Composition and Configuration Patterns in Multiple-View Visualizations

Xi Chen, Wei Zeng, Yanna Lin, Hayder Mahdi Al-maneea, Jonathan Roberts, and Remco Chang

Abstract— Multiple-view visualization (MV) is a layout design technique often employed to help users see a large number of data attributes and values in a single cohesive representation. Because of its generalizability, the MV design has been widely adopted by the visualization community to help users examine and interact with large, complex, and high-dimensional data. However, although ubiquitous, there has been little work to categorize and analyze MVs in order to better understand its design space. As a result, there has been little to no guideline in how to use the MV design effectively. In this paper, we present an in-depth study of how MVs are designed in practice. We focus on two fundamental measures of multiple-view patterns: *composition*, which quantifies what view types and how many are there; and *configuration*, which characterizes spatial arrangement of view layouts in the display space. We build a new dataset containing 360 images of MVs collected from IEEE VIS, EuroVis, and PacificVis publications 2011 to 2019, and make fine-grained annotations of view types and layouts for these visualization images. From this data we conduct composition and configuration analyses using quantitative metrics of term frequency and layout topology. We identify common practices around MVs, including relationship of view types, popular view layouts, and correlation between view types and layouts. We combine the findings into a MV recommendation system, providing interactive tools to explore the design space, and support example-based design.

Index Terms—Multiple views, design pattern, quantitative analysis, example-based design

1 Introduction

We present an in-depth study on how multiple views are used in practice, and integrate our results into a recommendation system for the layout design of a multiple-view visualization. Traditional visualization designs aim to maximize the utility of the visualization for specific data types or tasks. For example, line graphs show temporal information, maps display geographical information, etc. Multiple-view visualization (denoted as MV) is a technique that seeks to integrate these visualizations by compositing multiple views of different view types into a single cohesive representation [22, 39]. Since each visualization conveys a specific perspective of data, a well-designed MV enables the user to simultaneously see representations of the same data from different perspectives. In fact, the power of multiple views is well understood and the technique has nowadays become ubiquitous in exploratory data visualization [40].

However, despite the ubiquity of multiple views in visualization systems, there are few guidelines, and those that do exist are very general. For instance, researchers advise developers to use multiple views sparingly [5], and adopt consistent visual encodings across multiple views [36]. Additionally, while researchers have made recommendations for multiple displays [13], and collaborative tasks over large displays [27], the plethora of design considerations pose challenges to developers in practice.

The visualization community has developed many visualization authoring tools, such as Power BI [1], Tableau [3], and Spotfire [2]. These tools allow the user to quickly design prototypes of MVs using a set of predetermined templates for common data types, such as the sales dashboard templates offered by Tableau [3]. However, for more complex data and tasks, designers often still need to manually curate the layout of MVs through trial and error. This process can be tedious and time consuming, and sometimes produces results that fail to meet design guidelines [36]. Recently, researchers have developed techniques to automatically distribute multiple views in a visual space [18, 23, 26, 41]. While the layout of theses systems may appear arbitrary, users place the views side-by-side in a deliberate way.

The goal of this paper is to create an image corpus of real-world MV images, analyze patterns contained in this data, distill a set of guidelines, and finally to produce a recommendation system for the design of MVs. To code the design patterns of MVs, we first code each of the views in a MV in terms of its:

- *view type*: the mapping from data to visual form, *i.e.*, the result of a visualization technique (*e.g.*, bar and line charts);
- *bounding box*: position and size in the physical display space (most often in 2D) where the view is presented.

After each view is coded in terms of its *type* and *bounding box*, we then encode the overall MV design based on its:

- *Configuration*, including position and size of the bounding box of each view in the physical display space.
- *Composition*, including frequency, diversity, and correlations of view type usages.

Using this coding scheme, we collect and label images of MV designs from publications in *IEEE VIS*, *EuroVis*, and *IEEE PacificVis* conferences from 2011 - 2019 (Section 4.1). The result is a curated dataset of 360 MV designs, which are then manually coded using an annotation tool that we developed for this effort (see Section 4.2).

We perform in-depth analyses on this dataset, using a number of quantitative metrics from information theory and graphics, such as conditional probability and layout topology. The analyses reveal interesting composition and configuration patterns of MV design, including frequencies, aspect ratios, and positions of different view types (Section. 5). For example, aspect ratios of most view types are within [1/2, 2], except for some types like *Area* and *Panel* (see Figure 7).

Lastly, using the found composition and configuration patterns from the analyses, we develop an interactive recommendation system for designing MVs. In particular, this system: (1) enables multi-faceted exploration of existing MV designs (Section 6.1), and (2) recommends designs given view type and layout information (Section 6.2). We evaluate the utility and effectiveness of this recommendation via a formal user study. The results of the study suggest that our recommendation

- *Xi Chen, Wei Zeng, and Yanna Lin are with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. Xi Chen is also with University of Chinese Academy of Sciences. Wei Zeng is the corresponding author. E-mail: {xi.chen2, wei.zeng, yn.lin}@siat.ac.cn.*
- *Hayder M. Al-maneea and Jonathan C. Roberts are with Bangor University. E-mail: {h.m.almaneea, j.c.roberts}@bangor.ac.uk. Hayder M. Al-maneea is also with University of Basrah.*
- *Remco Chang is with Tufts University. E-mail: remco@cs.tufts.edu*

Manuscript received 30 Apr. 2020; revised 31 July 2020; accepted 14 Aug. 2020. Date of publication 13 Oct. 2020; date of current version 15 Jan. 2021. Digital Object Identifier no. 10.1109/TVCG.2020.3030338

tool can enhance the understanding of MV design space, and promote appropriate MV designs, for both visualization novices and experts.

In summary, we contribute to the MV design in the following way:

- We curate a new dataset of 360 MV designs. Using an annotation tool developed for this project, each of the MV designs has been carefully labeled and annotated. The resulting dataset will be released publicly to foster future research.
- We conduct a series of quantitative analyses on the dataset, and find common composition patterns of view type usage and configuration patterns of layout arrangement in the design of MVs.
- We develop an interactive recommendation system that supports multi-faceted exploration, and recommendations of MV designs. We conduct a formal user study showing the effectiveness of the recommendation system. The system is freely available for the academic purpose at `https://mvlandscape.bitbucket.io/`.

2 Related Work

Multiple Views. Card et al.'s reference model [10] states that visualizations are created in four steps: i) processing *raw data* into *data tables*; ii) mapping *data tables* to *visual structures*; iii) transforming *visual structures* to *views* through operations like zooming and brushing; and iv) rendering and displaying *views* to users. While the reference model is useful for designing a single visualization, it does not provide guidelines for designing visualizations for more complex and high-dimensional data. To help users examine and interact with large, complex, and high-dimensional data, multiple-view visualizations (MV) that can show different perspectives of data emerged [38]. For example, dashboards evolve from single- to multiple-view visualizations, rendering an increase of data visibility, enhancement of operational efficiency, and reduction of understanding cost [43]. The visualization community has contributed to MV design from various perspectives, *e.g.*, suggesting rules and guidelines [5, 36], developing authoring tools (*e.g.*, Polaris [47], Improvise [52], and ComVis [33]), and extending to mobile devices [18, 26, 41] and large displays [27].

Many theories have also been proposed to facilitate the understanding of the relationship between views. For example, VisLink [15] formalized multi-relation visualizations as side by side, in parallel, or in chosen placements. Javed and Elmqvist [22] and Gleicher et al. [17] categorized design space of composite visualizations into juxtaposition, superimposition, overloading, nesting, and integration. However, though much progress has been made, the design of the layout of a multiple views visualization is usually curated manually based on the designers' experience. This process can be difficult and laborious for professional designers, not to mention visualization novices.

The difficulty in designing MVs suggests a lack of structure and understanding of the design space of MVs. However, the view layouts are still created by humans, which suggests that MV design is not arbitrary [18]. This work helps to address this challenge, by performing an empirical study on how MVs are designed in practice, which we categorize as *composition* and *configuration* patterns. The goal of this study is to further our understanding of the design space of MVs and provide the foundation for data-driven MV design.

Data-driven visualization design. Mackinlay proposed APT (A Presentation Tool) [31] for automated visualization design based on the expressiveness and effectiveness of the visualization. APT builds on studies in graphical perception, *e.g.*, rankings of visual variables by data type [14] and analytical tasks [11]. Some of these findings have been integrated into the development of visualization authoring tools, *e.g.*, ShowMe [32].Following Mackinlay's work, there has been an increasing trend of using data-driven models for automated visualization design [42]. Most of these studies aim to learn an optimal mapping from inputs of data attributes and tasks to outputs of visualizations. For example, SEEDB [50] recommends visualizations that it deems useful or interesting based on the perceived utility of the visualization; Data2Vis [16] learns an end-to-end model for automatic visualization generation; DeepEye [30] and VizML [19] learn to rank visualization for input specifications of data, tasks, and context; Draco [34] learns soft constraints for visualization design; and Chen et al. [12] learns

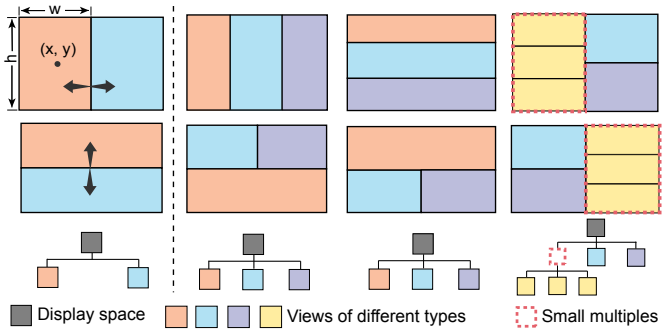


Fig. 1. Illustration of some example MV layouts. Tree structures below depict corresponding hierarchies of the views.

global and local features for timeline infographics. While useful, these works focus on mapping *data tables* to *visual structures* for a single visualization. The goal of this paper is to provide the foundation for designing the composition and configuration of these visualizations into a multiple-view design.

In the simplest form of a MV design, individual visualizations can be arranged in a grid layout, as shown in VizDeck [25]. However, as we show in Section 5, real-world composition and configuration patterns of MVs can be more complicated and divergent. Due the complexity of the design space, we develop a recommendation system to help a designer generate a MV design. Based on the analyses of the MVs in the VIS community over the last nine years, our recommendation system can propose useful and more nuanced MV designs that are more suited for real-world applications.

Layout Design. MV design can be regarded as a layout design problem studied in many fields, such as graphics design (*e.g.*, [35, 55]), architecture (*e.g.*, [53, 56]), and treemap (*e.g.*, [6, 44]). Here we briefly summarize closely related works in graphics design and treemap.

Many works in graphic layout design largely rely on rule-based approaches based on existing design principles. For example, Xu et al. [55] introduced the beautification metric for the global layout that aligns sketch-based interfaces. However, the design principles typically aim to optimize certain properties, which may ignore the resulting effect on other aspects of the design. As a result, in recent years researchers have begun to explore an exemplar-based approach by learning from existing designs. For example, O'Donovan et al. [35] optimized the arrangement of the input contents of a single-page infographic layout based on a small number of example layouts. Zheng et al. [59] showed that infographic layout can be synthesized using a deep generative model learned from a large-scale magazine layout dataset.

Related to the layout problem in graphic design, treemap is a nested enclosure visualization, which is often described as space-filling [44]. The design of treemaps (e.g. squarified treemap [9]) therefore shares similar considerations as the design of MVs. In particular, an effective MV design should also take into account the efficiency of the usage of space, which can be evaluated by quantitative metrics proposed for measuring the space efficiency of treemaps. Borrowing from literature in treemap design, we adopt a series of metrics, including *aspect ratio* [6], and *stability* and *relative-position-change* [46] when analyzing the configuration pattern of MVs.

3 Definition and View Specification

We define a multiple view (MV) as a layout that arranges two or more views in a display space (see Figure 1 for some example layouts). Each view in a MV consists of at least two perspective attributes:

- *View type* (denoted as *type*): Following Card et al.'s model [10], a view is formed by applying transforming operations to the visual structure. Many visual structures have been designed, which can be classified into various view types [28, 45]. Recently, Borkin et al. [7] created a taxonomy of 12 chart types for information visualization, including *Bar*, *Line*, and *Circle*, etc. We adopt the taxonomy in this study. In addition, we refer to scientific visualizations, including volume and flow visualization, collectively as *SciVis*. We treat graphic

widgets, including menus, legends, and narrative texts that are not overlaid on top of other views, as *Panel*. In practice, many widgets are arranged side-by-side, in which cases we treat them as one panel. Sometimes a MV arranges several menus on the periphery, forming a very narrow region that makes a marginal effect on the view layout. We ignore these menus following the convention established in [4].

In summary, in this work, we consider 14 views types (12 types of information visualizations [7] + SciVis + Panel). The use of the view types defines the *composition* pattern of a MV. The view types, together with their abbreviations and icon representations are presented in Supplementary Table S1, and examples of each view type are presented in Supplementary Table S2.

- *Bounding box* (denoted as *bbox*) specifies the center position (x, y) and size $w \times h$ of a view in the display space S. Multiple boxes fill up the display space, *i.e.*, no gaps or overlaps between two neighboring views. The display space S can fall in a wide range from small-size tablets, to medium-size desktop monitors, to large-size video walls. However, MVs studied in this work are collected from publication figures that do not state exact display size. Thus we adopt normalized parameters for *bbox*, *i.e.*, x, y, w, h are scaled to the range of [0, 1].

  The arrangement of multiple *bbox*es in the display space formalizes the view layout, *i.e.*, *configuration* pattern of a MV.

In practice, many MVs juxtapose several views of the same visual type together. This is referred to as "small multiples" by Tufte [49]. To distinguish small multiples from other MV designs, we further incorporate the concept of hierarchy: the display space S is regarded as the root node; views and small multiples filling up S are level-1 nodes; views forming level-1 small multiples are level-2 nodes; and so on. Although the process can repeat multiple times, we find that views in our corpora are at most level-2. Figure 1 illustrates some example layouts, and the tree structures below depict the view hierarchies. MVs on the right side are made up of five views, three of which (in yellow color) form a small multiples.

Given the constraint of a 2-level hierarchy, we can define a MV as:

$$MV := \{view_1, [view_2, view_3], \cdots, view_n\}$$

where $view_i$ is represented as a two-tuple $view_i := (type_i, bbox_i)$, and $[\cdots]$ denotes a small multiples that contains several level-2 views.

4 Dataset Construction

4.1 Multiple-View Visualization Collection

Table 1. Number of MVs collected in IEEE VIS (VAST, InfoVis, SciVis), EuroVis, and IEEE PacificVis from 2011 to 2019.

Venue		'11	'12	'13	'14	'15	'16	'17	'18	'19	Total	
IEEE VIS	VAST	18	13	18	34	25	21	25	26	17	197	273
	InfoVis	4	7	4	7	7	4	4	6	6	49	
	SciVis	0	2	2	3	7	3	4	3	3	27	
EuroVis		7	5	2	3	1	6	5	4	14	47	
IEEE PacificVis		3	1	2	3	3	6	5	7	10	40	
Total		32	28	28	50	43	40	43	46	50	360	

To ensure design diversity and quality, we select MVs from publications in *IEEE VIS*, *EuroVis*, and *IEEE PacificVis* from 2011 - 2019. The MVs are initially recorded as images, which are collected through the following steps:

- First, we crawl the publications by referring to their digital object identifiers (DOIs), using the information collected in the dataset [21] for *IEEE VIS* publications, and from the DBLP database for the other publications. The process produces a total of 1,976 publications, including 1,149 from *IEEE VIS*, 475 from *EuroVis*, and 352 from *IEEE PacificVis*.
- Next, we use a combination of several data preparation and image processing techniques to automatically extract figures from the papers: (1) We employ *PyMuPDF*[1] and *pdftohtml*[2] to convert *pdf* papers to *jpg* and *xml* files, respectively; (2) By querying the keywords of *Fig.* or *Figure* in the *xml* file, we can locate positions of all figures in one paper; and (3) We crop all figures from the *jpg* file based on the figure positions. In this way, we extract 16,891 figures from 1,976 papers.
- Lastly, we manually choose MVs made up of two or more views, by filtering out figures that fall into one of the following conditions: (1) figures of system interfaces, *e.g.*, architecture diagrams; (2) figures that only include a single view visualization; and (3) figures that contain multiple interfaces. These conditions filter out most figures, yet there still exist some figures that are difficult to verify from the image. In such cases, we check carefully the figure captions and corresponding text descriptions in the papers. Finally, if there is more than one MV figure in a paper, we choose the first one that appears in the paper.

In this way, we identify a total of 360 MV images. Table 1 presents the number of MVs collected in each conference per year. Most MVs are collected from *IEEE VIS* since there are many more papers in *IEEE VIS* than *EuroVis* and *PacificVis*. In detail, the table shows that *IEEE VAST* contributes the most (197/360) since systems in VAST typically utilize the MV design to present complex datasets. It is interesting to notice that *IEEE SciVis* has the smallest number. This is probably because many SciVis papers present advanced algorithms in rendering or interaction, which do not require multiple views. We also notice that numbers of MVs in *EuroVis* and *PacificVis* increased much in 2019, and the number of MVs in all the conferences is slightly growing.

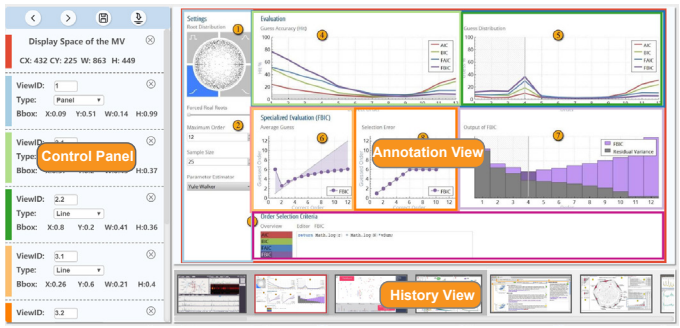


Fig. 2. Our annotation tool consists of three components: Annotation View, Control Panel, and Historical View.

4.2 View Annotation

After collecting images of MV designs, we need to extract attributes of *type* and *bbox* for each view. First, we tried automated approaches, such as deep learning techniques. We trained a YOLOv3 [37] model using visualization images self-generated and from open datasets such as [20]. We applied the model to predict views in the MV images, which however, produced unsatisfactory results. For instance, the intersection over union (IOU) – a standard performance metric that measures the size of intersection divided by the size of union for one pair of bounding boxes, was 73.3% for views of parallel coordinates. This accuracy could not support further analysis, so we decided to manually label the views.

We opted to develop an annotation tool that is dedicated to label attributes of *type* and *bbox* of each view. As shown in Figure 2, our annotation tool consists of three components.

- **Annotation View** is the main view that displays the MV image being annotated. The view allows users to annotate rectangles within the image $I \in \mathbb{R}^{W_I \times H_I}$, where W_I & H_I indicate width and height of the MV image. First, users need to annotate a rectangle that specifies display space $S \in \mathbb{R}^{W_S \times H_S}$ of the interface, where $W_S \leq W_I$ & $H_S \leq H_I$. Next, users can draw rectangles within S. Each rectangle indicates $bbox_i$ for a view $view_i$, and is assigned to a unique color. We constrain $bbox_i$ within S by clipping the intersection of user-specified rectangle $rect_i$ and S, *i.e.*, $bbox_i = rect_i \cap S$. Users can reposition and resize a $bbox_i$ by dragging control points on the corner of each edge. We allow overlap or gap between two neighboring $bbox_i$ and $bbox_j$, which will be fixed in a post-processing stage (see Section 4.3).

[1] www.pypi.org/project/PyMuPDF/
[2] www.sourceforge.net/projects/pdftohtml/

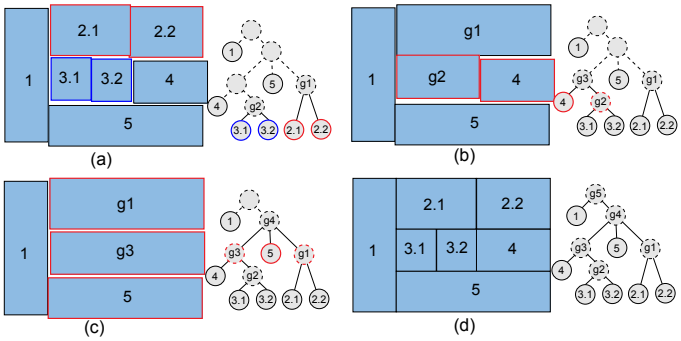


Fig. 3. A bottom-up approach is applied to refine manual layout annotations: views in a small multiples are grouped and aligned (a); neighboring views that can form a rectangle are grouped and aligned (b&c). The process is repeated until no more views can be grouped, generating the refined layout (d) without overlaps and gaps.

- **Control Panel** consists of a set of widgets allowing users to navigate or change view attributes. In the top, the blue region lays four buttons of *previous*, *next*, *save*, and *load*. Users can view the previous/next interface by clicking on *previous* or *next* button, respectively. If users feel comfortable with current annotations, they can *save* them; if users would like to make changes, they can *load* previous annotations.

  When a user annotates a rectangle in the *Annotation View*, a new section with a text field of *View ID* and a drop list of *Type* will be added to the panel. Each section is marked in the same color with the corresponding rectangle color in the *Annotation View*. In the *View ID* text field, the user can input a positive integer (*e.g.*, 1, 2) indicating the view is a level-1 view, or a one-decimal number (*e.g.*, 3.1, 3.2) indicating the view is a level-2 view, where the integral part indicates the small multiples ID, and the decimal part indicates the view number within the small multiples. In the *Type* drop list, the user can choose one out of the 14 view types (see Section 3). Users can also choose to delete the view using the *trash* button attached besides.
- **History View** overviews all MVs in the dataset by showing an image thumbnail for each MV. Annotated MVs are shown in the gray background; the one being annotated is marked with a red outline, and unannotated ones are shown in the white background.

After finishing annotating a MV, we store the labeling results in a JSON file named by the paper DOI. The JSON file records center position and size of the MV in the image, and an array named *views* storing information of *type* and *bbox* of all views. Level-2 views in a small multiples are stored as a nested array named *small multiples* within *views* array. Specifically, $bbox_i$ of a view $view_i$ stores the normalized center position (x_i, y_i) and size $w_i \times h_i$ to facilitate comparison among different MVs. By referring to the interface position and size, we can recover the exact position of each view in the image.

4.3 Layout Refinement

Manual annotation would inevitably cause overlaps or gaps between *bbox* of neighboring views. To remove the effect on follow-up analyses, we refine *bbox* annotations using a bottom-up approach, as illustrated in Figure 3. Here, the approach takes a set of rectangles $BB := \{bbox_1, [bbox_{2.1}, bbox_{2.2}], [bbox_{3.1}, bbox_{3.2}], bbox_4, bbox_5\}$ as input. Notice here *BB* is the annotation results for the MV displayed in Figure 2. The algorithm works as follows:

1. First, the algorithm checks if any two or more views are forming a small multiples, by referring to the view ID information stored in the JSON file. For example, $bbox_{2.1}$ and $bbox_{2.2}$ in Figure 3(a) are bounding boxes of two views forming a small multiples. Next, we group the *bbox*es together, forming a group of *bbox*es (denoted as BB_g). We identify a minimum rectangle $bbox_g$ that encloses all $bbox \in BB_g$. As in Figure 3, we can derive $bbox_{g1}$ for $BB_{g1} := \{bbox_{2.1}, bbox_{2.2}\}$, and $bbox_{g2}$ for $BB_{g2} := \{bbox_{3.1}, bbox_{3.2}\}$.

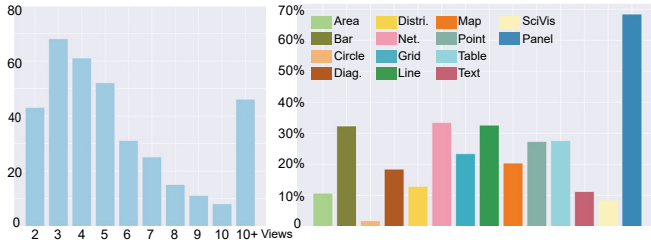


Fig. 4. Preliminary analyses on distribution of view count (left), and frequency of each view type (right).

2. Next, we update $bbox_i \in BB_g$ as follows: (i) in case if the top/left/bottom/right margin of $bbox_i$ to $bbox_g$ is smaller than a threshold θ, we align $bbox_i$ to the top/left/bottom/right of $bbox_g$; (ii) in case if an overlap or gap between two neighboring boxes $bbox_i$ & $bbox_j$ is smaller than θ, we stretch or shrink $bbox_i$ and $bbox_j$ to remove the misalignment. Here, we set θ as 3% of the average width and height of $bbox_g$.
3. All $bbox_i \in BB_g$ are removed from *BB* while $bbox_g$ is added into *BB*, forming a new set BB'. In Figure 3(b), $BB' := \{bbox_1, bbox_{g1}, bbox_{g2}, bbox_4, bbox_5\}$.
4. We then check if any two or more boxes in BB' can be grouped upon the following conditions:
   - The boxes are in the same *"neighborhood"* – centers of the boxes can be connected in a straight line without crossing some other box. For instance, $bbox_{g2}$ & $bbox_4$ are neighbors, while $bbox_{g1}$ & $bbox_5$ are not because connections between them will pass through $bbox_{g2}$ and $bbox_4$.
   - The box centers are in horizontal or vertical, and widths/heights of the boxes are nearly the same. For instance, $bbox_{g2}$ & $bbox_4$ satisfy the condition, while $bbox_{g2}$ & $bbox_1$ because their heights are very different.

   As in Figure 3(b), only $bbox_{g2}$ & $bbox_4$ meet the conditions.
5. We update the boxes as described in Step 2. Specifically, all sub-boxes in a group box, *e.g.*, $bbox_{3.1}$ & $bbox_{3.2}$ in $bbox_{g2}$ will be updated accordingly.
6. We repeat Step 2-5 until all boxes are grouped.

Finally, we generate a refined layout (Figure 3(d)) without overlaps and gaps for each MV. We refine all interfaces in the annotation dataset. Supplementary Table S3 presents some annotation results.

5 Composition and Configuration Analysis

After annotating all collected MV images (Section 4), we create a new corpus $C_{mv} = \{MV_i\}_{i=1}^{n}$, where $n = 360$ is the number of collected MVs in this work. Each view $view_i$ is represented as a two-tuple $view_i := (type_i, bbox_i)$. We first conduct some preliminary analyses on the distribution of view count and frequency of view type. Figure 4 presents the results. As shown in Figure 4 (left), most MVs (62.2%) comprise no more than five views, with 68 three-view being the most common MVs, followed by 61 four-view, 52 five-view, and 43 two-view. This indicates that designers opt for simple MVs with a small number of views. Figure 4 (right) shows that most MVs include a *Panel* (68.3%) of menus, legends, and narrative texts. Next to *Panel*, we see *SciVis* is not frequent (8.3%), as there are not many SciVis MVs in the collected dataset. Among view types of information visualization, *Bar* (32.2%), *Net.* (33.3%), and *Line* (32.5%) exceed 30%, whilst *Circle* charts are seldom (only 1.6%) adopted.

However, though interesting, the preliminary analyses do not provide answers for practical questions such as "which view types are frequently used together?", or "where to position each view?". We conduct further analyses on the composition pattern of view types (Section 5.1), configuration patterns of view layouts (Section 5.2), and integrated composition and configuration patterns (Section 5.3).

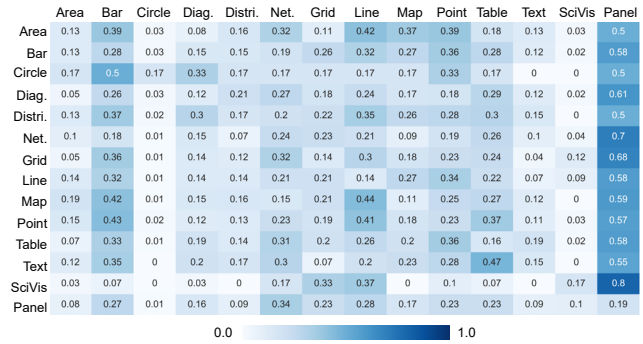


Fig. 5. Conditional probabilities of view types (columns) given other view types (rows) are employed in a MV.

5.1 Composition Pattern Analysis

The composition pattern is defined by the view types and their usage frequency in MV designs. Figure 4 (right) shows the frequency of individual view types, but not how view types are used together. We compute pairwise relationships between two view types ($type_i$ and $type_j$) using conditional probability:

$$P(type_i|type_j) = P(type_i \cap type_j)/P(type_j) \quad (1)$$

$P(type_i|type_j)$ ranges from [0, 1]: close to 0 values indicate $type_i$ rarely appears in *MV*s including $type_j$, whilst close to 1 values indicate $type_i$ always appears in *MV*s including $type_j$. Note that $type_i$ and $type_j$ could be the same, which indicates the probability that the same view type appears two or more times in one *MV*.

Figure 5 presents the results, where $type_i$ is arranged in the columns and $type_j$ in the rows. In the column of *Panel*, the conditional probabilities are more than 0.5 given the other 13 view types, but *P*(*Panel*—*Panel*) is rather low, indicating it is rare to have two or more separate *Panel*s in one *MV*. We can also notice that the column of *Bar* is quite dark, indicating that *Bar* charts are frequently used together with other view types. An exception is *SciVis*, with *P*(*Bar*—*SciVis*) of 0.07, indicating that few *SciVis* visualizations incorporate *Bar* charts. In contrast, *SciVis* frequently adopts *Panel*, as *P*(*Panel* — *SciVis*) = 0.8 is the highest value amongst all conditional probabilities. Moreover, $P(type_i|type_j)$ and $P(type_j|type_i)$ can be very distinct. For instance, *P*(*Bar*—*Circle*) reaches 0.5, whilst *P*(*Circle*—*Bar*) is only 0.03. The difference is probably caused by the frequency differences between *Circle* and *Bar* charts (see Figure 4 (right)).

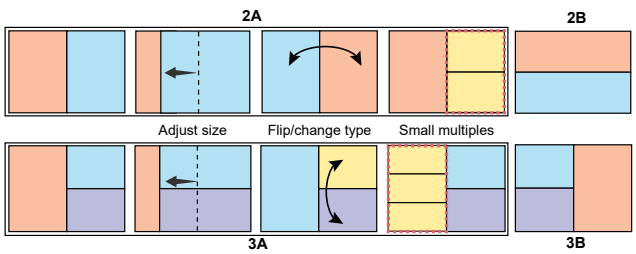


Fig. 6. Rules for coding MV layout: First, we assign a numeric number corresponding to the number of level-1 views/small multiples; Second, we assign an alphabetic character based on slice-and-dice order to distinguish layouts with the same view number.

5.2 Configuration Pattern Analysis

The configuration pattern characterizes the spatial arrangement of view layouts in the display space. We adopt a twofold coding rule to encode the configuration patterns of a MV design. In particular, each pattern is assigned two values, a numeric value followed by a letter (e.g. 2A, 3C, etc.). The rules of our coding scheme are described below.

- The numeric value of the coding rule refers to the number of "top-level" views in a MV (i.e. views that in the first level of the hierarchy). As in Figure 6, the MV designs in the top row will be assigned a 2, whereas the bottom ones will be assigned a 3.

Table 2. Top 10 layouts: numbers (green bars) and percentages (blue bar) in VAST, InfoVis, SciVis, EuroVis, and PacificVis.

Layout	VAST	InfoVis	SciVis	EuroVis	PacificVis	Total
2A	17 8.6%	18 36.7%	6 22.2%	7 14.9%	2 5.0%	50 13.9%
3C	17 8.6%	8 16.3%	1 3.7%	8 17.0%	2 5.0%	36 10.0%
3A	16 8.1%	4 8.2%	0 0.0%	1 2.1%	2 5.0%	23 6.4%
3B	6 3.0%	5 10.2%	4 14.8%	2 4.3%	2 5.0%	19 5.3%
4E	10 5.1%	0 0.0%	1 3.7%	2 4.3%	4 10.0%	17 4.7%
3F	6 3.0%	1 2.0%	0 0.0%	4 8.5%	0 0.0%	11 3.1%
3E	6 3.0%	2 4.1%	0 0.0%	1 2.1%	1 2.5%	10 2.8%
4H	4 2.0%	1 2.0%	1 3.7%	2 4.3%	1 2.5%	9 2.5%
4C	4 2.0%	1 2.0%	1 3.7%	1 2.1%	1 2.5%	8 2.2%
2B	1 0.5%	2 4.1%	1 3.7%	2 4.3%	1 2.5%	7 1.9%
Total	87 44.2%	42 85.7%	15 55.6%	30 63.8%	16 40.0%	

- The letter in the code refers to a specific type of layout. Table 2 shows some of these examples. The letters themselves are not immediately meaningful. They are enumerated based on a slice-and-dice approach [24] to distinguish between different layouts found in our corpus. For instance, the layout of ◫ is accomplished by a vertical slice, while layout of ⊟ is accomplished by a horizontal slice. As such in Figure 6, the top one is assigned the letter 'A' and the later the letter 'B'. Operations other than slicing, such as to adjust view size, and to flip or change view types, would not affect the slice-and-dice orders and are therefore disregarded. When used with the numeric value, we produce the twofold encoding scheme (e.g., 2*A* for ◫, and 2*B* for ⊟).

Based on this encoding scheme, we identify 98 unique layouts from 360 MV designs. Table 2 presents the top 10 layouts, and the frequency of their uses (both in counts and percentages) in VAST, InfoVis, SciVis, EuroVis, and PacificVis. The result suggests that most MV designs adopt simple layouts, as the top 10 MV layouts all employ four or fewer views. A closer investigation reveals differences between the conferences: The top 10 layouts account for 85.7% in InfoVis, followed by 63.8% in EuroVis, 55.6% in SciVis, 44.2% in VAST, and least 40% in PacificVis; and 2*A* occupies higher percentages in InfoVis (36.7%) and SciVis (22.2%) than the other three conferences. The findings suggest that InfoVis for abstract data, and SciVis for scientific data, tend to adopt simpler layouts, while VAST for data analysis requires more views to convey data from multiple perspectives. In addition, even though EuroVis and PacificVis are both venues for all disciplines, EuroVis tends to show more preference for simpler layouts.

5.3 Integrated Composition and Configuration Analysis

We further conduct integrated composition and configuration analysis, aiming to reveal the relationship between view types and layouts. Each view $view_i$ embraces two-tuple $(type_i, bbox_i)$, where $bbox_i$ consists of position (x, y) and size $w \times h$. From the data, we can associate view type with the position, and view type with size. Note that absolute view sizes and positions are dependent on the display size, which is not accessible from MV images. Instead, we analyze aspect ratios (Section 5.3.1) and relative positions (Section 5.3.2) of view types.

5.3.1 View Type & Aspect Ratio

We first compute aspect ratio (*ARC*) of a view $view_i$ as $ARC_i = w_i/h_i$. *ARC* ranges from (0, +∞): the value of 1 corresponds to a view in square, while values close to 0 or towards +∞ indicate that the views are vertically or horizontally long and narrow, respectively. Next, we group all views according to their view types, yielding 14 groups of *ARC* values. For each group, we depict its *ARC* distribution using the box plot as in Figure 7 (left). Here we show only the range [1/10, 10] that most *ARC* values fall in. Since *ARC* and $1/ARC$ are reciprocal, we

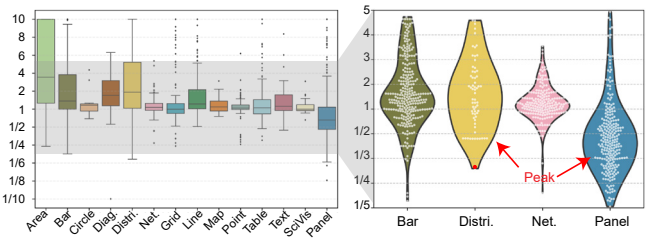
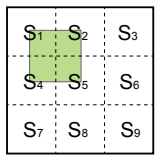


Fig. 7. Box plot (left) overviews aspect ratios of all 14 view types, and violin plot (right) depicts detailed aspect ratios of *Bar*, *Distribution* (*Distri.*), *Tree and Network* (*Net.*), and *Panel* within the range [1/5, 5].

put 1 at the center, and [1/10, 1) and (1,10] as symmetric around 1. We can observe that *ARC*s of most view types are within the range [1/2, 2], with mean values fall around 1. Some exceptions are *Area* charts with most *ARC*s larger than 1, and *Panel* with most *ARC*s less than 1. From follow-up investigation, we find that many MVs arrange horizontally long and narrow *Area* charts in small multiples, and many MVs employ vertically long and narrow *Panels* on the left/right side.

We further select four representative view types of *Bar*, *Distri.*, *Net.* and *Panel*, and observe their *ARC* distributions using the violin plot (Figure 7 (right)). We can notice that the mean *ARC* of *Bar* charts is near 1, yet there are some narrow *Bar* charts with *ARC*s towards 1/5 and 5. For *Distri.* views, there is a peak of *ARC*s (the white dots) around 1/2, and most *ARC*s are above 1/3 except one outlier. For *Net.* views (e.g. node-link diagrams), we see their *ARC*s are more concentrated within [1/2, 2] with a mean value around 1. This is probably because *ARC*s of *Net.* views are more independent with the underlying data, in contrast to other view types like *Bar* charts that need to increase its width to accommodate for the increased number of data attributes. Lastly, we can see *ARC*s of *Panel* are mostly below 1 with a peak around 1/3, which indicates that the *Panel*s are typically vertically long and narrow.

5.3.2 View Type & Relative Position



Position. To understand how individual views are positioned in MVs, we measure the relative positions of different view types. Inspired by stability measurement for treemap layout [46], we divide the display space into 3×3 grids $\{S_1, S_2, ..., S_9\}$ as shown in the inline figure. Next, we model the relative position of $view_i$ in the display space S as $S(view_i) = \{p_{i,1}S_1, ..., p_{i,9}S_9\}$, where $p_{i,k}$ stands for the proportion of overlapping area between $bbox_i$ and grid S_k multiplied by the size of S_k. For instance, consider the inline figure, the overlapping area between the green view (denoted as $view_1$) and S_1 is 1/4 the size of S_1, while S_1 is 1/9 the size of entire view space S. Hence, $p_{1,1} = 1/4 \times 1/9 = 1/36$. Similarly, we can compute $p_{1,2}$, $p_{1,4}$, and $p_{1,5}$ as 1/36, while the other $p_{1,j}$ are 0. Together, we can represent the relative position of the green view as $\{\frac{1}{36}S_1, \frac{1}{36}S_2, \frac{1}{36}S_4, \frac{1}{36}S_5\}$.

The use of this encoding scheme has several advantages: first, each view is explicitly represented – we can derive relative positions and sizes of different views in a MV. Second, the representation is consistent across MVs – we can compare positions and sizes of views in different MVs. Last, we can sum up multiple views of the same view type by simply adding up their p values. The sum stands for relative position and size for the view type rather than a specific view.

The stacked bar chart in Figure 8 (left) depicts the average p values for each view type in MV designs. The overall bar length depicts the average size of each view type. We can notice that several view types, including *Diag.*, *SciVis*, and *Net.*, occupy larger areas – over 25% of the display space. In contrast, *Area* and *Bar* occupy only ~5% of the display space, even though the frequency of *Bar* charts are quite high (see Figure 4 (right)). This may suggest that designers tend to assign small spaces for *Area* and *Bar* charts, as the view types are typically used for depicting summary statistics. We expected *Circle* to show similar p values with those of *Area* and *Bar*, but surprisingly *Circle* occupies much bigger space of about 25%. We investigated carefully the MVs containing *Circle* in the database and found that works on infographics design use *Circle* charts to illustrate their approach; see Supplementary Table S3-2A for an example. In contrast, *Circle* occupies much smaller sizes in other visual analytics MVs; see Supplementary Table S3-Other Layouts for an example.

From individual bars, we can derive relative positions of each view type. We notice that *Panel* is rarely distributed in the center of the display space, as its p values for grids S_2, S_5, and S_8 are very small. In contrast, *Diag.* is mostly placed in the center, as its p values for grids S_2, S_5 and S_8 are relatively high. In addition to *Diag.*, the other large view types, *i.e.*, *SciVis* and *Map*, also show higher p values at grids S_1, S_2, S_4, and S_5. This indicates that designers tend to place large views in the top-left and center regions of the display space. The other view types present balanced p values for the nine grids, indicating their positions can be inconsistent depending on designs.

Position Stability: To check if the relative positions of a view type can, in fact, be inconsistent, we measure position stability of a view type in different MVs. We adopt the metric of *relative-position-change* [46] to measure the distance between two views $view_i$ and $view_j$:

$$D(view_i, view_j) = \frac{1}{2}\sum_{k=1}^{9} |p_{i,k} - p_{j,k}| \quad (2)$$

D ranges from [0, 0.5]: close to 0 values indicate similar, consistent positions across MV designs, whilst close to 0.5 values indicate positions are exclusively distinct (i.e. highly inconsistent). From the *relative-position-change*, we can derive stability (denoted as STB) of $type_k$ in the collected MVs as follows:

$$STB(type) = \frac{1}{m(m-1)}\sum_{i=1}^{m}\sum_{j=1, j\neq i}^{m} D(view_i, view_j) \quad (3)$$

where m indicates the number of MVs that include the view $type$.

Figure 8 (right) presents the stability of each view type in the top 10 layouts. The cells with null values indicate that the layout contains at most one MV with the view type. We can observe that most *STB*s are below 0.2. For example, *STB*s of *Panel* are about 0.1 in all top 10 layouts, indicating positions and sizes of *Panel* are rather stable. This is probably because designers commonly allocate *Panel* in the periphery. Nevertheless, there are several unstable cases with *STB*s over 0.3, including *Text* in layout *4H*, *Point* in layout *4C*, and *Map* in layout *2B*. Taking *Map* in layout *2B* for example, we notice that chances of positioning maps in the top or the bottom are almost half and half. In contrast, *Map* in layout *2A* is rather stable, as we see most maps are positioned in the left, rather than in the right.

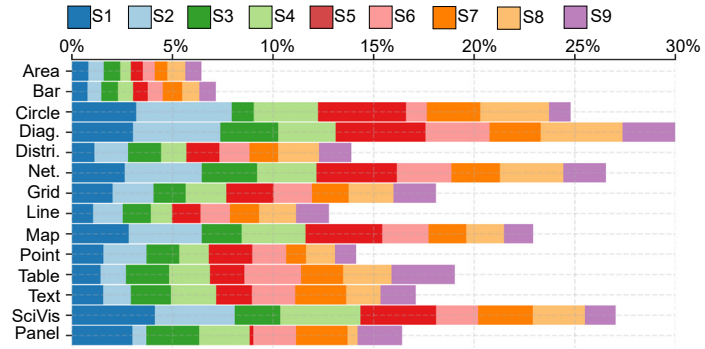
6 Recommendation System

Our analyses are nuanced and therefore difficult to turn into a cleanly articulated design guideline for MVs. As such, we aggregate our findings into a recommendation system that can help a designer choose the most appropriate MV designs given their data and needs. Our recommendation system can be used in two interactive modes: *Exploration* mode (Section 6.1) and *Design* mode (Section 6.2).

6.1 Exploration Mode

Exploration mode enables faceted exploration of existing MVs. Based on the prior analyses, each MV includes attributes of *view types*, *number of views*, and *layout*. Moreover, the MV images are derived from publications that include attributes of *year*, *venue*, *authors*, etc. We develop *Exploration* mode as illustrated in Figure 9, which consists of two main components:

- **Exploration View**, shown in Figure 9(b), adopts a unit visualization to present the query result. Each existing MV design is depicted as a dot, with color representing extrinsic attributes of *year* or *venue*. Users can group the queried MVs based on their intrinsic attributes of *Number of Views* or *Layout*. We only present the first 10 groups in case more than 10 groups are formed. Here we omit *View Type* since a MV typically consists of multiple view types. Coloring and grouping options are provided as a drop-down list.



	2A	3C	3A	3B	4E	3F	3E	4H	4C	2B
Area	—	0.05	—	0.08	0.04	0.03	0.16	—	—	—
Bar	0.07	0.07	0.03	0.07	0.11	0.21	—	0.11	0.04	—
Circle	0.11	—	—	—	—	—	—	—	—	—
Diag.	0.24	0.20	0.22	0.25	0.24	—	0.15	—	—	—
Distri.	—	0.23	0.20	0.20	0.13	—	—	—	—	—
Net.	0.19	0.19	0.21	0.18	0.15	0.16	0.14	—	0.10	0.05
Grid	0.25	0.19	—	0.09	0.21	0.12	0.12	0.19	0.15	—
Line	0.11	0.12	0.08	0.16	0.14	0.11	—	—	0.06	0.23
Map	0.13	0.15	0.04	0.21	0.06	—	0.16	—	—	0.30
Point	0.12	0.05	0.12	0.15	0.15	0.20	0.10	0.19	0.32	—
Table	0.22	0.25	0.15	0.08	0.14	0.17	—	0.10	0.13	—
Text	—	0.14	0.18	—	0.15	—	—	0.33	0.21	—
SciVis	0.23	0.08	—	0.09	—	—	—	—	0.23	—
Panel	0.14	0.10	0.09	0.10	0.11	0.16	0.11	0.11	0.08	0.06

Fig. 8. Relative position of each view type averaged from all MVs (left), and stability of each view type in the top 10 layouts (right).

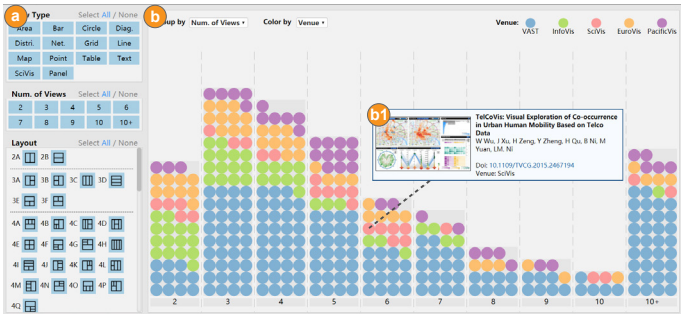


Fig. 9. Exploration mode mainly consists of *Query Panel* (a) that enables multi-faceted query, and *Exploration View* (b) that depicts the query results using unit-based visualization.

Details of a *MV* is shown when the mouse hovers over the dot; see Figure 9(b1) for an example. The detail view consists of two perspective information: first, the *MV* image is presented on the left side; second, the publication metadata, including *title*, *authors*, *doi*, and *venue*, are presented on the right side. Users can click on *doi* that links to the publication website.

- **Query Panel**, shown in Figure 9(a), consists of three faceted panels – *View Type* (T), *Number of Views* (N), and *Layout* (L), which are the essential elements of composition and configuration patterns. Each panel comprises of all possible attribute values, *e.g.*, *Bar* & *Line* in *View Type*, *2 to 10+* in *Number of Views*, and ◫ & ⊟ in *Layout*.

  Using the query panel, a user can filter and select the MV designs that meet their goals. Users can select multiple attribute values: in case the attribute values fall in the same panel, we take the union of filtering results by individual attribute value; in case the attribute values fall in different panels, we take the intersection of filtering results by individual attribute values.

Figure 9 presents a screenshot of our recommendation system in *Exploration* mode. Here, the query result is all MVs in the dataset, since all attribute values are selected. The grouping option is *Number of Views*, and the same distribution with that in Figure 4 (left) is presented. The coloring option is *Venue*: the most of dots are colored as light blue (*VAST*), and amounts of dots in the other four colors (*InfoVis, SciVis, EuroVis*, and *PacificVis*) are close. In Figure 9(b1), a dot corresponding to the MV in [54] is highlighted, which is a SciVis paper utilizing six views to present urban informatics.

6.2 Design Mode

Our recommendation system can also be used in the *Design* mode to recommend *MV* designs. The idea behind the design mode is for a user to "draw" their desired MV design, and the system will search through the database of MV designs to recommend the closest existing designs. Under the hood, the recommendation system takes into account the position of the individual views, their sizes, and the overall composition in finding the best matches. As shown in Figure 10, the recommendation system (in the Design Mode) consists of three views: *View Type List*, *Design Panel*, and *Recommendation Gallery*. The system supports the following operations:

- **Add/remove view**: Users can add a view by dragging-and-dropping a view icon from *View Type List* to *Design Panel* (Figure 10(a1)). A selected view will be surrounded by multiple view icons, with icon size corresponding to their correlations to the view of selection. Users can add a second view that is closely related to the selected view by clicking on the surrounding icon. Added views can be removed by clicking on the *delete* or *remove all* button in the top.
- **Adjust view**: Users can adjust the size of a selected view by dragging handles at its sides and corners (Figure 10(b1)) and adjust its position by dragging (Figure 10(b2)). Further, users can perform the "alignment" operation [55] to align several selected views (Figure 10(b3)).

With the above operations, users can already design the layout of a MV. However, the generated layouts may not reflect the good practice of MV designs. To suggest good and similar MV designs, our system searches through the database with the user-designed layout as input. This kind of exemplar-based recommendation has been widely applied in designing infographics (*e.g.*, [12, 35]), and mobile apps (*e.g.*, [48]). In our system, the recommendation algorithm works as follows:

- **Recommend layout**: As described in Section 5.3.2, we divide the display space of a *MV* into 3×3 grids. Each grid comprises up to 14 view types, with each view type occupies a certain proportion of the grid. By this, we can represent a *MV* as a three-dimensional tensor $T_{mv} \in \mathbb{R}^{3\times3\times14}$. Similarly, we can regard user input of multiple views as a 3D tensor $T_{in} \in \mathbb{R}^{3\times3\times14}$, by treating the minimum bounding box enclosing all views as the display space and dividing the space into 3×3 grids (see Figure 10(b3)).

  Next, we employ mutual information (*MI*) as a quantitative indicator for measuring proximity between T_{mv} and T_{in}. Since T_{mv} and T_{in} are both divided into 3×3 grids, they are geometrically aligned. Hence, the measurement is simplified without geometry matching. Before calculating the mutual information, we reshaped the T_{mv} and T_{in} to one-dimension vector, i.e., $1 * n$ and n is 126. Then we discretized the continuous data using Equal-Width discretization. *MI* is computed as

$$MI(T_{mv}, T_{in}) = \sum_{i \in T_{mv}} \sum_{j \in T_{in}} P(i,j) log \frac{P(i,j)}{P(i)P(j)} \quad (4)$$

  where $P(i)$ and $P(j)$ are the marginal probability distribution functions – computed via normalized intensity histograms – of T_{mv} and T_{in} respectively, and $P(i,j)$ is the joint probability function of T_{mv} and T_{in}.

  We iterate over all 360 MVs in the database, compute their *MI*s to user inputs, and sort them in descending order of *MI*. The sorted results are listed in *Recommendation Gallery*, with layouts and MV images arranged side-by-side. Users can filter the ordered MVs based on constraints of *Number of Views*. Layout of an example *MV* can then be applied to user inputs of views. Figure 10(d) presents two example layouts applied to user inputs.

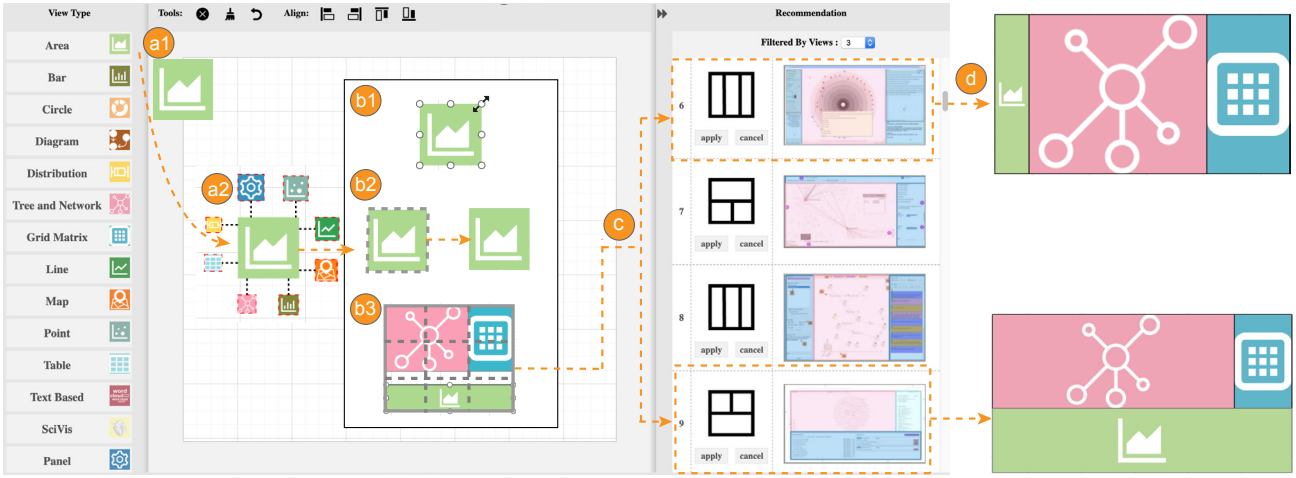


Fig. 10. Design mode incorporates a set of operations to support interactive design: to add view through drag-and-drop (a1), or icon selection (a2); to adjust view size (b1) or position (b2), or align multiple views together (b3); to recommend design based on configuration and composition proximity with existing designs (c). A recommended design can be further refined based on user preference (d).

Q4: improvements
1) overused colors;
2) more facets of information.

Q5: general feedback
enhance understandings of multiple-view visualizations.

Fig. 11. Qualitative feedbacks for the *Exploration* mode.

Note that we can divide a *MV* into finer grids, *e.g.*, 32×32, and represent a *MV* as a 3D tensor $\mathbb{R}^{32\times32\times14}$. The refinement can give better proximity between two MVs. However it will increase the computation time, and the effects are marginal as we observed from experiment results, probably because the number of views in a *MV* is limited. Therefore we empirically select 3×3 as the granularity of the grid in our system.

6.3 User Study

We conducted two studies to evaluate *Exploration* mode and *Design* mode of the recommendation system. Due to the COVID-19 pandemic, both studies were performed virtually.

6.3.1 Study 1: Qualitative Evaluation for Exploration Mode

Participants: We recruited 20 participants (6 females, 14 males, age: 23.65 ± 1.18) in the study. The participants all held a Bachelor's degree in fields like computer science, finance, and engineering. They were familiar with WIMP (Windows, Icons, Menus, and a Pointing device) interfaces, but none of them has designed a MV visualization before.
Procedure: We first explained to the participants the domain-specific terms, e.g., MVs and view types. Then we introduced functionalities of the *Exploration* mode and gave the participants about 15 minutes to freely explore the interface. After they felt comfortable using the tool, we asked the participants to find answers for 10 questions regarding composition and configuration patterns of MVs, *e.g.*, *"which layout is the most frequently used in MVs?"*, and *"how many MVs of six views contain bar chart?"*. The participants were asked to submit answers immediately when they felt confident with their solution. After the main study, the participants were asked to finish a questionnaire (see Supplementary Table S4). The questionnaire comprised three 7-point Likert scale questions regarding general impression (Q1), usefulness (Q2), and simplicity (Q3), and two free-form questions on what can be improved (Q4) and general feedback (Q5). Answers for the 7-point questions range from strongly disagree (1) to strongly agree (7).
Results: All participants finished the 10 questions in eight minutes. The accuracy of answers for each question is over 90%, and overall the average accuracy is up to 97.8%. Figure 11 presents the results for the questionnaire. Overall, the participants had an overall positive impression of the system (Q1) (mean = 6, SD = 0.75), found the system to be useful (Q2) (mean = 6.11, SD = 0.74), and perceived the *Exploration* mode as easy to use (Q3) (mean = 6.21, SD = 0.85). In the free-form question on improvements (Q4), two suggestions for improvement were raised: (i) some colors are overused, and (ii) more facets of information can be integrated, such as *authors* and *research topics*. The second aspect could be addressed in the near future by building a more comprehensive database of MV designs.

6.3.2 Study 2: Quantitative Evaluation for Design Mode

Participants: We recruited 13 participants (2 female and 11 male) who are graduating Ph.D. students, PostDocs, and junior professors in data visualization. All participants have experience in visualization design for at least four years.
Procedure: We first introduced the motivation of the work and functionalities of the *Design* mode. Next, we gave the participants about 15 minutes to freely explore the interface. After that, the participants were asked to complete three tasks of arranging 3 (Task 1), 5 (Task 2), and 7 (Task 3) views into a MV. Three design modes were provided:

- *Basic* mode only includes basic functions of *add/remove view*, *adjust view* by resizing and dragging;
- *Partial* mode includes an additional function of *adjust view* by alignment; and
- *Full* mode further includes the function of *recommend layout*.

The completion time of each task and design mode was recorded. The order of the design mode was randomly assigned to each participant in order to counter-balance learning effects. In the end, participants were asked to fill out user feedback (Supplementary Table S5) on the general impression of the *Design* mode.
Hypotheses: We anticipate completion time is dependent on both the number of views and design mode. We formulated two hypotheses: first (**H1**), the task of arranging 7 views will be slower than arranging 5 views, and arranging 5 views will be slower than arranging 3 views. We expect an increase in completion time when the number of views increases. Second (**H2**), *Full* mode is the fastest, followed by *Partial* mode and *Basic* mode. We expect alignment and recommendation functions will save time when designing MVs.
Result: We collected in total 3 (tasks) $\times$ 3 (modes) $\times$ 13 (participants) = 117 trails. We removed abnormal trails of completion times exceeding two times than the others by two participants. The remaining data were in line with the normal distribution after testing with Shapiro-Wilk test, and we performed a two-way ANOVA (3 tasks $\times$ 3 modes) to analyze the data. Figure 12 (left) summarizes the result. Significant effects of task complexity on completion time ($F(2,90) = 32.21, p < 0.001$) were observed. We further performed post-hoc comparisons of completion time among the number of views. Task 1 is on average $36.7s$ faster than Task 2 ($p < 0.001$), while Task 2 is on average $57.8s$ faster than Task 3 ($p < 0.001$). The results confirmed **H1**. However, no

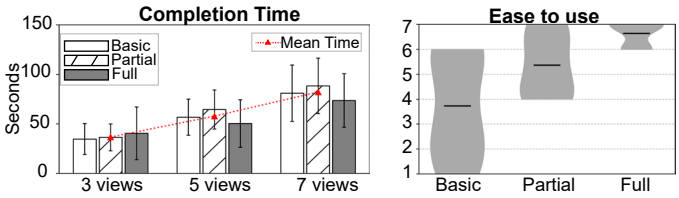


Fig. 12. Results of quantitative evaluation for *Design* mode: completion time (left), and feedback on the easy of use (right).

significant effects of design mode on the completion time ($F(2,90) = 0.86, p > 0.1$) were observed. This result leads us to reject **H2**.

Observations: Though **H2** was not supported, we observed interesting design practices from the experiments. We noticed that in *Full* mode, some participants looked through recommended layouts, and tried out different layouts attempting to find an optimal one, whilst in *Basic* and *Partial* modes, the participants directly arranged the views by dragging and resizing. Looking up and trying out example layouts in *Full* mode cost additional time in completing the task, which explains why completion times of *Full* mode are more diverse, especially for Task 1. We also observed that when more views were provided, participants tended to leave gaps and overlaps in the design in *Basic* mode, whilst they carefully aligned all views using alignment or recommendation functions in *Partial* and *Full* modes. The participants complained that it is too difficult to generate a complete layout in *Basic* mode.

Feedback. After the experiments, we collected feedback from the participants using 7-point Likert scale questions. Figure 12 (right) presents results on the usability of the three design modes. The participants felt that the *Full* mode (mean = 6.6) is easier to use than *Partial* (mean = 5.2) and *Basic* mode (mean = 3.5). Using the Bonferroni post-hoc test, we found a significant difference among the modes ($p < 0.01$). More results are provided in Supplementary Table S5. Overall, the participants thought the idea of example-based MV designs is promising: The system is novel, interesting, and useful for designing MVs (Q4); All participants agreed that the interface is intuitive (Q5), and interactions and recommendation results are useful (Q6); The idea could also be useful for other applications like infographics and mobile app design (Q9). For future work, the participants suggested showing mockup views of real data rather than icons, to consider view directions, and to export the layout as a JSON file (Q7).

7 Discussion, Limitation, and Future Work

We conducted a comprehensive analysis on the *composition* and *configuration* patterns of MVs in visualization literature. The analyses revealed some common practices:

- Most MVs present less than five views. Designers shall be careful when the MV design compromises too many views. Simply putting more views together may cause information overload.
- Simpler and perceptually more accurate view types are preferred, *e.g.*, *Bar* and *Line* charts. On the other hand, *Circle* charts, such as donut and pie charts, were seldom used.
- Most MVs adopt simple layout, *e.g.*, 2A ◫ and 3A ▥, especially for InfoVis and SciVis papers. VAST papers tend to adopt a bit more complex layouts, as shown in Table 2.
- Most view types show a medium range of aspect ratio within [1/2, 2], except for some types like *Area* and *Panel*.
- There are common positions for certain view types. For instance, *Panel* typically are not positioned in the center of the display space, whilst *Diagrams* are the opposite.

The findings advance our understanding of the design space of MVs. The revealed composition and configuration patterns also provide a foundation for the development of a recommendation system that can assist a designer in choosing appropriate MV designs. Usability and effectiveness of the recommendation system are confirmed by feedbacks from both visualization novices and experts.

Limitations. Nevertheless, there are certain limitations in our work. First, we categorize a view into one of the 14 view types. This categorization is not exhaustive, and many views cannot be classified within any of these view types. We found many novel designs with compound view types. In such scenarios, we categorize the views using the main view types, *e.g.*, *Line* chart for [57] that combines scatter points in parallel coordinates, and *Map* for [58] that overlays a novel chord diagram on maps. A more solid solution would be to adopt the concept of composite visualization views [22] that identified five design patterns of compositing views, *i.e.*, juxtaposition, superimposition, overloading, nesting, and integration. Second, we only analyze view type usages and view layouts, but not the underlying mechanism for selecting view types and arranging views. There are many factors that can affect the designs, such as underlying data, analytical tasks, individual preferences, and user perceptions. For instance, Wang et al. [51] proposed a comprehensive strategy for determining the aspect ratio of the line chart. A thorough consideration of all the factors would certainly be beneficial. We aim to include these factors in the future. Last, the current recommendation system is solely based on MV designs from visualization conferences. The dataset is however rather small, and the design space is limited. For instance, few MV designs in the dataset are for large-size displays. We will increase the corpus size by collecting more MV designs from other venues such as IEEE TVCG, ACM CHI, the Internet, and visualization authoring tools like Tableau.

Future Work. There are several promising directions for future works. First, many participants in Study 2 suggested incorporating views for real data instead of view type icons. Such functionalities can be supported by visualization libraries like D3 [8], and we plan to realize it in the near future. Second, we would like to analyze the underlying semantic structure that links multiple views, similar to visual information flow in infographics [29]. We anticipate that studying the semantic structure would improve the visualization community's understanding on how to arrange multiple views in the display space. Third, we would like to examine visual consistency between multiple views. Though the visualization community proposed many guidelines in keeping visual consistency (*e.g.*, [36]), we found many violations when annotating the MVs. We expect to find out some criteria that can be used to evaluate MVs. Lastly, dashboards that mostly use multiple views recently gain much attention in the visualization community. Sarikaya et al. [43] built a dataset including a series of dashboard images and corresponding design goals. This complements the limitation of our dataset lacking information of design goals. We would like to analyze composition and configuration patterns of these dashboards, and associate the patterns with concrete design goals. We anticipate that a deeper analysis would provide much clearer guidelines for MV design.

8 Conclusion

We have presented an empirical study on how the visualization community combines multiple views of different types in the display space, and have developed an interactive recommendation system that uses data from the analysis. The benefit of this work is prominent: First, the study advances the community's understanding on view type associations and view layout arrangements. The study was conducted on the basis of a new dataset containing 360 images of MVs collected from IEEE VIS, EuroVis, and PacificVis publications 2011 to 2019, and fine-grained annotations of view types and layouts for these visualization images. Second, the study provides the foundation for a recommendation system that assists a designer in designing MV, by enabling faceted exploration of existing MV designs. Third the tool recommends MV designs based on revealed composition and configuration patterns and user inputs. We release the dataset to advance future studies on MV design at the project homepage `https://mvlandscape.bitbucket.io/`.

Acknowledgments

The authors wish to thank all experiment participants for their valuable feedbacks, the anonymous reviewers for their fruitful suggestions, and Min Chen for his helpful conversation. This work was supported in part by National Natural Science Foundation of China (61802388), the National Science Foundation (OAC-1940175, OAC-1939945, IIS-1452977, DGE-1855886), and DARPA (FA8750-17-2-0107).

REFERENCES

[1] Power BI. https://powerbi.microsoft.com, last accessed on 26/04/2020.

[2] Spotfire. https://www.tibco.com/products/tibco-spotfire/, last accessed on 26/04/2020.

[3] Tableau. https://www.tableau.com/, last accessed on 26/04/2020.

[4] H. M. Al-maneea and J. C. Roberts. Towards quantifying multiple view layouts in visualisation as seen from research publications. In *Proceedings of the IEEE Visualization*, 2019.

[5] M. Q. W. Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, pages 110–119, 2000.

[6] B. B. Bederson, B. E. N. Shneiderman, and M. Wattenberg. *Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies*, pages 257–278. Morgan Kaufmann, 2003.

[7] M. A. Borkin, A. A. Vo, Z. Bylinskii, P. Isola, S. Sunkavalli, A. Oliva, and H. Pfister. What makes a visualization memorable? *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2306–2315, 2013.

[8] M. Bostock, V. Ogievetsky, and J. Heer. D^3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

[9] M. Bruls, K. Huizing, and J. van Wijk. Squarified treemaps. In *In Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42. Press, 1999.

[10] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in Information visualization: using vision to think*. Morgan Kaufmann, 1999.

[11] S. M. Casner. Task-analytic approach to the automated design of graphic presentations. *ACM Trans. Graph.*, 10(2):111–151, 1991.

[12] Z. Chen, Y. Wang, Q. Wang, Y. Wang, and H. Qu. Towards automated infographic design: Deep learning-based auto-extraction of extensible timeline. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):917–926, 2020.

[13] H. Chung, C. North, S. Joshi, and C. Jian. Four considerations for supporting visual analysis in display ecologies. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 33–40, 2015.

[14] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984.

[15] C. Collins and S. Carpendale. VisLink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1192–1199, 2007.

[16] V. Dibia and D. C. Data2Vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE Computer Graphics and Applications*, 39(5):33–46, 2019.

[17] 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.

[18] T. Horak, A. Mathisen, C. N. Klokmose, R. Dachselt, and N. Elmqvist. Vistribute: Distributing interactive visualizations in dynamic multi-device setups. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 616: 1–13, 2019.

[19] K. Hu, M. A. Bakker, S. Li, T. Kraska, and C. Hidalgo. VizML: A machine learning approach to visualization recommendation. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 128: 1–12, 2019.

[20] K. Hu, S. N. S. Gaikwad, M. Hulsebos, M. A. Bakker, E. Zgraggen, C. Hidalgo, T. Kraska, G. Li, A. Satyanarayan, and C. Demiralp. VizNet: Towards a large-scale visualization learning and benchmarking repository. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 662: 1–12, 2019.

[21] P. Isenberg, F. Heimerl, S. Koch, T. Isenberg, P. Xu, C. D. Stolper, M. Sedlmair, J. Chen, T. Möller, and J. Stasko. vispubdata.org: A metadata collection about IEEE Visualization (VIS) publications. *IEEE Transactions on Visualization and Computer Graphics*, 23(9):2199–2206, 2017.

[22] W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 1–8, 2012.

[23] W. Javed and N. Elmqvist. ExPlates: Spatializing interactive analysis to scaffold visual exploration. *Computer Graphics Forum*, 32(3pt4):441–450, 2013.

[24] B. Johnson and B. Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceeding of IEEE Visualization*, pages 284–291, 1991.

[25] A. Key, B. Howe, D. Perry, and C. Aragon. VizDeck: self-organizing dashboards for visual analytics. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 681–684, 2012.

[26] R. Langner, T. Horak, and R. Dachselt. VisTiles: Coordinating and combining co-located mobile devices for visual data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):626–636, 2018.

[27] 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(1):608–618, 2019.

[28] G. L. Lohse, K. Biolsi, N. Walker, and H. H. Rueter. A classification of visual representations. *Communication of the ACM*, 37(12):36–49, 1994.

[29] M. Lu, C. Wang, J. Lanir, N. Zhao, H. Pfister, D. Cohen-Or, and H. Huang. Exploring visual information flows in infographics. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[30] Y. Luo, X. Qin, N. Tang, and G. Li. Deepeye: Towards automatic data visualization. In *Proceedings of the IEEE International Conference on Data Engineering*, pages 101–112, 2018.

[31] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141, 1986.

[32] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007.

[33] K. Matkovic, W. Freiler, D. Gracanin, and H. Hauser. Comvis: A coordinated multiple views system for prototyping new visualization technology. In *Proceedings of the International Conference Information Visualisation*, pages 215–220, 2008.

[34] D. Moritz, C. Wang, G. L. Nelson, H. Lin, A. M. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in Draco. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):438–448, 2019.

[35] P. O'Donovan, A. Agarwala, and A. Hertzmann. Learning layouts for single page graphic designs. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1200–1213, 2014.

[36] 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, 2018.

[37] J. Redmon and A. Farhadi. YOLOv3: An incremental improvement. *ArXiv e-prints*, 2018.

[38] J. C. Roberts. On encouraging multiple views for visualization. In *Proceedings of the IEEE Conference on Information Visualization.*, pages 8–14, 1998.

[39] J. C. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Proceedings of International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pages 61–71, 2007.

[40] J. C. Roberts, H. 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*, 38(3):79–93, 2019.

[41] R. Sadana and J. Stasko. Designing multiple coordinated visualizations for tablets. *Computer Graphics Forum*, 35(3):261–270, 2016.

[42] B. Saket, D. Moritz, H. Lin, V. Dibia, C. Demiralp, and J. Heer. Beyond heuristics: Learning visualization design. *arXiv:1807.06641*, pages 1–4, 2018.

[43] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher. What do we talk about when we talk about dashboards? *IEEE Transactions on Visualization and Computer Graphics*, 25(1):682–692, 2019.

[44] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics.*, 11(1):92–99, 1992.

[45] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336 – 343, 1996.

[46] M. Sondag, B. Speckmann, and K. Verbeek. Stable treemaps via local moves. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):729–738, 2018.

[47] 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, 2002.

[48] A. Swearngin, M. Dontcheva, W. Li, J. Brandt, M. Dixon, and A. J. Ko. Rewire: Interface design assistance from examples. In *Proceedings of the*

ACM Conference on Human Factors in Computing Systems, pages 1–12, 2018.

[49] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, second edition, 1983.

[50] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis. Seedb: Efficient data-driven visualization recommendations to support visual analytics. *Proceedings of the VLDB Endowment.*, 8(13):2182–2193, 2015.

[51] Y. Wang, Z. Wang, L. Zhu, J. Zhang, C. Fu, Z. Cheng, C. Tu, and B. Chen. Is there a robust technique for selecting aspect ratios in line charts? *IEEE Transactions on Visualization and Computer Graphics*, 24(12):3096–3110, 2018.

[52] C. Weaver. Building highly-coordinated visualizations in improvise. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 159–166, 2004.

[53] W. Wu, L. Fan, L. Liu, and P. Wonka. Miqp-based layout design for building interiors. *Computer Graphics Forum*, 37(2):511–521, 2018.

[54] W. Wu, J. Xu, H. Zeng, Y. Zheng, H. Qu, B. Ni, M. Yuan, and L. M. Ni. Telcovis: Visual exploration of co-occurrence in urban human mobility based on telco data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):935–944, 2016.

[55] P. Xu, H. Fu, T. Igarashi, and C.-L. Tai. Global beautification of layouts with interactive ambiguity resolution. In *Proceedings of the ACM symposium on User interface software and technology*, pages 243–252, 2014.

[56] Y.-L. Yang, J. Wang, E. Vouga, and P. Wonka. Urban pattern: layout design by hierarchical domain splitting. *ACM Transactions on Graphics*, 32(6):1–12, 2013.

[57] X. Yuan, P. Guo, H. Xiao, H. Zhou, and H. Qu. Scattering points in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1001–1008, 2009.

[58] W. Zeng, C.-W. Fu, S. Müller Arisona, and H. Qu. Visualizing interchange patterns in massive movement data. *Computer Graphics Forum*, 32(3pt3):271–280, 2013.

[59] X. Zheng, X. Qiao, Y. Cao, and R. W. H. Lau. Content-aware generative modeling of graphic design layouts. *ACM Transactions on Graphics*, 38(4):1–15, 2019.