Towards Systematic Design Considerations for Visualizing Cross-View Data Relationships

Maoyuan Sun, Akhil Namburi, David Koop, Jian Zhao, Tianyi Li, and Haeyong Chung

Abstract—Due to the scale of data and the complexity of analysis tasks, insight discovery often requires coordinating multiple visualizations (views), with each view displaying different parts of data or the same data from different perspectives. For example, to analyze car sales records, a marketing analyst uses a line chart to visualize the trend of car sales, a scatterplot to inspect the price and horsepower of different cars, and a matrix to compare the transaction amounts in types of deals. To explore related information across multiple views, current visual analysis tools heavily rely on brushing and linking techniques, which may require a significant amount of user effort (e.g., many trial-and-error attempts). There may be other efficient and effective ways of displaying cross-view data relationships to support data analysis with multiple views, but currently there are no guidelines to address this design challenge. In this paper, we present systematic design considerations for visualizing cross-view data relationships, which leverages *descriptive aspects* of relationships and *usable visual context* of multi-view visualizations. We discuss pros and cons of different designs for showing cross-view data relationships, and provide a set of recommendations for helping practitioners make design decisions.

Index Terms—Cross-view data relationship, multiple views, visual analytics.

1 Introduction

Exploring data by coordinating multiple visualizations in different views [1] has been a common approach for data analysis in various fields (e.g., bioinformatics [2], business intelligence [3], cybersecurity [4], text analytics [5], and time-series analysis [6]). Each visualization supports certain analysis tasks by showing either different parts of data, or the same data from different perspectives. Analysts need to constantly relate information from multiple visualizations to gain a comprehensive understanding of a dataset [1].

As an example, suppose an analyst, Emma, is using Jigsaw [5] to explore a collection of intelligence reports. After loading the dataset, Jigsaw provides a *list view* that shows relations between named entities extracted from the documents, a *graph view* that displays connections between the entities and documents, and a *document view* that shows the original text (Figure 1). Emma investigates each view and relates the information from multiple views to explore hidden clues in the dataset. Jigsaw uses coordination-based techniques to support relating data across views. As Emma selects entities in the *list view*, Jigsaw highlights corresponding entities in the *graph view* and the related text in the *document view*. However, there are numerous ways to connect the information from the three views. Every time Emma selects a data point, she needs to check highlighted information in other views. After selecting twenty entities,

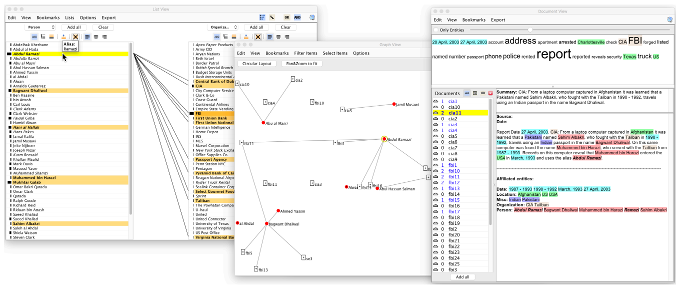


Fig. 1. An example of relating data from three visualizations in Jigsaw: a list view (left), a graph view (middle), and a document view (right).

Emma gets confused about which parts of highlighted text in the document view are related to each entity.

Relating data across multiple visualizations is not as straightforward as it looks like. Due to the lack of systematic design guidelines, prior research has primarily relied on brushing and linking based techniques [7], [8], [9], [10], [11], [12], [13], which have been used in many visual analysis tools (e.g., Caleydo [2], Canopy [14], IN-SPIRE [15], Jigsaw [5], and Tableau [16]). While brushing and linking based techniques can reveal cross-view data relationships, user interactions are limited to visual elements within the views (e.g., nodes in a graph) rather than visual relations at the view level. There is a need to support users viewing and inspecting visual marks that directly encode cross-view data relationships. Such relationship-based visual marks can help reduce user interaction effort. However, visual marks that encode relationships at the view level are usually defined by bundled edges that link multiple data points from different views. This can cause visual clutter, especially when a large number of visual marks are added.

This brings a design challenge of visualizing cross-view data relationships. In particular, what important factors do

- *Maoyuan Sun, Akhil Namburi, and David Koop are with the Department of Computer Science, Northern Illinois University, DeKalb, IL 60115. E-mail: {smaoyuan, namburi.akhil12, dakoop}@niu.edu.*
- *Jian Zhao is with the School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1. E-mail: jianzhao@uwaterloo.ca.*
- *Tianyi Li is with the Department of Computer Science, Loyola University Chicago, Chicago, IL. E-mail: tli@cs.luc.edu.*
- *Haeyong Chung is with the Department of Computer Science, University of Alabama in Huntsville, Huntsville, AL 35805. E-mail: haeyong.chung@uah.edu.*

Manuscript received April 19, 2020; revised August 26, 2020.

visualization designers and practitioners need to consider to show data relationships across multiple views? To address this, we present systematic design considerations for visualizing cross-view data relationships, which leverage 1) descriptive aspects of relationships and 2) usable visual context of multi-view visualizations. The former formalizes cross-view data relationships with a descriptive framework that highlights four aspects: *schema*, *structure*, *weight*, and *size*. The latter regards usable visual context of multi-view visualizations. Combining both, the design of visualizing cross-view data relationships can be considered as using certain, available visual resource of multi-view visualizations to present some particular aspects of the relationship. Based on this, visualization design options can be systematically examined, which enables comparing different designs.

In summary, this work highlights the following three key contributions:

- We present systematic design considerations for visualizing cross-view data relationships, including both descriptive aspects of relationships and usable visual context of multiple views.
- We analyze a variety of visualization design options, and discuss their pros and cons.
- We make a set of recommendations to help visualization designers and practitioners make design decisions.

2 Background

In this section, we clarify the notion of key terms used in this paper: data relationship, visual element and view. Data relationship is the focus of this work. Visual element and view are major components in multi-view visualizations.

2.1 Data Relationship

Our notion of data relationship is based on the concept of *entity set*. An entity set is a collection of *non-duplicated* data entities that share the same attribute(s) (e.g., a set of images, or a list of people's names). Given two entity sets A and B, a relationship between them, $R(A, B)$, is a subset of $A \times B$, where $\times$ denotes the Cartesian product operation. If $R(A, B)$ is not empty, we say that A is related to B. Otherwise, we say that A and B are independent from each other. In different scenarios, the relationship R may be determined differently. For instance, in text analytics, R can be defined as word co-occurrence; while in cybersecurity, R may be determined by communication-based attributes between MAC addresses and web pages. Based on this notion, we identify four key aspects of data relationships (Section 3) and further consider them in the context of multiple views (Section 4). This drives our design considerations as using *what available visual context* in multiple views to show *which aspects* of cross-view data relationships that users care about.

2.2 Visual Element

In a broad sense, a visual element can be considered to be anything that is shown on a display. Based on the visualization reference model [17], visual elements refer to spatial substrates, marks, and graphical properties, which consider possible visual forms mapped from data. Following this rationale, our notion of visual element is *a graphical representation unit that is designed based on data attributes and values*. For example, each node in a scatterplot reveals a data entity and the positions of nodes correspond to values on two specific attributes. Moreover, an aggregation of visual elements can be perceived as major components of a unit visualization (e.g., several spatially organized dots forming a perceived rectangle that indicates a bar in a bar chart) [18], [19]. Thus, visual elements serve critical parts of available visual context of multiple views, where visual encodings and user interactions can be applied to support exploring cross-view data relationships (Section 5.4 - 5.6).

2.3 View

There is no broad consensus on the definition of a view [20]. Based on prior work in information visualization and multiple coordinated views, a visualization view can be understood from the following four perspectives:

- A **process**-centric perspective: *visual mapping product*. A view results from a visual mapping of data [17], [21], and it is considered the final stage of Chi's data state reference model [21] that directly interacts with users.
- A **model**-centric perspective: *data + representation modeling*. A view is a set of data and specifications for displaying them [10], and different specifications lead to different *forms* of visual representations [9], [22].
- A **perception**-centric perspective: *spatial separation*. A view is an independent, separated and bounded area (e.g., windows) [7], [23], where data is displayed in some types of visual forms [20].
- A **task**-centric perspective: *analytics scaffolding*. A view supports users performing certain analytical tasks on data [13], exploring particular attributes of data [24], or recording analytical insights [25].

While these descriptions emphasize different aspects of a view, an important aspect, *semantic coherence*, seems missing. We regard semantic as the meaning of a series user actions corresponding to certain tasks [26]. Yet, a view reflects data transformations in some visual forms, and can be shown in a bounded area, intended to separate one from another. Such a boundary becomes vague when considering user tasks. For example, when exploring a scatterplot matrix, it is hard to say whether each scatterplot or the whole matrix is a view. When users try to understand a dataset based on two attributes only, one scatterplot is considered a view. When users attempt to explore eight attributes, it is reasonable to consider the whole matrix as a view.

We define a view as *a collection of visual elements that are spatially organized in a semantically coherent manner to support specific analysis tasks*. A view can be a stand-alone window of a desktop application (e.g., Jigsaw's Graph View [5]), or a bounded area in a web application (e.g., a chart with visible borders in MyBrush [27] and Vega-Lite [28]) that holds a specific visualization (e.g., a scatterplot). Moreover, a view can also be one component of a complex visualization (e.g., a block in Domino [29]) where such a component is perceptually distinguishable and usable for data explorations. Based on this notion, we have identified three types of cross-view data relationships (Section 3.2) and components of visual context in multiple views (Section 4).

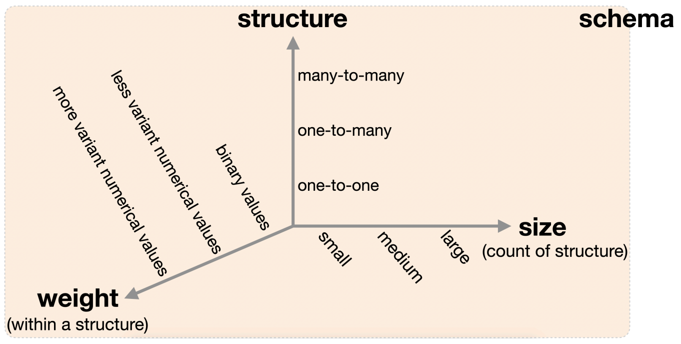


Fig. 2. A descriptive framework of data relationships with four aspects: schema, structure, strength, and size.

3 Data Relationship Framework

In this section, we first introduce aspects that characterize data relationships and then describe different types of cross-view data relationships.

3.1 Data Relationship Characterization

We characterize data relationships as analytical units in a descriptive framework with four aspects (Figure 2): *schema*, *structure*, *weight*, and *size*.

Schema is the overarching strategy used in an analytical task. It considers the *types* of information selected from a dataset. For example, a dataset has three types of information: person, location and organization. Data relationships can be built between entities of various types of information (e.g., a relationship between person and organization, or a relationship among location, computer MAC address, and website URL). It is similar to the relational database schema.

The types of information in a relationship-schema are a subset of all possible combinations of different types of information in a dataset. If a dataset has n different types of information (e.g., person, location, organization, date, and phone number), the number of total possible schemata is $C_n^1 + C_n^2 + C_n^3 + ... + C_n^n = 2^n - 2$, where C_n^m denotes the number of ways to combine m types of information from a total of n types of information. Specifically, C_n^1 indicates relationships of entities of the same type (e.g., clustering persons in an ego-centric network [30]). Thus, the more types of information a dataset has, the larger number of schemata may be generated from it. Moreover, users may use different schema for different analyses. Thus, for the same dataset, different schemata may be selected by different users, which further leads to discovering different relationship structures.

Structure is the organization of entities in a data relationship. Different structures can be described by the number of entities in each involved type of information [31], similar to the cardinality of a set.

As is shown in Table 1, there are four types of structures, defined by the number of entities in a data relationship. A *one-to-one* structure is an *individual*-level of relations between two entities. For example, one location in a list view is related to one person in a graph view. It is the most basic structure, based on which higher-level structures can be built. A *one-to-many* structure is a *single-group* level of relationships that associate a group of entities with one individual entity. For example, one location in a list view is related to seven

TABLE 1
Four types of relationship structures between visual elements.

Relationship Structure	Relationship Level	Mathematical Format
One-to-One	Individual	$1:1$
One-to-Many	*Single*-Group	$1 : e\ (e \geq 2)$
Many-to-Many (simple)	*Bi*-Group	$e : f\ (e, f \geq 2)$
Many-to-Many (complex)	*Multi*-Group	$e : f : ... : z\ (e, f, ..., z \geq 2)$

organizations in a graph view. A *many-to-many* structure is a high-level relationship that connects at least two groups of entities. In a simple case at the *bi-group* level, it connects two groups of entities. For example, five locations in a list view are related to seven organizations in a graph view. In a complex case at the *multi-group* level, it relates multiple groups (e.g., [32]). For example, five locations in a list view are related to seven organizations in a graph view, which are mentioned by six documents in a document view.

Weight is the strength of a data relationship. For example, during an investigation, suppose we discover that suspect Alan had a conflict with the victim at work, but Ivan owed a huge debt to the victim. We might consider next steps based on the strength of these links between the suspects and victim. Weight measures how strong entities are related. A simple model is binary, where the value is 0 for no relation, and 1 when a relation between two entities exists. In a complex case, the weight can take a continuous value (e.g., in the range $[0, 1]$), where a larger value indicates a stronger relationship between two entities.

Weight can be used to find relationship structures. As discussed by Sun et al. [33], biclusters, a type of many-to-many relationship structure, can be discovered by aggregating entities from two groups, where each entity in one group is related to all entities in the other group (i.e., the values corresponding to all their relations are 1). The weight of a bicluster relation can be measured by the number of entity-wise connections between the two groups. Also, given a relationship structure, we can check its weight by exploring each individual relations within the structure.

Size describes the *number* of relationship structures (e.g., ten individual-level relationships, three single-group relationships and five bi-group level relationships). Because we consider relationship structure as a basic analytical unit, size focuses on the number of such units, instead of the count of individual-level relationships within a relationship structure. For example, a person making three calls to a phone number is considered as *one* individual-level relationship.

In summary, the *structure*, *weight* and *size* of relationships are determined by the *schema*. The complexity of each aspect can impact the visualization design of data relationships.

3.2 Data Relationships across Multiple Views

We define data relationships across multiple views as 1) *between visual elements*, 2) *between views*, and 3) *between visual elements and views* (Figure 3).

The first type defines cross-view data relationships as relations between visual elements from different views. Each view has a collection of visual elements that encode data entities and/or attributes (e.g., nodes in data context map [34]). For a dataset, if relations between data exist, their corresponding visual elements, from different views, may logically inherit their relations.

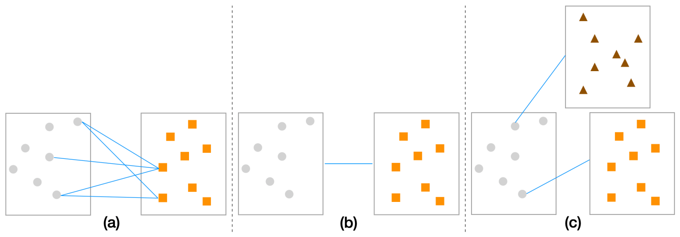


Fig. 3. Three types of cross-view relationships: a) between visual elements, b) between views, and c) between visual elements and views. Each box is a view. Gray circles, orange squares, and purple triangles are marks for visual elements, and blue lines indicate relations.

The second type treats view as an atomic component in a relationship. Relationships between views can reveal high-level insights [35], which go beyond simply aggregating relations between visual elements. For example, in ForceSPIRE [36], each document is shown in a window-based visual metaphor and users can spatially organize documents. The spatialization can reveal certain relations (e.g., documents with similar topics are displayed in a cluster). In this case, the data relationships are defined between document views rather than entities within the documents.

The third type involves both visual elements and views. It can be considered as a reflection of Shneiderman's visual information seeking mantra [37]. Consider an overview visualization that shows a scatter plot of persons being examined in an analysis. The visual elements in the plot can be related to other views (e.g., tables, maps and histograms) that show detailed information for specific individuals.

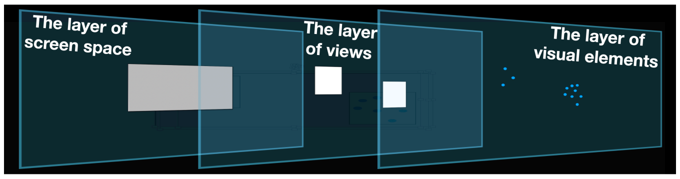
While all three types of relationships are important for cross-view analysis, the first type is more challenging to visualize than the other two. First, it is more granular than others since it involves detailed one-to-one relationship structures between visual elements. Visualizing such relationships requires encoding various relationship structures and their corresponding weights. As the complexity of data increases in views, such relationships can be hard to show and manage. Second, the number of visual elements is often larger than the number of views in real-world cases, presenting scalability challenges. Especially for the three group-level relationship structures between visual elements (rows 2-4 in Table 1), techniques like Euler Diagrams face potential problems [38]. Understanding design considerations for the first type of relationships, which are defined by the relations between visual elements, is therefore critical for visualizing cross-view relationships. In this work, we focus on addressing the above two challenges for visualizing the first type of cross-view data relationship.

4 Visual Context in Multiple Views

The previous section discusses the characteristics of data relationships, and we next seek visual encodings and user interactions that reveal these aspects. Thus, in this section, we explore available *visual context* in multiple views, that is, where can possible encodings and interactions be applied?

4.1 Components of Visual Context

We identify three *conceptual layers* with five *perceivable spaces* to characterize the visual context in multiple views.



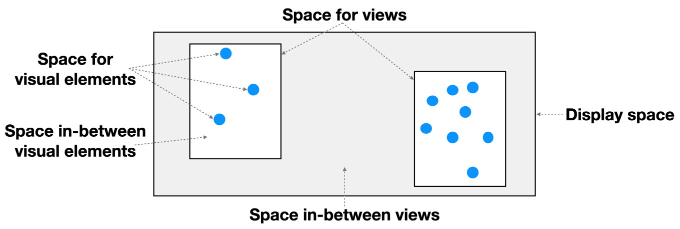


Fig. 4. Three conceptual layers (top) and five perceivable spaces (bottom) in multiple views.

Conceptual layers refer to spaces along the z-axis in a multi-view environment (Figure 4 top). We identify three conceptual layers: *screen space*, *views*, and *visual elements*. The *screen space* layer serves a canvas to hold multiple views. The layer of *views* is on top of the *screen space* layer, and is composed of all views that are shown. Finally, on top of the view layer, there is a layer of *visual elements*. It is home to all visual elements, regardless of which views they come from.

Within these conceptual layers, there are five **perceivable spaces**, which are areas in the xy-space in multi-view design (Figure 4 bottom). The *display space* allows users to add, remove, and organize visualizations (e.g., the workspace in the analyst's workstation [39]). This space is bounded by the available screen space in a system or tool that supports multiple views. The *space for views* refers to the space taken by the visualizations (e.g., the two white areas in Figure 4 bottom). The *space in-between views* refers to the space between views in the display space. It can be none if the space for views fully cover the display space. Within the space for views, there is the *space for visual elements* and the *space in-between visual elements*. Note that in some cases (e.g., space-filling visualizations), a visualization view can be full of visual elements (e.g., treemaps). In such cases, there is no space in-between visual elements.

4.2 Relations between Visual Context Components

The five perceivable spaces can be mapped to the three conceptual layers. The display space is in the screen space layer. The view and the visual element spaces belong to the layer of views and the layer of visual elements, respectively. The space in-between views is mapped to the screen space layer, and consists of areas not covered by the projections of view spaces from the layer of views. Similarly, the space in-between visual elements is in the view layer, and consists of areas not covered by the projections of visual elements on the top layer. Since the screen space works as a container for views and each view serves a container for its visual elements, we put the two in-between spaces in these containing layers. The three conceptual layers and five perceivable spaces offer usable, visual context of multiple views. For example, we can connect visual elements using visual links (e.g., lines) in the space in-between views, and further encode properties of the connections by varying appearances

TABLE 2
A summary of our identified design requirements for exploring cross-view data relationships.

Relationship Aspects	Design Requirements
Schema	R1: enabling users to specify a schema to construct cross-view data relationship R2: supporting users to modify an existing schema of cross-view data relationship
Structure	R3: displaying a cross-view data relationship structure R4: supporting users to change a cross-view data relationship structure
Weight	R5: displaying the detailed weight value for each individual-level relation inside a cross-view data relationship structure R6: displaying an overall weight value for a cross-view data relationship structure R7: supporting users to change the weight for each individual-level relation inside a cross-view data relationship structure
Size	R8: displaying many cross-view data relationship structures R9: supporting users to manage many cross-view data relationship structures

(e.g., color) of the visual elements. A clear understanding of the visual context helps visualization designers consider where to show cross-view data relationships.

5 Design for Cross-View Data Relationship

Based on the *data relationship framework* and the *visual context* of multiple views, we can systematically study the visualization design of cross-view data relationships. The data relationship framework characterizes which aspect(s) of the relationship a design may focus on. The visual context identifies available resources that a design may rely on. Thus, designing cross-view data relationship visualizations can be considered as: *using the available visual context of multi-view visualizations to present particular aspects of the relationship*.

5.1 Design Requirement Analysis

Based on the relationship framework discussed in Section 3, we derived a set of design requirements (Table 2) by considering possible user needs corresponding to each relationship aspect when exploring cross-view data relationships.

There are two major design requirements for schema: 1) **schema construction** and 2) **schema modification**. The former means that users need to specify the schema of cross-view data relations (e.g., a user selects six views to be considered in such relationships). The latter occurs when users change an existing schema. As the usage of different types of visualizations in multiple views is driven by user tasks, users do not always relate information from all displayed visualizations. For example, Emma decides to analyze relationships between people and locations (*schema construction*) from intelligence reports with Jigsaw [5]. She creates two scatterplot views for all the people and locations separately. When she selects a person's name, all locations that are in the same document get highlighted. Soon, she discovers several persons of interest and decides to focus on the relationship between these people (*schema modification*). She disables the synchronization feature between the two views. Now when she analyzes the relationship between suspects, nodes in the location view are not highlighted.

For structure, key design requirements include: 1) **structure display** and 2) **structure management**. To understand a structure, users need to view its components (e.g., data entities and their detailed relations). As there are multiple levels of relationships, users need to recognize each of them. For example, Emma noticed that three people mentioned in the intelligence reports all had records indicating a visit to the same four locations (bi-group relationship), and one location was related to three highly sensitive organizations (single-group relationship). In some cases (e.g., due to the relationship not matching the domain knowledge or a computation error), users need to revise a relationship structure. For example, Emma finds that one suspect uses an alias, so she wants to add that alias to all the related structures.

For weight, users need to understand two levels of information: 1) **detailed individual level** and 2) **overall structure level**. The former means that users need to understand the detail of a relationship structure. The latter indicates that users need a summative overview of a structure before digging into its detail. For a one-to-one relationship between visual elements (Table 1), both levels are the same. For higher-level structures, the two-level weights are different. While weight is not always considered when using multiple views (i.e., users only care about whether two entities are related or not), we argue that it is important to consider the design requirements for weights. First, computed relationships can have real-valued weights from probabilities, so designers should have the freedom to choose between an existential (0 or 1) or a more granular description (continuous values) of data relationships. Second, the overall structure level weight serves as a summative overview of multiple individual-level relationships. Thus, it is related to the design requirements of the other components in the data relationship framework. Moreover, at the detailed individual level, users may need to **change weight**. For example, a relationship between two people becomes more important as an analysis unveils more hidden clues. This can further influence the weights at the overall structure level. Furthermore, the computed weight is not always aligned with user domain knowledge. Visualizing such disparities is thus important for discovering new knowledge and making progress in an analysis.

For size, the key design requirement is to support users to see, explore and navigate many cross-view data relationships, instead of each individual relationship. For real-world problems, since it is rare that only one relationship structure exists, users need to *see and manage many relationships* for analysis. To fulfill this, users need to first organize data relationships by the type of relationship structures. Then, users may want to manage them based on their understanding. For example, after analyzing several person-location relationships and person-person relationships, Emma discovers that these relationships all point to a coordinated crime. She aggregates these two types of relationship structures and proceeds with further analysis.

In summary, we have identified 9 design requirements (Table 2). They offer a systematic and organized view of key user needs when exploring cross-view data relationships. It is important to note that they are not an exhaustive list of design requirements. It is also possible that one design does not meet all the requirements, since different analyses may focus on different aspects of cross-view data relationships. Following these requirements, we investigate the design for visualizing cross-view data relationships.

5.2 Method of Collecting Related Designs

To collect designs for visualizing cross-view data relationships, we recursively searched papers in related fields: information visualization, visual analytics and human-computer interaction. We defined a set of criteria to determine the relevance of a paper. Based on this, we collected a small group of closely related ones as seed papers, checked their references and papers that refer to them, and then used our criteria to find more relevant ones. We performed this recursively with a maximum recursion depth of 3. In case this search strategy favored relatively, highly cited papers, we also searched on Google Scholar with keywords: "coordinated multiple views", "brushing and linking", "small multiples", "dashboard visualization" and "set visualization" (we consider set visualizations as group-level relations that may overlap by sharing entities). For each keyword, we checked results from the first 10 pages, used our criteria to identify related papers and performed further recursive searches.

Our criteria for determining the relevance of a paper are:

- Visualizations should focus on a 2D layout, instead of in 3D or in an immersive environment.
- The work should have a software prototype that uses multi-view visualizations and clearly describe their design and interactive features. This ensures that a wide range of papers are initially included.
- The work should have key contributions to the design and usage of multi-view visualizations. This helps us identify a small and focused group of papers as seeds for further searches.
- The work should have an in-depth discussion about relating information from multi-view visualizations. This helps us exclude works simply using coordinated multiple views, brushing and linking, or small multiples, since they are common in many visual analysis tools. We consider a paper having an in-depth discussion, if it covers two or more of the following.
  - Specification of cross-view data relationships, including types of relationships and relationship structure.
  - Proposing new designs for exploring or relating data from multiple views.
  - Evaluation of designs or techniques for exploring or relating data from multiple views.
  - Lessons learned in the process of design and development of visual analysis systems in which relating data across multiple views is a key design concern.

Our search started with 8 closely-related seed papers: Improvise [40], Snap-Together [41], Cross-filtered Views [42], VisLink [43], Visual links across applications [44], ConnectedCharts [24], MyBrush [27] and Domino [29], and ended with 32 related papers in total. Improvise and Snap-Together highlight critical contributions to designing multiple coordinated views. Cross-filtered Views is a typical example of using brush and linking techniques to filter data across views. VisLink is a representative example that explicitly links visual elements across different views. Visual links across applications and MyBrush offer more flexibility in direct, visual linking by including more types of visualizations and allowing users to control the types of links, respectively. ConnectedCharts presents a design space of linking multiple bar-based charts and identifies two types of connections, so it is closely related to this work. Moreover, Domino supports users to interactively and progressively build connected multiform visualizations. It covers different levels of relationships by using a number of visual encodings besides simple lines.

While this is not an exhaustive list, it includes a variety of related works that support us to study possible designs for showing cross-view data relationships. Figure 5 summarizes them. Each row is a related work. Columns are categorized into three groups. We manually adjusted the order of rows to place examples with similar designs near each other.

5.3 Design for Schema Construction and Modification

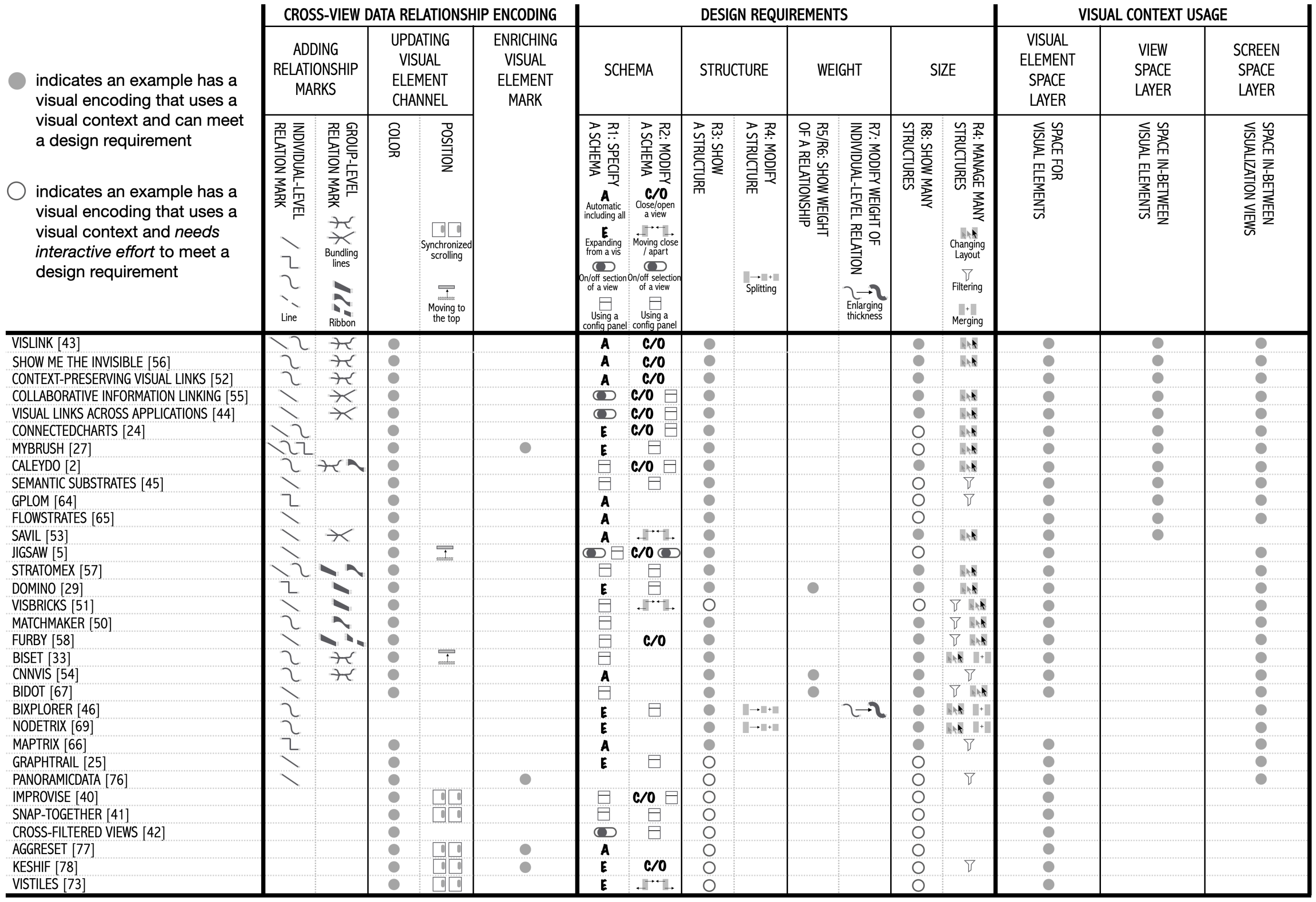
We identified four designs that support users to specify the schema of cross-view data relationships.

Automatically including all shown visualizations is the simplest way for users to manage cross-view data relationship schema. Users do not need any prior knowledge about data under investigation, and they can perform flexible explorations by trying different visualizations that an analysis tool supports. As users open or close a visualization, it is added to or removed from a schema automatically. Moreover, when users load a pre-defined set of visualizations (e.g., semantic substrates [45]), this design allows an analysis tool to automatically generate a fixed schema that include the whole set of visualizations.

Incrementally expanding from a visualization supports users to gradually build a schema. With a visualization currently under investigation, this design assumes that users want to follow the lead of identified, useful information in the current visualization, and perform further exploration. It allows users to request pulling in more relevant information by interacting with the visualization, and an analysis tool automatically finds and shows this in another visualization (e.g., Bixplorer [46], ConnectedCharts [24], Domino [29], and GraphTrail [25]). In such an exploratory process, a cross-view data relationship schema is incrementally specified. If users consider a newly added visualization useless and close it, it is removed from the schema.

Building while investigating a few visualizations provides a rich visual guidance for users to specify and modify cross-view data relationship schema. Different from incremental expanding, it allows users to see multiple visualizations and decide which of them to be included in a schema. Compared to expanding from one visualization, multiple visualizations offer users more guidance for decision-making. To support user decisions, each visualization has a widget (e.g., a checkbox or a switch button) to control its inclusion.

Creating a schema before showing any visualizations is a design that asks users to specify a schema first, and then related visualizations are shown accordingly. Compared to other designs, it requires users to have some knowledge



● indicates an example has a visual encoding that uses a visual context and can meet a design requirement

○ indicates an example has a visual encoding that uses a visual context and *needs interactive effort* to meet a design requirement

	CROSS-VIEW DATA RELATIONSHIP ENCODING					DESIGN REQUIREMENTS								VISUAL CONTEXT USAGE		
	ADDING RELATIONSHIP MARKS		UPDATING VISUAL ELEMENT CHANNEL		ENRICHING VISUAL ELEMENT MARK	SCHEMA		STRUCTURE		WEIGHT		SIZE		VISUAL ELEMENT SPACE LAYER	VIEW SPACE LAYER	SCREEN SPACE LAYER
	INDIVIDUAL-LEVEL RELATION MARK	GROUP-LEVEL RELATION MARK	COLOR	POSITION		R1: SPECIFY A SCHEMA	R2: MODIFY A SCHEMA	R3: SHOW A STRUCTURE	R4: MODIFY A STRUCTURE	R5/R6: SHOW WEIGHT OF A RELATIONSHIP	R7: MODIFY WEIGHT OF INDIVIDUAL-LEVEL RELATION	R8: SHOW MANY STRUCTURES	R4: MANAGE MANY STRUCTURES	SPACE FOR VISUAL ELEMENTS	SPACE IN-BETWEEN VISUAL ELEMENTS	SPACE IN-BETWEEN VISUALIZATION VIEWS
	╲ ┐ ∿ ⋱ Line	⋈ Bundling lines ▰ Ribbon		◫◫ Synchronized scrolling ⤒ Moving to the top		A Automatic including all; E Expanding from a vis; ⊙ On/off section of a view; ⊟ Using a config panel	C/O Close/open a view; ↔ Moving close / apart; ⊙ On/off selection of a view; ⊟ Using a config panel		▮→▪+▪ Splitting		∿→∿ Enlarging thickness		↖ Changing Layout; ▽ Filtering; ▪+▪ Merging			
VISLINK [43]	╲ ∿	⋈	●			A	C/O	●				●	↖	●	●	●
SHOW ME THE INVISIBLE [56]	∿	⋈	●			A	C/O	●				●	↖	●	●	●
CONTEXT-PRESERVING VISUAL LINKS [52]	∿	⋈	●			A	C/O	●				●		●	●	●
COLLABORATIVE INFORMATION LINKING [55]	╲	⋈	●			⊙	C/O ⊟	●				●	↖	●	●	●
VISUAL LINKS ACROSS APPLICATIONS [44]	╲	⋈	●			⊙	C/O ⊟	●				●	↖	●	●	●
CONNECTEDCHARTS [24]	╲ ∿		●			E	C/O ⊟	●				○	↖	●	●	●
MYBRUSH [27]	╲ ∿ ┐		●		●	E	⊟	●				○	↖	●	●	●
CALEYDO [2]	∿	⋈ ▰	●			⊟	C/O ⊟	●				●	↖	●	●	●
SEMANTIC SUBSTRATES [45]	╲		●			⊟	⊟	●				○	▽	●	●	●
GPLOM [64]	┐		●			A		●				○	▽	●	●	●
FLOWSTRATES [65]	╲		●			A		●				○		●	●	●
SAVIL [53]	╲	⋈	●			A	↔	●				●	↖	●	●	
JIGSAW [5]	╲		●	⤒		⊙ ⊟	C/O ⊙	●				○		●		●
STRATOMEX [57]	╲ ∿	▰ ▰	●			⊟	⊟	●				●	↖	●		●
DOMINO [29]	┐	▰	●			E	⊟	●		●		●	↖	●		●
VISBRICKS [51]	╲	▰	●			⊟	↔	○				○	▽ ↖	●		●
MATCHMAKER [50]	∿	▰	●			⊟		●				●	▽ ↖	●		●
FURBY [58]	╲	▰ ▰	●			⊟	C/O	●				●	▽ ↖	●		●
BISET [33]	∿	⋈	●	⤒		⊟		●				●	↖ ▪+▪	●		●
CNNVIS [54]	∿	⋈	●			A		●		●		●	▽	●		●
BIDOT [67]	╲		●			⊟		●		●		●	▽ ↖	●		●
BIXPLORER [46]	∿					E	⊟	●	▮→▪+▪		∿→∿	●	↖ ▪+▪			●
NODETRIX [69]	∿					E		●	▮→▪+▪			●	↖ ▪+▪			●
MAPTRIX [66]	┐		●			A		●				●	▽	●		●
GRAPHTRAIL [25]	╲		●			E	⊟	○				○		●		●
PANORAMICDATA [76]	╲		●		●			○				○	▽	●		●
IMPROVISE [40]			●	◫◫		⊟	C/O ⊟	○				○		●		
SNAP-TOGETHER [41]			●	◫◫		⊟	⊟	○				○		●		
CROSS-FILTERED VIEWS [42]			●			⊙	⊟	○				○		●		
AGGRESET [77]			●	◫◫	●	A		○				○		●		
KESHIF [78]			●	◫◫	●	E	C/O	○				○	▽	●		
VISTILES [73]			●	◫◫		E	↔	○				○		●		

Fig. 5. A summary of our identified designs based on the collected papers.

of a dataset being explored. It has been implemented as a configuration panel in an analysis tool (e.g., the snap specification dialog in Snap-Together [41]). Users rely on the panel to modify a schema, instead of interacting with views.

For schema modification, we have identified two key designs. One relies on displayed views. The other is independent from views. A *view-dependent* design requires users to interact with a view to determine whether a view is included in a schema or not (e.g., opening or closing a view, moving one view close to or away from another, or turning on or off of coordinating a view). A *view-independent* design allows users to use a configuration panel to modify a schema. A view-dependent design supports users to modify a schema while investigating a view, but the schema is not explicitly revealed. A view-independent design does show the schema but requires users to switch between visualizations and the configuration panel which may interfere with explorations.

5.4 Design for Structure Display and Management

We have identified three key design concepts to show cross-view data relationship structure: 1) *adding relationship marks*, 2) *updating channels of existing visual elements* and 3) *enriching marks of existing visual elements*. In this paper, *mark* and *channel* follow Munzner's definitions in [20]. These design concepts emphasize different visual encodings for relationship structure and need not be mutually exclusive, instead working together. Over 80% of our collected works used more than one of them. Figure 6 overviews them with six examples. (b)–(e) reflect the design concept (1) by adding different types of marks to encode cross-view relationships. (a) corresponds to the design concept (2) by using highlighting. (f) shows the design concept (3) by enriching marks for visual elements in a view. The examples are not exhaustive. Other visualizations may use other visual encodings.

5.4.1 Adding Relationship Mark

Adding relationship marks is the most straightforward design concept, as the marks directly encode cross-view data relationship structures. There are two types of relationship marks: *individual-level relationship mark* and *group-level relationship mark*. Both use the space in-between visual elements and the space in-between views. Figure 7 gives examples of possible designs for relationship marks, organized by rows. For each row, different columns present design alternatives.

An **individual-level relationship mark** is a line, which links visual elements from two views (the top row of Figure 7). Such a line can have multiple forms: straight, stepwise, curved or "broken" (as people can still perceive continuity based on the Gestalt principles [47]). Compared to straight lines, curved lines can improve perceived aesthetics [48] and stepwise lines with orthogonal edge routing can improve

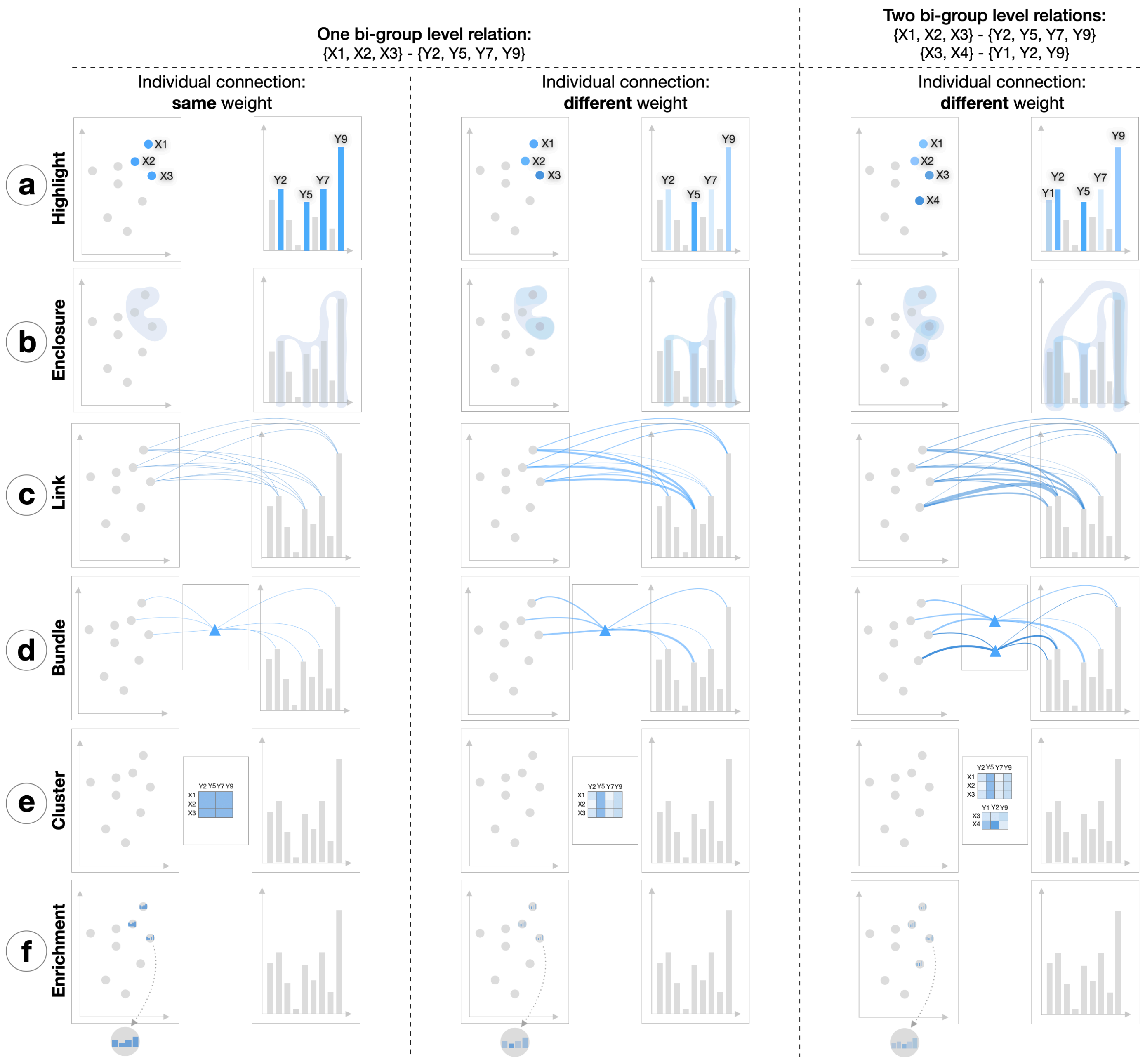

One bi-group level relation:
{X1, X2, X3} - {Y2, Y5, Y7, Y9}
Two bi-group level relations:
{X1, X2, X3} - {Y2, Y5, Y7, Y9}
{X3, X4} - {Y1, Y2, Y9}
Individual connection: same weight
Individual connection: different weight
Individual connection: different weight
a Highlight
b Enclosure
c Link
d Bundle
e Cluster
f Enrichment
X1
X2
X3
X4
Y1
Y2
Y5
Y7
Y9

Fig. 6. Examples of visual encodings for cross-view data relationships: (a) highlight, (b) enclosure, (c) link, (d) bundle, (e) cluster and (f) enrichment.

chart readability [49], but they use more pixels of the display space. While "broken" lines may produce fewer edge crossings than other forms, they require more user effort to identify connected visual elements due to the gaps.

The individual-level relationship mark can be used to reveal the same set of entities across multiple views. A number of them can lead to a perceived bi-group level relationship structure (Figure 6(c) left), but they can lead to visual clutter due to line crossings. When there are several bi-group level relationship structures, using a collection of lines cannot effectively show different structures. For example, in Figure 6(c) right, users need to trace lines and check shared entities to identify two different bi-group level structures. To better serve complex relationships, group-level relationship marks have been created and studied [50], [51], [52], [53].

The design of group-level relationship marks has three grouping strategies: 1) *relationship-grouping* (Figure 6(d) and Figure 7 row 2 and 3), 2) *entity-grouping* (Figure 6(b), and Figure 7 row 4 and 5) and 3) *grouping entities and relationships as a cluster* (Figure 6(e) and Figure 7 last row). They can show a bi-group level relationship structure, while one may work better than others in different cases (e.g., layouts of visual elements and availability of the space in-between views).

A **relationship-grouping** design depends on individual-level relationship marks. It emphasizes grouping individual relations, which is affected by the layout of visual elements in views. Based on whether visual elements are neighboring and aligned in the 2D space, there are two different designs: **bundling edges** and **using ribbons**. The former creates edge bundles (Figure 7 row 2) in the space in-between views [2],

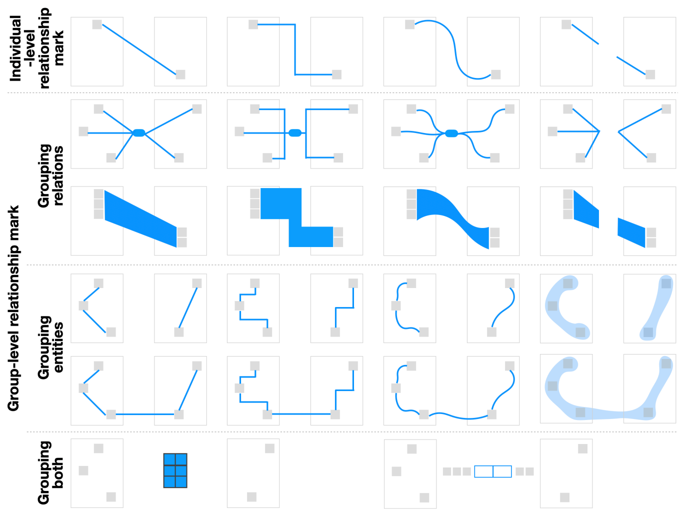


Fig. 7. Examples of possible designs for relationship marks.

[33], [54], or in the space in-between visual elements [55], [56]. The latter uses ribbons to link two groups of visual elements. Within a group, visual elements are neighboring and aligned (Figure 7 row 3). As the grouping is applied to sets of individual-level relations, the way that individual-level relationship marks are displayed can impact the design of group-level relationship marks. Thus, design variations exist when using the relationship-grouping strategy, such as bundling lines in different forms and using ribbons in multiple forms (e.g., straight [57], curved [50], [51] and "broken" [58]). Using ribbons can help reduce the number of shown lines, but it is not as flexible as edge bundling due to its requirement for the layout of visual elements.

An **entity-grouping** design highlights using marks to group visual elements of a bi-group level relationship, similar to set visualizations [38]. Such marks can be lines [59] or areas [60], [61], [62], [63], [64], which link or enclose visual elements of the same relationships. For a bi-group level relationship, if we consider its related visual elements as multiple sets, the marks group visual elements in the same view (Figure 7 row 4). If we consider them as one set, the marks group them all (Figure 7 row 5). Compared to relationship-grouping with edge bundling, grouping entities reduces the number of marks to be added, (e.g., comparing row 2 with row 4 and row 5 in Figure 7). However, it cannot reveal detailed cross-view connections between visual elements.

The strategy, **grouping both relationships and entities**, creates marks to aggregate and reveal both aspects of a relationship structure. Due to this aggregation, such marks locate in the space in-between views. Moreover, marks created with this strategy may duplicate related visual elements; while the other two do not make copies of visual elements. Possible designs of marks for this strategy includes matrix [65], [66] and BiDot [67] (related entities are placed on both sides of a box in which there are vertical lines indicating connections between entities).

Adding relationship marks has three benefits. First, it enables users to see relationships with newly added marks that are separated from existing visual element marks. Second, it can be flexibly adapted to different levels of relationships and include encodings to reveal the weight of a structure. Third, relationship marks can serve as handlers that support users to directly manipulate cross-view data relationships (e.g., dragging edge bundles [68] or matrices [69], and merging similar ones [70]). Moreover, adding relationship marks has two major drawbacks. For one thing, it may cause visual clutter (e.g., many edge crossings). For another thing, newly added relationship marks may impact user perception of existing visual elements in multiple views, especially those that overlap existing visual elements.

5.4.2 *Updating Channels of Visual Elements*

Different from adding relationship marks, this design relies on visual elements already displayed in multiple views. It emphasizes updating certain channels of visual elements for directing user attention to those of a relationship (Figure 6 (a)). Two channels are commonly used: **color** and **position**, which primarily work on the space for visual elements.

When using color, visual elements of cross-view relationships get highlighted (e.g., using different colors) [71], [72]. To reveal multiple relationship structures, color hue can be used. This highlight-oriented design preserves the layout of visual elements, and it helps users recognize visual elements of cross-view relationships. Moreover, as design alternatives, changing the size, shape or texture of visual elements can also make them visually salient. In addition, color and these listed design alternatives can also be used to show the same set of entities in multiple views.

When using position, the key design idea is to place related visual elements near each other. It includes changing user focus area or shifting positions of related visual elements in one view, as users interacting with another view. Examples of changing user focus area include synchronized scrolling (e.g., Snap-Together [41] and VisTiles [73]), and context + focus in multiple views [74]. Moreover, a typical example of shifting positions of related elements is Jigsaw's List View, which automatically moves related entities to the top of a list, when users select an entity in another list.

This design has two advantages. It avoids visual clutter caused by newly added marks and existing views are preserved. However, it has three issues. First, users can hardly see detailed cross-view links between visual elements. Second, it is hard for users to check weight of a relationship structure. Third, it is difficult for users to recognize different relationship structures. For example, in Figure 6 (a) right, if not informed, users may consider it one relationship, but actually it has two bi-group level of cross-view relationships.

5.4.3 *Enriching Marks of Visual Elements*

Enriching visual element marks places other marks (as guest marks) on top of existing visual element marks (as host marks). It only uses the space for visual elements, different from adding relationship marks with the entity-grouping strategy (since relationship marks use the space in-between visual elements). There are two types of enrichment: 1) **filled with visual element marks from another view** and 2) **transformed into a meaningful mini chart**. The former aims to reveal connections between a host mark and one or multiple guest marks with a containment spatial organization. The latter attempts to show in-depth detail of data encoded by a host mark. Figure 8 shows an example of them.

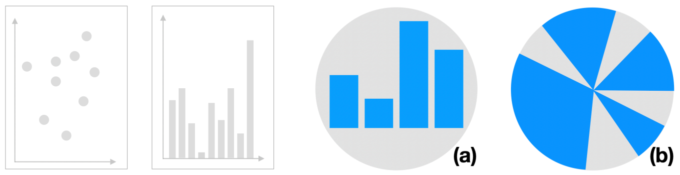


Fig. 8. Design examples of enriching visual element marks. (a) shows the design of filling a visual element mark in a scatterplot with related visual element marks from a histogram. (b) demonstrates the design of transforming a visual element mark in a scatterplot into a pie chart.

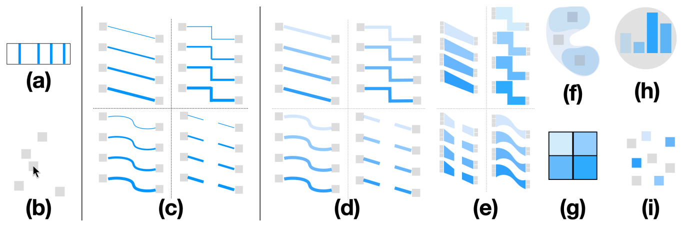


Fig. 9. Examples of visual encodings for relationship weight. (a) uses the channel of position on common scale. (b) is based on the position on unaligned scale channel. (c) encodes weight using 1D size, the thickness of lines. (d)-(i) uses color saturation to reveal weight.

Similar to the design of nested views in composite visualizations [75], guest marks in one view can be the same as related visual element marks in another view (Figure 8 (a)). It directly shows an individual-level relationship structure (between one guest mark and one host mark), or a single-group level relationship (between multiple guest marks and one host mark). Guest marks can be made by considering data that a host mark encodes [76], [77], [78]. In this case, based on cross-view data relations, it delves into details of the host and reveals it with a mini chart (Figure 8 (b)), rather than "borrow" visual element marks from another view.

Compared to adding relationship marks, this design avoids visual clutter caused by crossing lines or ribbons. It better shows details of a relationship structure than the design of updating channels of visual elements. When using guest marks, color saturation or luminance can be used to encode weight (Figure 6 (f) middle). Moreover, guest marks are placed on top of host marks, so the layout of existing visual elements from related views are well preserved. Also, since newly added marks are inside visual element marks of existing views, it helps to reduce user effort on handling context switching when exploring multiple views [10].

This design has three limitations. First, not all marks can be easily enriched, especially those taking up a small area, since guest marks need enough space to be held. Second, while the containment-based spatial organization matches individual-level and single-group level relationships, it is difficult for users to recognize different bi-group level relationships. If multiple bi-group level relationships are displayed with this design, users may not discriminate them. Third, newly added marks for enrichment may impact user perception of existing visualizations, especially when a large number of visual elements are enriched.

In summary, these three design concepts lead to a variety of designs to reveal cross-view data relationship structures. Since they can work together, multiple design choices from different design concepts can be applied together (e.g., using lines to link related visual elements and highlighting them).

5.5 Visual Encodings for Relationship Weight

Based on the three design concepts to visualize relationship structures, extra visual encodings can be applied to reveal weight. Weights of a relationship can be a set of numerical values or pre-defined categories (e.g., strong, medium and weak). Such information implies order, so it is reasonable to consider using magnitude channels [20].

Five magnitude channels can be applied on top of the encodings for cross-view relationship structures: **color saturation**, **color luminance**, **1D size**, **position on unaligned scale** and **position on common scale**. Based on perception studies, of the five channels, position on common scale is the most effective, and color saturation/luminance are the least effective [79]. Thus, if users need to learn details of weight, position on common scale is a top choice; while if users just need to roughly know weight, color saturation/luminance works. Figure 9 shows examples that use one of these channels based on the visual encodings of relationship structure, discussed in Section 5.4.

Four channels, except position on unaligned scale, can be used to show weight, when relationship structures are encoded by newly added relationship marks. Specifically, we can use thickness (1D size) or color saturation/luminance of a line to show individual-level relationship weight (Figure 9 (c) and (d)), and use color saturation/luminance of ribbons, area marks, and matrix cells to display weight of a bi-group level relationship (Figure 9 (e), (f) and (g)). In addition, to precisely show weight of a bi-group level relationship, we can used the design of BiDot [67]. It uses a box as a common scale, which holds a set of vertical lines. Each line indicates an individual-level relationship, and its horizontal position in the box encodes weight (Figure 9 (a)).

Three channels (position on unaligned scale, color saturation, and color luminance) support weight displaying, when using the design of updating channels of visual elements to show relationship structures. Relationship weight can be revealed as proximity of relative distance between visual elements [36], [39], [80] (Figure 9 (b)). Also, different color saturation/luminance of visual elements can indicate different relationship weights (Figure 9 (i)). Similarly, when relationship structure is displayed with the design of enriching visual element marks, the color saturation/luminance of guest marks is useful to present weight.

5.6 Interactions for Handling Many Relationships

Enabling user interactions is critical to support users to see and manage many relationship structures. The enabled user interactions are complementary to selected visual encodings for relationship structures.

For the design of adding relationship mark, newly added marks can offer handlers for users to directly manipulate relationships. Possible interactions include: selecting or hovering a relationship mark to check its related visual elements, selecting multiple relationship marks for filtering and comparison, moving relationship marks to spatially organize them, dragging one relationship mark on top of another to merge them, and splitting a relationship mark. User can also

Designs for Each Relationship Aspect				Visual Context Usage	Pros	Cons
Constructing **schema**	Automatically including all displayed visualizations			- Space for views - Space for visual elements	- A schema is automatically built as users open visualizations. - Users can flexibly explore visualizations without paying much attention to schema.	- Not all displayed visualizations should be included in a schema. - Useless visualizations cannot be removed from a schema unless they are closed.
	Incrementally expanding from a visualization				- A schema can be progressively built as user analyses proceed. - Schema construction follows previously identified, useful information.	- Users have to put effort on schema construction. - A schema cannot be explicitly shown, instead revealed by a few involved views. - Users need to check involved views to understand a constructed schema.
	Building while investigating visualizations				- Schema construction is performed after users investigate multiple visualizations. - Users can flexibly decide which visualizations should be included in a schema.	
	Creating a schema before showing any visualizations			- Space in-between views	- A schema is explicitly displayed in a schema configuration widget. - Schema construction is separated from visualizations.	- Users need to have some prior knowledge about the data being investigated. - Users cannot rely on displayed visualizations for schema construction.
Changing **schema**	View-dependent designs (e.g., using a widget on a view)			- Space for views - Space for visual elements	- Users can modify a schema while investigating a view. - Users can flexibly decide which visualizations should be included in a schema.	- A schema cannot be explicitly shown, instead revealed by a few involved views. - It requires users to work on visualization views to change a schema.
	View-independent designs (e.g., using a configuration panel)			- Space in-between views	- A schema can be explicitly displayed in a schema configuration widget. - Users can focus on using a configuration widget for schema modification.	- It needs some extra UI widget. - It requires users to switch attention between visualizations and schema configuration widgets.
Displaying one or multiple structure(s) **(structure and size)**	Adding relationship mark	Individual-level relationship mark		- Space for visual elements - Space in-between visual elements - Space in-between views	- It can reveal multiple levels of relationship structures. - It can clearly show detailed connections and weight of a relationship structure.	- Users have to aggregate individual-level relationship marks for group-level relationships. - It can cause visual clutter, especially when many relationship structures exist. - It impacts user perception of existing views.
		Group-level relationship mark	Grouping entities		- It can reveal members of a relationship structure. - It can reveal different group-level relationship structures.	- It cannot clearly show detailed connections of a relationship. - It can cause visual clutter, especially if there are many relationship structures. - It impacts user perception of existing views.
			Grouping relationships		- It helps reduce the number of displayed individual-level relationship marks. - It can serve as handlers for users to manipulate relationship structures.	- It heavily relies on individual-level relationship marks. - It can cause visual clutter, especially when many relationship structures exist. - It impacts user perception of existing views.
			Grouping both	- Space in-between views	- It separates relationships from existing views, so existing views are preserved. - It can potentially provide an overview of relationships.	- It requires that existing views do not fully cover the display space. - It requires users to switch attention between visualization views and the area that shows relationships.
	Updating channels of visual elements - Using color - Using position			- Space for visual elements	- It avoids possible visual clutter caused by adding new marks. - It avoids affecting existing views by adding new marks. - It helps reduce user effort on context switching when exploring multiple views.	- It cannot clearly show detailed connections of a relationship structure. - It is difficult for users to check weight of a relationship structure. - It is hard for users to visually separate different relationship structures.
	Enriching marks of visual elements - Filled with marks from another view - Transformed into a meaningful mini chart				- It directly shows individual-level and single-group level relationships. - It is possible for users to get in-depth details of visual elements that are enriched. - It preserves the layout of visual elements in existing visualizations. - It helps reduce user effort on context switching when exploring multiple views.	- Not all visual marks can be easily enriched, especially those with a small area. - It is hard for users to discriminate multiple bi-group level relationship structures. - Newly added marks impact user perception of existing visualizations.
Handling many structures **(size)**	- Interacting with added relationship marks (e.g., select, hover, drag and move)			- Space for visual elements - Space in-between visual elements - Space in-between views	- It allows users to check the detail of relationship structures. - It supports users to recognize and compare different relationship structures. - It supports users to modify (e.g., merge and split) relationship structures.	- It requires user effort on managing newly added relationship marks. - It requires users to switch attention between relationship marks and related visual elements in exiting views.
	- Interacting with existing visual elements (e.g., brushing and linking)			- Space for visual elements	- It allows users to check the detail of relationship structures. - It supports users to recognize and compare different relationship structures.	- It requires many trial-and-error attempts to see different relationship structures. - Relationships are implicitly revealed as sets of visual elements.
Displaying **weight**	Encoded based on relationship marks - color saturation/luminance - 1D size (e.g., line thickness) - position on common scale			- Space for visual elements - Space in-between visual elements - Space in-between views	- It can reveal weight of relationship structures in detail. - It allows users to compare detailed individual-level weights within relationship structures, especially when using the encoding of position on common scale.	- The color and size based encodings can cause visual clutter, especially when many individual relationship marks exist. - The encoding of position on common scale may need to add extra visual marks to precisely reveal the value of weight.
	Encoded without relationship marks - color saturation/luminance - position on unaligned scale (e.g., spatial proximity)			- Space for visual elements	- It avoids possible visual clutter caused by adding new marks. - It preserves visual elements in existing visualizations.	- It takes users interactive effort to check weight of relationship structures. - As weight is roughly revealed, it is hard for users to check weight in detail. - It is hard for users to compare weight of relationship structures. - The overall weight of relationship structures cannot be clearly revealed.

Fig. 10. A summary of our analyzed design options corresponding to the four descriptive relationship aspects. The grey section regards *schema*. The yellow section considers *structure* and *size*. The blue section corresponds to *weight*.

interact with visual elements, such as selecting or hovering a visual element to check its related relationships.

For the other two design concepts, without relationship marks, enabling user interactions on visual elements helps to address the problem that multiple relationship structures cannot be clearly revealed. Brushing and linking is a typical example of user interactions used, especially when no relationship marks are added. For example, as users select visual elements, related ones get highlighted. If the highlight can be accumulated and users are allowed to make multi-selections, it is possible for users to recognize different structures by iteratively selecting visual elements and checking highlighted ones. Besides enabling users to recognize different relationship structures, brushing and linking helps to avoid visual clutter caused by added relationship marks and preserve existing views. Moreover, brushing and linking supports highlighting the same set of entities in multiple views (e.g., exploring nodes in a scatterplot matrix) based on one-to-one mapping.

6 Design Recommendations

We make a set of design recommendations based on the analysis of designs for cross-view data relationships, summarized in Figure 10. In summary, we suggest considering: 1) structure, weight and size of cross-view data relations, 2) complementary designs, 3) available in-between space, 4) creating an overview of cross-view data relationships, and 5) existing visualization views. The first two regard the descriptive aspects of relationships. Others consider usable visual context of multiple views.

6.1 Making Data-Driven Design Decisions

It is important to make data-driven design decisions. Given the framework, discussed in Section 3, structure, weight and size are the three aspects that can impact design decisions. Key impacting factors related to them are: level of relationship structures, variance of weight values, and the number of relationship structures to be displayed.

Conducting pre-design analyses of cross-view data relationships to gain an in-depth understanding of these factors helps make reasonable design decisions. It is possible that not all levels of structures are present in a dataset, and not all of them may be useful for user analysis tasks. Thus, a clear understanding of important levels of relationship structures helps narrow the design options down to a focused group.

It is necessary to check values for weight (e.g., binary or quantitative) and their variance. For example, if bi-group level relations are the most important and weight is binary, we consider the design of adding group-level relationship marks. As the variance is low, we can focus on the design of grouping relations (e.g., bundling lines). In contrast, if the weights have a high variance, lines may be a poor option and we need to consider other designs like matrices.

The size of cross-view data relationship impacts design decisions. When there is a single relationship, as is shown in Figure 6, all designs clearly show it. However, for multiple relationships, not all designs allow users to effectively recognize different ones. When users need to explore a large number of cross-view data relationships (which is common for real-world problems), of the three design concepts, adding relationship marks is a better choice than the others.

6.2 Considering Complementary Design Options

Choose design options that can be complementary to each other, instead of relying on a single design, especially when exploring more than one relationship aspect. Complementary means that multiple design options can benefit each other to reveal desired aspects of cross-view data relations (e.g., adding individual-level relationship marks and using color saturation on these marks to show relationship structure and weight). It is hard to find one design that meets all design requirements. As discussed before, each design option has its own advantages and drawbacks to reveal the structure, weight and size of cross-view data relationships. However, they can work together and be complementary to one another. Based on our analysis of collected designs, a majority made combining multiple options in their work.

To identify complementary designs, a clear understanding of analysis tasks is critical (e.g., which aspects of cross-view data relationships that users care about). For example, if users need to check and compare a large number of bi-group level relationships in detail, we can choose designs from adding relationship marks and updating visual element channels, and apply them together, such as using matrices to show relations, encoding weight as the color saturation of cells in matrices, and enabling user manipulation on matrices to make comparisons between relations.

6.3 Considering Trade-offs of The In-Between Space

Regarding the in-between space (e.g., the space in-between views or visual elements), we need to consider two trade-offs. The first considers the availability of the in-between space. It is determined by the layout of views or visual elements (i.e., fixed v.s. adjustable). A fixed layout (e.g., each view is placed next to each other) may limit the availability of the in-between space, since users cannot move anything to make room for the in-between space. While an adjustable layout is more flexible, not all analysis tasks need it (e.g., finding outliers in a scatterplot matrix). Moreover, layout management takes extra user effort. Thus, we need to consider user analysis tasks to decide whether to provide the capability of enabling the in-between space.

The second trade-off regards the usage of the in-between space, when it is available. It highlights reserving the in-between space for cross-view linking v.s. leaving it as empty. When using it for cross-view linking (e.g., by adding relationship marks), a key advantage is that users can directly see relationships. It potentially transforms the in-between space to a view that holds relationships. However, it has two drawbacks. First, the added cross-view linkings can cause visual clutter (e.g., many lines and line crossings). Second, the displayed cross-view linkings in the in-between space can impact user perception of existing views. When leaving the in-between space as empty, the display space is primarily used for existing views, which are preserved. However, the cost for this is that users have to rely on visual elements in existing views to explore cross-view relationships.

6.4 Creating An Overview

If possible, create an overview of cross-view data relations, as it can guide user explorations [37]. Based on a clear understanding of relationships that users need to investigate,

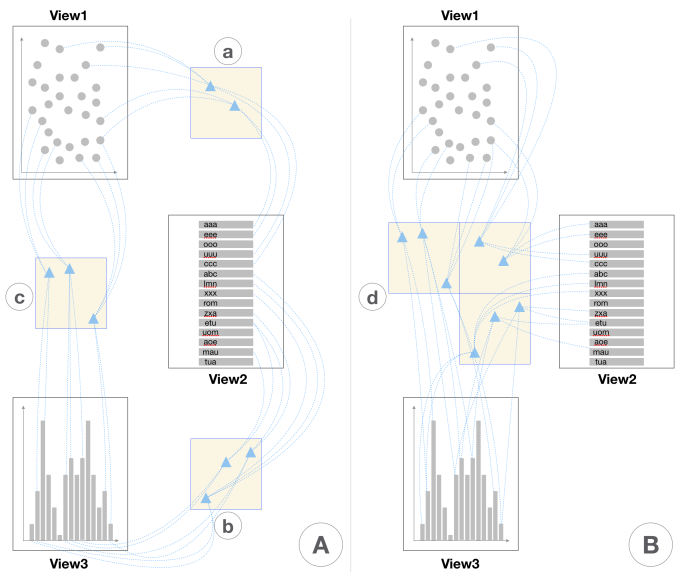


Fig. 11. Two possible designs of overviews of cross-view data relationships: multiple separated stand-alone overviews (A) and a centralized stand-alone overview (B). Note: (a)-(d) present stand-alone overviews where visual elements indicate cross-view data relationships, and blue lines reveals connections between relationships with their members.

it is possible to compute them and display the results as an overview. Since the design concepts of updating visual element channels and enriching visual element marks rely on displayed visual elements, showing an overview on top of that is not a good choice. As discussed before, it is hard for users to distinguish different relationships. Moreover, it may confuse users that an overview of cross-view data relations is shown as a collection of existing visual elements.

Instead, the design of adding relationship marks in the in-between space potentially supports creating an overview of cross-view data relationships. With available in-between space, a stand-alone view can be created, which holds newly added relationship marks together. Such a view serves an overview of cross-view data relationships. Since computed relationships and existing visual elements are separated into different views, existing views are well-preserved and user perception of them are not interfered by newly add marks. Moreover, such an overview helps address Emma's problem while exploring cross-view data relations, discussed in Section 1, as she has more visual guidance, rather than manually track highlighted visual elements in different views.

Such an overview has two possible designs (Figure 11). When there are more than two views, we can create either multiple stand-alone overviews for pairwise views, or one centralized stand-alone overview for all views. Relationship marks in such overviews can be linked to related visual elements via lines and interactive highlighting. Moreover, it is possible to compute the layout of relationship marks and encode their similarity with proximity of relative distance, which supports users to explore similar relationships.

6.5 Considering Visualizations Used in Multiple Views

Make design decisions by considering visualizations used in multiple views from two aspects: 1) whether they are the same type and 2) how they and their visual elements are spatially organized. They both impact design choices. For example, as is shown in Figure 7, when visual elements are spatially aligned, we can replace a collection of line-based relationship marks with one ribbon. However, when visual elements are organized in a graph and do not neighbor each other, we need to avoid using ribbons.

Considering the types of visualizations used in multiple views (i.e., same v.s. different) may bring additional design options. When visualizations used in multiple views are the same type (e.g., a collection of matrices and multiple lists of entities), we may consider some specific interaction techniques to support users to explore cross-view data relations, besides the design discussed in Section 5.4. For example, it is possible to allow users to drag and move one matrix on top of another and such an overlay supports users to see their differences [81]. However, if visualizations are of different types, this overlay design may not work (e.g., users can get confused by overlaying a line chart on top of a graph).

The spatial organization of multiple views needs attention. It determines the availability of the space in-between views. Moreover, whether it allows users to flexibly manage the positions of multiple views significantly impacts design decisions. For example, if the layout of multiple views is fixed and they are juxtaposed with each other (e.g., MyBrush [27]), there is no space in-between views. Thus, we should not try to design the stand-alone overview, mentioned before. However, if the layout of multiple views can be flexibly changed, it is possible to make the space in-between views available (e.g., by dragging and moving views), which leads to considering more design choices.

7 Discussion

7.1 Compared to Designs for Composite Visualizations

A key goal of our design considerations is to help visualization researchers and practitioners leverage and display cross-view data relationships of items across multiple views for supporting data analysis. Earlier, Javed et al. [75] presented a design space called composite visualization views (CVVs), which concerns visually showing data relationships among the same or different visualizations within one visualization view. Both CVVs and our design considerations address the design challenge of visualizing data relationships among multiple views. Specifically, both design approaches present a set of designs, which are coupled with available visual representations and depict the correlation between visual elements in two or more visualizations.

However, there are key differences between CVVs and our design considerations. CVVs focus on the design for combining or coordinating multiple visualizations within a single visualization view. While CVVs cover multiple-view visualizations, they deal with multi-view visualizations in which the layout of multiple views is fixed and coordinated (e.g., [27], [45] and [82]). In contrast, we consider the availability of empty space (called the "in-between space") among views or visual elements and the possibility of preserving the context of individual views, with cross-view visual representations. In our design considerations, it is possible to change the layout of views (e.g., dragging and moving views), instead of a fixed layout. These specific characteristics of our design considerations offer potential paths to further improve visual analysis applications.

7.2 Towards the Design for Using Multiple Displays

The in-between space and draggable views enable users to flexibly arrange the positions of views, so they can impact design decisions. This aspect of our design considerations facilitates a greater leap into physical space or visual analytics tools that emphasize the formation of insight via the usage of large displays (i.e., expanded visual space). Our design considerations suggest different ways for users to arrange views into certain layouts based on their relationships (e.g., using alignment, clustering or ordering). In so doing, the spatial organization of views facilitates the development of more cohesive insights, based on the spatial position of a group of independent views on the screen [39].

The in-between space serves a useful resource to hold visual encodings for data relationships among views, even when using multiple displays. Based on our analysis of existing designs, we note that the use of in-between space allows for holding newly added relationship marks (e.g., lines, ribbons and edge bundles). These relationship marks can potentially construct another conceptual layer of data relationships, thereby further extending the context of multiple views on multiple displays. Particularly, this aspect of our design considerations can address challenges associated with visual analysis using multiple displays. In a multi-display environment (MDE), information and tasks may often be scattered and disconnected among physically separated displays. Thus, an inherent design challenge associated with visual analysis is relating and synthesizing information across separate displays [53]. In this regard, the relationship marks present various strategies for connecting information and visual objects among views on different displays. For instance, to visualize relationships among items on multiple displays, we can add a new display that holds relationship marks among items in multiple displays (Figure 6 (e)). Additionally, inspired by the group-level relationship marks, we can also consider approaches for showing relationships of data items on separate displays corresponding to one or more visual links from a source to multiple targets across multiple displays.

In summary, making design decisions by considering the in-between space and separate view context is particularly important, if large displays and MDEs are used for visual analysis and sensemaking [73], [83]. In such analysis environments, users need appropriate strategies to understand and relate information distributed among separate displays and more seamlessly transition among different views on multiple displays during the insight formation process [84].

8 Conclusion

We discuss systematic design considerations for visualizing cross-view data relationships, with regards to both descriptive relationship aspects and usable visual context of multiple views. Based on these, the visualization design of cross-view data relationships can be understood as encoding desired relationship aspects by using available visual context of multiple views. We have analyzed a variety of design options, and made five design recommendations.

Despite proposing a systematic view of design considerations for cross-view data relationships, this work has four limitations. First, our notion of data relationship assumes relational data, but not all data in visualizations is relational, or even discrete. Thus, the designs that we have examined and discussed do not cover them. Second, we do not consider using multiple views in a 3D or immersive environment, which may lead to a broader set of design considerations with more advanced user interactions (e.g., ImAxes [85] and [86]). Third, our search of relevant designs is based on published papers by using known exemplars and keyword search. Due to the small size of this group, even with recursive searches, our collected paper list is not exhaustive, potentially omitting relevant work. Last, while we have studied a variety of designs and analyzed their pros and cons, the evaluation of our discussed design considerations is weak. Conducting user studies and expert interviews will be helpful to further evaluate these design considerations. Moreover, similar to cross-validation, performing case studies by using our presented design considerations to analyze existing multi-view visualizations, besides our collected ones, can also help validate our design considerations. However, we believe this work can inspire and guide future studies to further formalize a design space of visualizing cross-view data relationships.

ACKNOWLEDGMENTS

This research was supported in part by the NSF grant IIS-2002082 / IIS-1850036, and the NSERC Discovery Grant.

REFERENCES

[1] J. Zhao, "Interactive visual data exploration: A multi-focus approach," Ph.D. dissertation, Department of Computer Science, University of Tornoto, Jul 2015.

[2] A. Lex, M. Streit, E. Kruijff, and D. Schmalstieg, "Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context," in *IEEE Pacific Visualization Symposium*. IEEE, 2010, pp. 57–64.

[3] M. Sun and G. Convertino, "Interver: Drilling into categorical-numerical relationships," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, 2016, pp. 44–51.

[4] H. Zhang, M. Sun, D. Yao, and C. North, "Visualizing traffic causality for analyzing network anomalies," in *Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics*, 2015, pp. 37–42.

[5] J. Stasko, C. Görg, and Z. Liu, "Jigsaw: supporting investigative analysis through interactive visualization," *Information visualization*, vol. 7, no. 2, pp. 118–132, 2008.

[6] J. Zhao, F. Chevalier, E. Pietriga, and R. Balakrishnan, "Exploratory analysis of time-series with chronolenses," *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2422–2431, 2011.

[7] C. North and B. Shneiderman, "A taxonomy of multiple window coordination," Tech. Rep., 1997.

[8] J. C. Roberts, "Exploratory visualization with multiple linked views," in *Exploring geovisualization*. Elsevier, 2005, pp. 159–180.

[9] J. Roberts, "State of the art: Coordinated & multiple views in exploratory visualization," in *International Conf. on Coordinated & Multiple Views in Exploratory Visualization*. IEEE, 2007, pp. 61–71.

[10] 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*. ACM, 2000, pp. 110–119.

[11] J. S. Yi, Y. ah Kang, and J. Stasko, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1224–1231, 2007.

[12] N. Boukhelifa, J. C. Roberts, and P. J. Rodgers, "A coordination model for exploratory multiview visualization," in *Proceedings of International Conference on Coordinated and Multiple Views in Exploratory Visualization*. IEEE, 2003, pp. 76–85.

[13] T. Pattison and M. Phillips, "View coordination architecture for information visualisation," in *Proceedings of the 2001 Asia-Pacific Symposium on Information Visualisation-Volume 9*. Australian Computer Society, Inc., 2001, pp. 165–169.

[14] R. Burtner, S. Bohn, and D. Payne, "Interactive visual comparison of multimedia data through type-specific views," in *Visualization and Data Analysis 2013*, 2013, p. 86540M.

[15] "In-spire," https://in-spire.pnnl.gov/.

[16] "Tableau software," https://www.tableau.com/.

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

[18] D. Park, S. M. Drucker, R. Fernandez, and N. Elmqvist, "Atom: A grammar for unit visualizations," *IEEE Transactions on Visualization and Computer Graphics*, 2017.

[19] S. Huron, R. Vuillemot, and J.-D. Fekete, "Visual sedimentation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2446–2455, 2013.

[20] T. Munzner, *Visualization analysis and design*. CRC press, 2014.

[21] E. H.-h. Chi, "A taxonomy of visualization techniques using the data state reference model," in *IEEE Symposium on Information Visualization*. IEEE, 2000, pp. 69–75.

[22] J. C. Roberts, "Multiple view and multiform visualization," in *Visual Data Exploration and Analysis VII*, vol. 3960. International Society for Optics and Photonics, 2000, pp. 176–186.

[23] S. Knudsen and S. Carpendale, "View relations: An exploratory study on between-view meta-visualizations," in *Proceedings of 9th Nordic Conference on Human-Computer Interaction*, 2016, pp. 1–10.

[24] C. Viau and M. J. McGuffin, "Connectedcharts: explicit visualization of relationships between data graphics," in *Computer Graphics Forum*, vol. 31, no. 3pt4, 2012, pp. 1285–1294.

[25] C. Dunne, N. Henry Riche, B. Lee, R. Metoyer, and G. Robertson, "Graphtrail: Analyzing large multivariate, heterogeneous networks while supporting exploration history," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 1663–1672.

[26] M. C. Chuah and S. F. Roth, "On the semantics of interactive visualizations," in *Proceedings IEEE Symposium on Information Visualization'96*. IEEE, 1996, pp. 29–36.

[27] P. Koytek, C. Perin, J. Vermeulen, E. André, and S. Carpendale, "Mybrush: Brushing and linking with personal agency," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 605–615, 2018.

[28] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-lite: A grammar of interactive graphics," *IEEE Trans. on Visualization & Computer Graphics*, vol. 23, no. 1, pp. 341–350, 2017.

[29] S. Gratzl, N. Gehlenborg, A. Lex, H. Pfister, and M. Streit, "Domino: Extracting, comparing, and manipulating subsets across multiple tabular datasets," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2023–2032, 2014.

[30] J. Zhao, M. Glueck, F. Chevalier, Y. Wu, and A. Khan, "Egocentric analysis of dynamic networks with egolines," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2016, pp. 5003–5014.

[31] M. Sun, C. North, and N. Ramakrishnan, "A five-level design framework for bicluster visualizations," *IEEE Transactions on Visualization & Computer Graphics*, vol. 20, no. 12, pp. 1713–1722, 2014.

[32] H. Wu, M. Sun, P. Mi, N. Tatti, C. North, and N. Ramakrishnan, "Interactive discovery of coordinated relationship chains with maximum entropy models," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 12, no. 1, pp. 1–34, 2018.

[33] M. Sun, P. Mi, C. North, and N. Ramakrishnan, "Biset: Semantic edge bundling with biclusters for sensemaking," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 310–319, 2016.

[34] S. Cheng and K. Mueller, "The data context map: Fusing data and attributes into a unified display," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 121–130, 2016.

[35] C. North, "Toward measuring visualization insight," *IEEE computer graphics and applications*, vol. 26, no. 3, pp. 6–9, 2006.

[36] A. Endert, P. Fiaux, and C. North, "Semantic interaction for visual text analytics," in *Proc.of the SIGCHI Conf. on Human Factors in Computing Systems*. ACM, 2012, pp. 473–482.

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

[38] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers, "Visualizing sets and set-typed data: State-of-the-art and future challenges," in *Eurographics Conference on Visualization (EuroVis)–State of The Art Reports*, 2014, pp. 1–21.

[39] C. Andrews, A. Endert, and C. North, "Space to think: large high-resolution displays for sensemaking," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010, pp. 55–64.

[40] C. Weaver, "Building highly-coordinated visualizations in improvise," in *IEEE Symposium on Information Visualization*. IEEE, 2004, pp. 159–166.

[41] 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*. ACM, 2000, pp. 128–135.

[42] C. Weaver, "Cross-filtered views for multidimensional visual analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 192–204, 2010.

[43] C. Collins and S. Carpendale, "Vislink: Revealing relationships amongst visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1192–1199, 2007.

[44] M. Waldner, W. Puff, A. Lex, M. Streit, and D. Schmalstieg, "Visual links across applications," in *Proceedings of Graphics Interface*. Canadian Information Processing Society, 2010, pp. 129–136.

[45] B. Shneiderman and A. Aris, "Network visualization by semantic substrates," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 733–740, 2006.

[46] P. Fiaux, M. Sun, L. Bradel, C. North, N. Ramakrishnan, and A. Endert, "Bixplorer: Visual analytics with biclusters," *Computer*, vol. 46, no. 8, pp. 90–94, 2013.

[47] M. Wertheimer, "Gestalt theory." 1938.

[48] C. Ware, H. Purchase, L. Colpoys, and M. McGill, "Cognitive measurements of graph aesthetics," *Information visualization*, vol. 1, no. 2, pp. 103–110, 2002.

[49] S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Hola: Human-like orthogonal network layout," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 349–358, 2015.

[50] A. Lex, M. Streit, C. Partl, K. Kashofer, and D. Schmalstieg, "Comparative analysis of multidimensional, quantitative data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1027–1035, 2010.

[51] A. Lex, H.-J. Schulz, M. Streit, C. Partl, and D. Schmalstieg, "Visbricks: multiform visualization of large, inhomogeneous data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2291–2300, 2011.

[52] M. Steinberger, M. Waldner, M. Streit, A. Lex, and D. Schmalstieg, "Context-preserving visual links," *IEEE Trans. on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2249–2258, 2011.

[53] H. Chung and C. North, "Savil: cross-display visual links for sensemaking in display ecologies," *Personal and Ubiquitous Computing*, pp. 1–23, 2017.

[54] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Trans. on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 91–100, 2016.

[55] M. Waldner and D. Schmalstieg, "Collaborative information linking: Bridging knowledge gaps between users by linking across applications," in *IEEE Pacific Visualization Symposium*. IEEE, 2011, pp. 115–122.

[56] T. Geymayer, M. Steinberger, A. Lex, M. Streit, and D. Schmalstieg, "Show me the invisible: visualizing hidden content," in *Proceedings of the 32nd annual ACM Conference on Human Factors in Computing Systems*. ACM, 2014, pp. 3705–3714.

[57] A. Lex, M. Streit, H.-J. Schulz, C. Partl, D. Schmalstieg, P. J. Park, and N. Gehlenborg, "Stratomex: visual analysis of large-scale heterogeneous genomics data for cancer subtype characterization," in *Computer graphics forum*, vol. 31, no. 3pt3, 2012, pp. 1175–1184.

[58] M. Streit, S. Gratzl, M. Gillhofer, A. Mayr, A. Mitterecker, and S. Hochreiter, "Furby: fuzzy force-directed bicluster visualization," *BMC bioinformatics*, vol. 15, no. S6, p. S4, 2014.

[59] B. Alper, N. Riche, G. Ramos, and M. Czerwinski, "Design study of linesets, a novel set visualization technique," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2259–2267, 2011.

[60] C. Collins, G. Penn, and S. Carpendale, "Bubble sets: Revealing set relations with isocontours over existing visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1009–1016, 2009.

[61] N. H. Riche and T. Dwyer, "Untangling euler diagrams," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1090–1099, 2010.

[62] K. Dinkla, M. J. van Kreveld, B. Speckmann, and M. A. Westenberg, "Kelp diagrams: Point set membership visualization," in *Comp. Graphics Forum*, vol. 31, no. 3pt1, 2012, pp. 875–884.

[63] W. Meulemans, N. H. Riche, B. Speckmann, B. Alper, and T. Dwyer, "Kelpfusion: A hybrid set visualization technique," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 11, pp. 1846–1858, 2013.

[64] J.-F. Im, M. J. McGuffin, and R. Leung, "Gplom: the generalized plot matrix for visualizing multidimensional multivariate data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2606–2614, 2013.

[65] I. Boyandin, E. Bertini, P. Bak, and D. Lalanne, "Flowstrates: An approach for visual exploration of temporal origin-destination data," in *Computer Graphics Forum*, vol. 30, no. 3, 2011, pp. 971–980.

[66] Y. Yang, T. Dwyer, S. Goodwin, and K. Marriott, "Many-to-many geographically-embedded flow visualisation: an evaluation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 411–420, 2017.

[67] J. Zhao, M. Sun, F. Chen, and P. Chiu, "Bidots: Visual exploration of weighted biclusters," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 195–204, 2018.

[68] M. Sun, J. Zhao, H. Wu, K. Luther, C. North, and N. Ramakrishnan, "The effect of edge bundling and seriation on sensemaking of biclusters in bipartite graphs," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 10, pp. 2983–2998, 2018.

[69] N. Henry, J.-D. Fekete, and M. J. McGuffin, "Nodetrix: a hybrid visualization of social networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1302–1309, 2007.

[70] M. Sun, D. Koop, J. Zhao, C. North, and N. Ramakrishnan, "Interactive bicluster aggregation in bipartite graphs," in *2019 IEEE Visualization Conference (VIS)*. IEEE, 2019, pp. 246–250.

[71] D. Hienert, B. Zapilko, P. Schaer, and B. Mathiak, "Vizgr: linking data in visualizations," in *International Conf. on Web Information Systems and Technologies*, 2011, pp. 177–191.

[72] N. Boukhelifa and P. J. Rodgers, "A model and software system for coordinated and multiple views in exploratory visualization," *Information Visualization*, vol. 2, no. 4, pp. 258–269, 2003.

[73] 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*, vol. 24, no. 1, pp. 626–636, 2018.

[74] S. Bjork and J. Redstrom, "Redefining the focus and context of focus+ context visualization," in *IEEE Symposium on Information Visualization*. IEEE, 2000, pp. 85–89.

[75] W. Javed and N. Elmqvist, "Exploring the design space of composite visualization," in *Visualization Symposium (PacificVis), 2012 IEEE Pacific*. IEEE, 2012, pp. 1–8.

[76] E. Zgraggen, R. Zeleznik, and S. M. Drucker, "Panoramicdata: Data analysis through pen & touch," *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2112–2121, 2014.

[77] M. A. Yalcin, N. Elmqvist, and B. B. Bederson, "Aggreset: Rich and scalable set exploration using visualizations of element aggregations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 688–697, 2016.

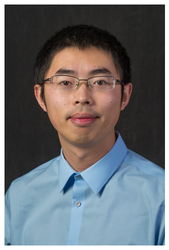
[78] M. A. Yalçın, N. Elmqvist, and B. B. Bederson, "Keshif: Rapid and expressive tabular data exploration for novices," *IEEE Transaction on Visualization and Computer Graphics*, 2017.

[79] J. Heer and M. Bostock, "Crowdsourcing graphical perception: using mechanical turk to assess visualization design," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010, pp. 203–212.

[80] C. Andrews and C. North, "Analyst's workspace: An embodied sensemaking environment for large, high-resolution displays," in *IEEE Conference on Visual analytics Science and Technology*. IEEE, 2012, pp. 123–131.

[81] R. Sadana, T. Major, A. Dove, and J. Stasko, "Onset: A visualization technique for large-scale binary set data," *IEEE Transaction on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1993–2002, 2014.

[82] J. Fekete, D. Wang, N. Dang, A. Aris, and C. Plaisant, "Overlaying graph links on treemaps," in *Proceedings of the IEEE Symposium on Information Visualization Conference Compendium (InfoVis' 03)*. IEEE, 2003, pp. 82–83.
[83] H. Chung, C. North, S. Joshi, and J. Chen, "Four considerations for supporting visual analysis in display ecologies," in *IEEE Conference on Visual Analytics Science and Technology*. IEEE, 2015, pp. 33–40.
[84] H. Chung, C. North, J. Z. Self, S. Chu, and F. Quek, "Visporter: facilitating information sharing for collaborative sensemaking on multiple displays," *Personal and Ubiquitous Computing*, vol. 18, no. 5, pp. 1169–1186, 2014.
[85] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott, "Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation," in *Proc. of ACM Symposium on User Interface Software and Technology*. ACM, 2017, pp. 71–83.
[86] A. Prouzeau, A. Lhuillier, B. Ens, D. Weiskopf, and T. Dwyer, "Visual link routing in immersive visualisations," in *Proceedings of the ACM International Conference on Interactive Surfaces and Spaces*, 2019, pp. 241–253.



Maoyuan Sun is an assistant professor with the Department of Computer Science, Northern Illinois University. His research falls in the areas of visual analytics, information visualization, human-computer interaction and human-centered machine learning, with a variety of applied domains, including STEM education, intelligence analysis, business intelligence and cyber security.

Akhil Namburi is a graduate student with the Department of Computer Science, Northern Illinois University. His research focuses on creating usable, interactive user interface for supporting sensemaking of big data.

David Koop is an assistant professor with the Department of Computer Science, Northern Illinois University. His research concerns provenance, data science, and visualization.

Jian Zhao is an assistant professor in the Cheriton School of Computer Science at the University of Waterloo, where he directs the WatVis (Waterloo Visualization) group. His research interests include information visualization, human-computer interaction, and data science. His work contributes to the development of advanced interactive visualizations that promote the interplay of human, machine, and data.

Tianyi Li is an Assistant Professor with the Department of Computer Science, Loyola University Chicago. She designs and develops systems for computer-supported cooperative work. Her most recent research is about crowdsourced sensemaking, to scaffold collective intelligence of novice crowds for tasks such as intelligence analysis.

Haeyong Chung is Assistant Professor with the Department of Computer Science, the University of Alabama in Huntsville. His research focuses on the design and development of visual analysis techniques and tools that leverage mixed displays including mobile devices and tiled display walls.