical to have a system with hundreds of views). We can demonstrate this situation by modelling a normal distribution from 1 to 20, with an average of 3 (as per the most frequent occurrence), and comparing our observations with this model. In fact we get statistically similar results to our coding observations: a Pearson correlation is calculated as 0.960 with a ttest $p(0.885)$.

6 Results and Discussion: Q2, View Arrangements

From our tabletop layout of the tiles, we were able to examine the view layouts. We exchanged the tile sketches into the nomenclature, as explained in Section 4.2 and shown in Table 1. Results for the top ten views are shown in Table 2. We have complete data for all view counts up to 20, but only show 10 views due to space limitations in this paper. The results, provide even more fine grain detail. To calculate these results we performed some vertical aggregation. Consequently if the topology was the same on the left (such as) and to the right (such as) we chose the same label, in this case 3A. While we miss out on calculating if the views are more left biased or right, it is a pragmatic decision that allows us to simplify the tallying.

Analysing the layouts in fine detail is interesting but challenging. There are many layouts, and as the quantity of views increase so does the number of arrangements. From Table 2 we notice the higher ranking layouts such as 2A , 3A , 3B . It is interesting that the popular layouts are not necessarily those with fewer views, for instance 5X and 6X (two irregular layouts with little structure) are within the top ten. Layout 6A is a small gridded layout and features also with the top ten. We also see that there are many familiar structures. 44 views show side-by-side views, and a further 18 have a two-way split (top to bottom). Four layouts have 3-views, and nine layout strategies with 4-views. We notices a two-thirds design strategy being prominent, and more than half of the views have a significant left/right division somewhere in their strategy. The others follow a 3-way split.

We believe that designers are following principles of *balance* in their design decisions. Symmetrical balance encourages an equal

Table 2: Results of tallying the specific layouts, per years. We show data for the top 10 view strategies. Where f is frequency, $\%Vf$ is percentage frequency of that View type, and $\%f$ is percentage overall. we order the rows by their overall rank.

Layout		Y12	Y13	Y14	Y15	Y16	Y17	Y18	f	$\%Vf$	$\%f$		Rnk
1A		10	7	12	12	9	8	11	69	100.00	14.05		1
2A		9	4	3	5	4	10	9	44	70.96	8.96		2
3A		1	3	3	3	8	7	9	34	40.00	6.92		3
3B		6	3	2	4	4	4	6	29	34.11	5.90		4
4A		4	3	1	5	4	2	5	24	30.76	4.88		5
6X		6	3	0	2	4	3	5	23	40.35	4.68		6
5X		3	2	2	3	1	5	5	21	41.17	4.27		7
2B		1	2	4	4	1	1	5	18	29.03	3.66		8
6A		1	2	1	4	2	3	5	18	31.57	3.66		8
3C		0	2	1	2	1	5	3	14	16.47	2.85		10
7X		1	1	1	3	3	3	2	14	100.00	2.85		10
5A		2	0	1	1	3	2	2	11	21.56	2.24		12
8X		3	1	1	3	2	1	0	11	68.75	2.24		12
4B		2	0	3	1	3	1	0	10	12.82	2.03		14
4C		2	0	2	0	1	1	4	10	12.82	2.03		14
3D		1	3	0	1	1	0	2	8	9.41	1.62		16
5B		1	1	2	3	0	1	0	8	15.68	1.62		16
6B		1	1	1	1	0	2	2	8	14.03	1.62		16
6C		0	1	1	2	2	0	2	8	14.03	1.62		16
4D		2	2	0	0	0	3	0	7	8.97	1.42		20
4E		0	0	2	2	1	0	1	6	7.69	1.22		21
4F		2	1	1	0	0	1	1	6	7.69	1.22		21
5D		0	0	0	1	0	2	3	6	11.76	1.22		21
9A		2	1	0	1	0	2	0	6	60.00	1.22		21
4G		0	2	0	1	1	1	0	5	6.41	1.01		26
4H		0	0	2	2	0	0	1	5	6.41	1.01		26
4I		0	0	0	2	0	1	2	5	6.41	1.01		26
5C		1	1	1	1	0	0	1	5	9.80	1.01		26
8A		0	0	2	0	1	2	0	5	31.25	1.01		26
10X		1	0	2	0	0	1	1	5	100.00	1.01		26

weight of both sides. We propose that more designs are symmetrical than not. To explore this hypothesis we plot the quantity of symmetrical views (whether vertical or horizontal symmetry) against the non symmetrical. We do observe that there may be a trend to utilise more symmetrical layout strategies. But our results are not conclusive, because we also see that there is a natural tendency for layouts with more views to be less symmetrical. And it is difficult to tease apart these two observations. Another observation is that the long thin views (such as 3D, 4G, 4H) are lower in the rankings. These views seem to be used for timelines, or line graphs, and maybe these types of visualisations are less popular in higher quantities.

7 Discussion and Conclusions

We have comprehensively examined views as presented in the visualisation literature, and have started to answer our two questions: quantity of views and view layouts. We have also presented strategies to identify views, and to count and code layouts. This work helps the community move closer to developing a theory of visualisation, and to provide guidelines on visualisation phraseology and use. Certainly our in-depth evaluation and presentation of results will give quantitative data to researchers to develop new sets of guidelines over layout strategies. Subsequently, we summarise our findings in four headlines:

1. More authors present systems with fewer views. The most frequent view was a 3-view system.
2. Authors prefer symmetrical over non-symmetrical layouts.
3. Dual view and four-view layouts are popular, see Table 2.
4. 84.68% of the layouts are 6-view or less.

Developers of multiple view systems have used many different layout strategies to position their views on the screen. But there is commonality in design choices, and we have discovered that 3-view layouts are the most popular, and that bar charts and scatter plots are widespread, and that together with line charts, heatmaps and node-link diagrams they form over 50% of all views in multiple view systems. We have discovered that the top eight visualisation layout strategies are 1A, 2A, 3A, 3B, 4A, 6X, 5X, 2B (see Table 2), and that there appears to be preference for symmetrical layouts. We have demonstrated that although there is a preference for fewer views, some developers still display systems with a huge quantity of views (layouts with 7-views and more equals over 15% of the total quantity of layouts). But we are sure that there is still much information hidden in this data that could be extracted. In particular we hope that our results and analysis will restart the debate over how many views are suitable, and will help to focus the minds of the developer as they create multiple view systems, to contemplate how they are laying out their views, and how many views they are using.

One aspect, in particular, that we feel should be debated and further researched, is the question of who has control over the layout? If we look programming languages, we find structures such as the GridBagLayout in Java, to help programmers layout structures. We see panel layouts in web structures (such as top, left, main, right or bottom panel), and templates to help Web developers layout their information. But there has been little research in the best layout strategies for visualisation. Certainly it will be developer of a visualisation system who will determine how much control the user has over the tool, and over the layout of views, and we propose five options for developers:

Developers can predetermine the layout. This is a fixed strategy, and is usually used reserved for fewer views or bespoke systems (designed for a particular purpose or user). For instance, in **two view** systems there is little choice: the views can be laid out left/right or above/below. Note such systems are also known as *side-by-side*, *parallel* or *dual view* [11, 14, 16] systems, or if one view is more important then *primary/secondary*, *focus+context* or *overview and detail* systems.

Views positioned based on data. For instance, Roberts [18] positions the views based on hierarchical tree of data exploration, or the splom layout [4] (lattice charts) positions the small multiples to allow pairwise comparison of scatterplots, and Polaris [23] and spreadsheet visualisation approach by Chi et al. [6] display information in grid-based layouts.

Group views that are coordinated together. For instance, views that sharing linked highlight, or linked navigation can be positioned closely and maybe in the same window. I.e., the type of coordinated manipulation can be used to control the positions of views; Roberts named these "render groups", [18] and Weaver puts them side-by-side and visually connects them with lines and arrows [27].

The screen size can be used to determine the layout; where for instance, small multiples are laid on the screen in order that wraps onto the next line, and as the window size is changed so the viewed quantity changes (in the same way that a responsive/mobile-aware adjusts the content determined by the width).

User can determine the layout. Often systems are created whereby the user can drag (from a toolbox of possible visualisation types) and drop the selection onto a canvas, where the views are snapped to align together (such as with Improvise [27], Jigsaw [22], Vinca [7] and many other tools).

There are many more questions that we could ask, and there is a much further work to be done. E.g., while we have started to investigate the connection between view layouts and the visualisation forms, it would seem sensible, from a design standpoint, that there is a strong correlation between view type and position in the layout. With (for instance) long and thin structures, such as timelines or line graphs, would be placed in long and thin layouts. Additionally, there may be a connection between the view layout strategy and its position of the layout in the article. Where, for instance, visualisations that have more views are placed along the top of the article, with those with less views in a column. We leave these questions to future work.

REFERENCES

[1] H. M. Al-maneea and J. C. Roberts. Study of Multiple View Layout Strategies in Visualisation. In *Posters presented at the IEEE Conference on Visualization (IEEE VIS 2018), Berlin, Germany*, Oct. 2018.

[2] R. A. Becker and W. S. Cleveland. Brushing Scatterplots. *Technometrics*, 29(2):127–142, 1987.

[3] N. Boukhelifa, J. C. Roberts, and P. J. Rodgers. A coordination model for exploratory multiview visualization. In *Proceedings International Conference on Coordinated and Multiple Views in Exploratory Visualization - CMV 2003*, pp. 76–85, July 2003. doi: 10.1109/CMV.2003.1215005

[4] D. B. Carr, R. J. Littlefield, W. L. Nicholson, and J. S. Littlefield. Scatterplot matrix techniques for large n. *Journal of the American Statistical Association*, 82(398):424–436, 1987. doi: 10.1080/01621459.1987.10478445

[5] H. Chen. Compound brushing explained. *Information Visualization*, 3(2):96–108, 2004. doi: 10.1057/palgrave.ivs.9500068

[6] E.-H. Chi, P. Barry, J. Riedl, and J. Konstan. A spreadsheet approach to information visualization. In *Symposium on Information Visualization*, pp. 17–24. IEEE, 1997.

[7] R. George, P. Robbins, A. Davies, and J. Roberts. Interactive visual analytics of coastal oceanographic simulation data. In *Posters presented at IEEE VIS 2012*, October 2012.

[8] M. Gleicher. Considerations for visualizing comparison. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):413–423, Jan 2018. doi: 10.1109/TVCG.2017.2744199

[9] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011. doi: 10.1177/1473871611416549

[10] H. Hauser, F. Ledermann, and H. Doleisch. Angular brushing for extended parallel coordinates. In *Proceedings of the IEEE Symposium on Information Visualization*, pp. 127–130, 2002.

[11] N. Henry and J. d. Fekete. Matrixexplorer: a dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):677–684, Sept 2006. doi: 10.1109/TVCG.2006.160

[12] R. Langner, U. Kister, and R. Dachselt. Multiple coordinated views at large displays for multiple users: Empirical findings on user behavior, movements, and distances. *IEEE Transactions on Visualization and Computer Graphics*, 25:608–618, 1 2019. doi: 10.1109/TVCG.2018.2865235

[13] K. Matkovic, W. Freiler, D. Gracanin, and H. Hauser. Comvis: A coordinated multiple views system for prototyping new visualization technology. In *2008 12th International Conference Information Visualisation*, pp. 215–220, July 2008. doi: 10.1109/IV.2008.87

[14] G. M. Namata, B. Staats, L. Getoor, and B. Shneiderman. A dual-view approach to interactive network visualization. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pp. 939–942. ACM, New York, NY, USA, 2007. doi: 10.1145/1321440.1321580

[15] C. North and B. Shneiderman. Snap-together visualization: A user interface for coordinating visualizations via relational schemata. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pp. 128–135. ACM, New York, NY, USA, 2000. doi: 10.1145/345513.345282

[16] M. Plumlee and C. Ware. Integrating multiple 3d views through frame-of-reference interaction. In *Proceedings International Conference on Coordinated and Multiple Views in Exploratory Visualization - CMV 2003 -*, pp. 34–43, July 2003. doi: 10.1109/CMV.2003.1215001

[17] Z. Qu and J. Hullman. Keeping multiple views consistent: Constraints, validations, and exceptions in visualization authoring. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):468–477, Jan 2018. doi: 10.1109/TVCG.2017.2744198

[18] J. C. Roberts. On encouraging multiple views for visualization. In *Proceedings IEEE Conference on Information Visualization.*, pp. 8–14, Jul 1998. doi: 10.1109/IV.1998.694193

[19] J. C. Roberts. Waltz - an exploratory visualization tool for volume data, using multiform abstract displays. In R. F. Erbacher and A. Pang, eds., *Visual Data Exploration and Analysis V, Proceedings of SPIE*, vol. 3298, pp. 112–122. Bellingham, Washington, USA, 1998. doi: 10.1117/12.309533

[20] J. C. Roberts. State of the art: Coordinated multiple views in exploratory visualization. In *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV 2007)*, pp. 61–71, July 2007. doi: 10.1109/CMV.2007.20

[21] J. C. Roberts, H. M. A. Al-Maneea, P. W. S. Butcher, R. Lew, G. Rees, N. Sharma, and A. Frankenberg-Garcia. Multiple views: different meanings and collocated words. *Computer Graphics Forum*, 3 2019.

[22] J. Stasko, C. Görg, and Z. Liu. Jigsaw: supporting investigative analysis through interactive visualization. *Information visualization*, 7(2):118–132, 2008.

[23] C. Stolte, D. Tang, and P. Hanrahan. Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, Jan 2002. doi: 10.1109/2945.981851

[24] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pp. 110–119. ACM, New York, NY, USA, 2000. doi: 10.1145/345513.345271

[25] M. O. Ward. Creating and manipulating n-dimensional brushes. In *Proceedings of Joint Statistical Meeting*, pp. 6–14, 1997.

[26] C. Ware. *Information Visualization: Perception for Design*. Elsevier (Morgan Kaufmann), Amsterdam, 2012.

[27] C. Weaver. Building highly-coordinated visualizations in improvise. In *IEEE Symposium on Information Visualization*, pp. 159–166, 2004. doi: 10.1109/INFVIS.2004.12