

# Exploring the Design Space of Composite Visualization

Waqas Javed\*

Niklas Elmqvist†

Purdue University



Figure 1: Four different visual composition operators (from the left): juxtaposition, superimposition, overloading, and nesting.

## ABSTRACT

We propose the notion of composite visualization views (CVVs) as a theoretical model that unifies the existing coordinated multiple views (CMV) paradigm with other strategies for combining visual representations in the same geometrical space. We identify five such strategies—called CVV design patterns—based on an extensive review of the literature in composite visualization. We go on to show how these design patterns can all be expressed in terms of a design space describing the correlation between two visualizations in terms of spatial mapping as well as the data relationships between items in the visualizations. We also discuss how to use this design space to suggest potential directions for future research.

**Index Terms:** H.5.1 [Information Systems]: Multimedia Information Systems—Animations; H.5.2 [Information Systems]: User Interfaces; I.3 [Computer Methodologies]: Computer Graphics

## 1 INTRODUCTION

While the design space of visual representations is far from exhausted, it is clear that it is becoming increasingly difficult to develop entirely novel visual representations that significantly extend the existing vocabulary of such representations in our field. It is also clear that there is generally no visual representation that is obviously superior for a given dataset; all visual representations have strengths and weaknesses. In recent years, efforts have been made towards **combining** different visualizations to balance these strengths and weaknesses. This also addresses novelty: new visual representations can be generated by combining existing ones.

However, there exists many ways to combine two or more visualizations in a single space. One common approach is *coordinated multiple views* (CMV) [31], where the visualizations are often juxtaposed in the same space and coordinated using some form of linking mechanism. However, there exist many examples where multiple visualizations are combined in other ways than CMV-style juxtaposition. For example, the NodeTrix [17] technique combines adjacency matrices inside a node-link diagram, SparkClouds [21] overlays a temporal visualization over tag clouds, and semantic substrates [34] connect nodes in different views using links. These examples show that juxtaposition, used for many CMV-based visualization systems, is not an isolated approach to combining multiple visualizations, but that there exists a spectrum of different patterns for composing visualizations. However, although these examples are discussed in the literature, there is no formal characterization that organizes these in the same way as for CMV.

\*e-mail: wjaved@purdue.edu

†e-mail: elm@purdue.edu

In this paper, we identify the design space of *composite visualization views* (CVVs) that allows us to combine multiple visualizations in the same visual space. As a starting point, we survey the literature of composite visualization and find five general design patterns for how existing work merges two different visualizations into one: juxtaposition, integration, overloading, superimposition, and nesting. Some of these patterns are already known and formally recognized; for example, juxtaposition gives rise to the CMV composition pattern, where views are simply placed next to each other. Other design patterns have so far not been formally defined in the literature, but we try to highlight each pattern with examples. We then use these patterns to define a design space that captures the salient aspects of composite visualization. We proceed to use this design space to suggest avenues for future research.

## 2 COMPOSITE VISUALIZATION VIEWS

We define a *composite visualization* as the *visual composition* of two or more *visual structures* in the same *view*. In this definition, we use the following concepts from Card et al. [9]’s pipeline:

- *visual composition*: the combination (placement or arrangement) of multiple visual objects;
- *visual structure*: the mapping from data to visual form (i.e., the result of a visualization technique);
- *view*: the physical display space (most often 2D) where a visual structure is rendered.

The nature of the composition governs the resulting type of composite visualization. As we shall see in this paper, composite visualizations are relatively common. However, only one type of composite visualization—coordinated multiple views (CMV) [4, 32, 40], where the visual composition is often a juxtaposition—is formally recognized as a visualization design strategy in the literature.

Composite visualizations are used primarily for situations where a single visualization is not sufficient because of high complexity, large scale, or heterogeneous data [31]. In these situations, displaying data in several different ways may benefit user cognition. For example, the same file system hierarchy could be visualized in both a treemap [20] as well as a radial layout (such as Sunburst [35]), each representation allowing the user to focus on different aspects of the data. Furthermore, different types of data have varying representation affinities. For example, locations are best represented in a geospatial visualization, whereas multidimensional data fit best in a parallel coordinate plot [18] or a scatterplot matrix [10].

### 2.1 Method

Our approach in this work is to derive a design space of composite visualization based on the literature of visualization techniques where several visual structures are combined in the same view. We

collect such composite visualizations using literature searches and prior experience. We then let existing work inform our model by organizing this prior art into rough categories that emerge from the characteristics of the techniques. In later sections, we discuss each category in more detail. Finally, we construct a design space that captures all aspects of these composite visualization techniques.

## 2.2 Visual Composition

The method for *visual composition* is an emerging theme when surveying composite visualizations in the literature. In other words, the different ways of composing two visualizations *A* and *B* in the same visual space seems to be a useful organizing principle in this domain. Based on the literature, we derive the four visual compositions (Figure 1) that give rise to four rough categories—we call them *CVV design patterns*—for composing visualizations:

- **Juxtaposition** → **Juxtaposed Views**: Placing visualizations side-by-side in one view (Coordinated Multiple Views [32]);
- **Superimposition** → **Superimposed Views**: Overlaying two visualizations in a single view;
- **Overloading** → **Overloaded Views**: Utilizing the space of one visualization for another; and
- **Nesting** → **Nested Views**: Nesting the contents of one visualization inside another visualization.

In addition, another emergent CVV design pattern is to juxtapose visual structures, but to add graphical objects such as arrows, dotted lines, or glyphs to visually link one view with another. We therefore think this method deserves a design pattern of its own:

- **Integration** → **Integrated Views**: Placing visualizations in the same view with visual links.

## 2.3 Design Patterns

Identifying and characterizing composite visualization views (CVVs) as a unified design approach not only allows us to explore this space in a structured fashion, but also provides a method for comparing the effectiveness of different designs. The reason we use the term *design pattern* [15] here is that these are high-level approaches where the actual composition generally differs on a case-by-case basis. This is consistent with the notion of a design pattern as a general and reusable solution to a common problem.

We should also note that these design patterns are very different from the software design patterns for visualization proposed by Heer and Agrawala [16]. The latter deal with software engineering design aspects, whereas our CVV patterns are defined on a visual design level. While the pattern movement is popular in software engineering, the reader should note that design patterns first were proposed by Alexander et al. [2] for urban planning, and so our use of the concept is in fact closer to its original spirit.

Below we describe the five rough categories of composite visualization that we identified in the literature. In each section, we first describe each pattern and then give a couple of in-depth examples of representative composite visualization techniques. These examples are not intended to be exhaustive, but to be illustrative of practical implementations of each pattern.

## 2.4 Existing Formalisms

Using multiple views for visualization is not a new concept, and early examples date back to the beginnings of the field [27]. Baldonado et al. [4] gave general guidelines on the use of multiple views in information visualization, and North and Shneiderman [30, 28, 29] discussed relational models for achieving this.

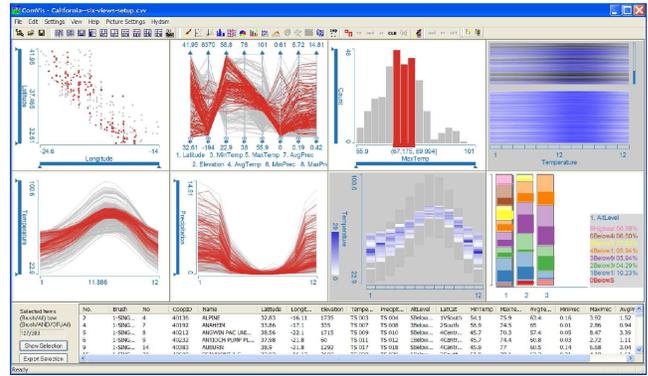


Figure 2: **ComVis** [24] (Juxtaposed Views). Meteorology data.

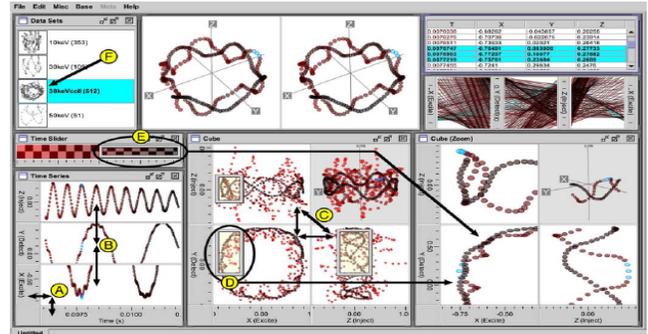


Figure 3: **Improvise** [39] (Juxtaposed Views). Juxtaposed views are used to explore the simulated ion trajectory in a cubic ion trap.

These discussions were later formalized into the concept of *coordinated multiple views* (CMV) [31, 32], where multiple views of different visualizations are combined in visual space and are implicitly linked together, often using brushing [5].

In their work on multiple and explicitly linked visualizations, Collins et al. [11] discuss the formalization of multi-relation visualizations, in the process deriving three different techniques for this practice. Their formalism is related to our work but of a preliminary nature, lacks the discussion of some of the design patterns discussed here, and also does not identify CVVs as a unified approach.

## 3 JUXTAPOSITION → JUXTAPOSED VIEWS

Juxtaposed views (Figures 2 and 3) are the most prominent—and probably the most flexible and easy to implement—design pattern for composing visualizations in a single view [4, 28, 31, 33]. The design pattern is based on juxtaposing multiple visualizations side by side. Any linking between visualizations is implicit, i.e., it is not a part of the visual representation. Examples include brushing [5], synchronized scrolling [27], and synchronized drill-down [23].

The effectiveness of juxtaposed views has been an important research topic. North and Shneiderman presented a taxonomy [29] of such visualization. They showed that a well-designed juxtaposed view increases user performance while exploring relations among multiple data dimensions. However, designing effective juxtaposed views can be a challenging task and requires efficient relational linking and spatial layout. Weaver’s cross-filtered views [41] addresses this by abstracting the relations between the views to make defining, implementing, and reusing them easier.

There currently exists a large number of visualization tools based on juxtaposed views in the literature; e.g. [3, 7, 36]. Below we review two such tools that are representative of these.

### 3.1 ComVis

ComVis [24] is a multidimensional visualization system supporting multiple coordinated views for exploring complex datasets (Figure 2). The dataset is shown in the form a table view at the bottom of the main window. Beyond basic interactions, ComVis also support interactive brushing using both single and composite brushes.

Figure 2 shows a visual exploration of meteorology data using ComVis. The user has created eight different views, each with a different visualization. The analyst has then used a single brush to select three bins in the histogram view, causing all the other views to highlight the corresponding data items.

### 3.2 Improvevis

Improvise [39, 40] is a visualization framework based on the juxtaposed views design pattern. The framework allows users to build and browse multiple visualizations while coordinating relational linking among them. The system is highly extensible and modularized, allowing it to be adapted for virtually any type of data and visual representation. To explore relational data in an interactive manner, Improvise provides support for coordinated queries, a visual abstraction language designed for relational databases. More recent work on cross-filtered views [41] adds to the expressive power of the framework for relation linking between different views.

Figure 3 shows a visual exploration of a simulated ion trajectory in a cubic ion trap using Improvise. The tool allows user to visualize different portions of the data set, selected using dynamic queries [1]. All the visualizations are coordinated and data selection in one view is projected in all others.

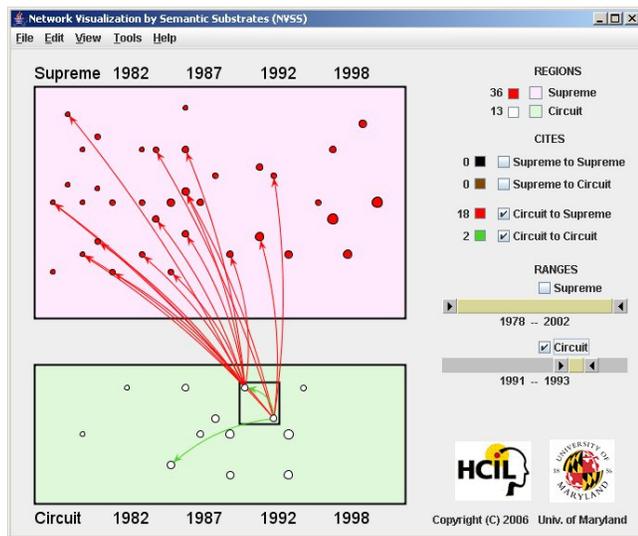


Figure 4: **Semantic Substrates** [34] (Integrated Views). Network visualization of a dataset of court cases using semantic substrates.

## 4 INTEGRATION → INTEGRATED VIEWS

The integrated views design pattern is also based on juxtaposing (or tiling) the component visualizations (Figures 4, 5). For this reason, the visual composition for integrated views is identical to that of juxtaposed views. However, contrary to the implicit linking used in juxtaposed views, integrated views use explicit linking, normally in the form of graphical lines that relate data items in different views another [11]. One prominent example of integrated views is Charles Minard’s famous visualization of Napoleon’s march on Moscow [37], where explicit linking shows the relations between temperature and the number of surviving soldiers during the retreat.

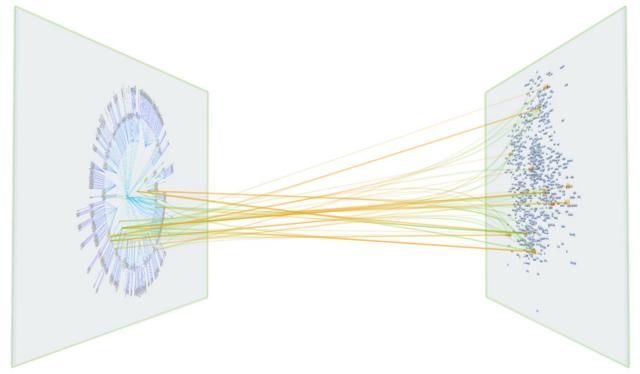


Figure 5: **VisLink** [11] (Integrated Views). Radial and force-directed graphs on separate visualization planes linked with visual edges.

The use of explicit linking in integrated views, compared to implicit linking in juxtaposed views, allows for better relational cognition, but at the cost of added visual clutter. However, as the number of data points increases in the visualizations, the visual clutter arising from the explicit links may become a major hindrance. Commonly used strategies to avoid this problem are to aggregate the links, or to show relational links only for selected data values [11].

### 4.1 Semantic Substrates

Shneiderman and Aris [34] proposed a network visualization layout based on a user-defined semantic substrate with node-links diagram as an underlying visualization (Figure 4). Semantic substrates are spatially non-overlapping regions that are built to hold nodes based on some category present in the dataset. The individual regions are sized proportionally to the number of data entries for the category they visualize. This scheme allows users to get a quick idea about the cardinality of different categories present in the underlying dataset. Their approach is in line with the integrated view design pattern because the techniques add visual links to connect the nodes in different substrates. To reduce clutter arising from the links, the tool allows for toggling their visibility.

Figure 4 shows semantic substrates used for the exploration of a subset of federal judicial cases on the legal issue of regulatory takings from 1978 to 2005. The nodes in different views are placed based on their chronological order along the horizontal axis and links among the nodes highlight citation between different cases.

### 4.2 VisLink

VisLink [11] (Figure 5) creates multiple 2D planes, one for each visualization, and shows relational linking between the different visualization planes. Visualization planes generated in VisLink are interactive and users can re-position them in the view to explore data relations. In contrast with semantic substrates, VisLink allows the use of different visualizations while exploring the dataset.

As with semantic substrates, the VisLink relational linking is done using visual lines that connect visual marks in one plane with the corresponding mark in the other plane. To reduce the inherent occlusion due to the explicit relational links between visualizations, the tool supports two kinds of edges: straight edges are used to show one-to-one linking, while bundled curved edges are used to highlight one to many linking. To reduce visual clutter the tool shows relational links only between adjacent planes, and the planes must be reordered for the user to see relations between other planes. Figure 5 shows VisLink being used for exploring a dataset of English words based on the IS-A relation over synonym sets.

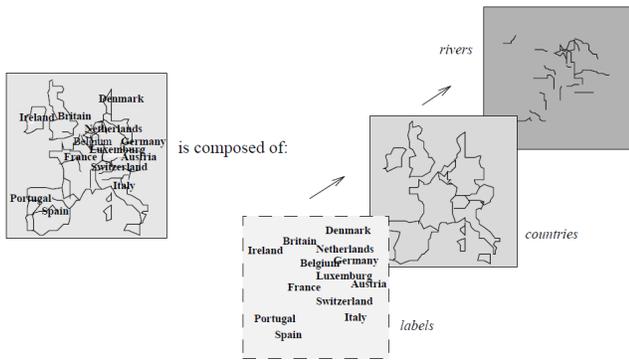


Figure 6: **Mapgets** [38] (Superimposed Views). Presentation stack, with superimposed layers for rivers, borders, and labels, in Mapgets.

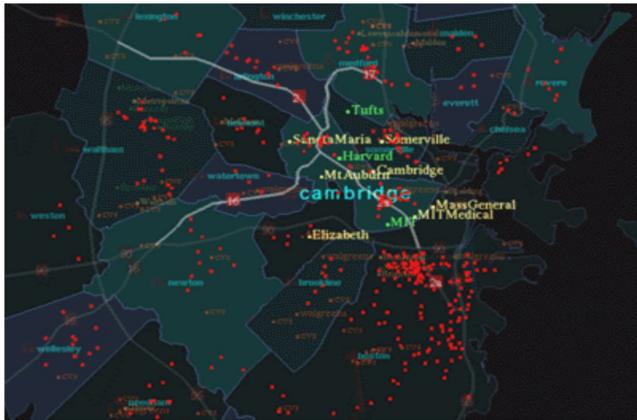


Figure 7: **GeoSpace** [22] (Superimposed Views). A crime data layer superimposed on a geographical map of the Cambridge, MA area.

## 5 SUPERIMPOSITION → SUPERIMPOSED VIEWS

Superimposed views overlay two or more visual spaces on top of each other (Figures 6 and 7). The resulting visualization becomes the visual combination of the component visualizations, often using transparency to enable seeing all views. Superimposed views are generally used to highlight spatial relations in the component visualizations. In other words, the spatial linking present in these views is one-to-one, i.e., all the overlay visualizations share the same underlying visual space. Line graph visualizations with several data series, where more than one graph is superimposed in a single chart (e.g., [19]), is a very commonly used example of this design pattern.

The spatial linking in the superimposed views allows for easy comparison across different datasets because the user does not have to split their attention between different parts of the visual space. Furthermore, the fact that visualizations are stacked means that they can each use the full available space in the view. However, because the composition simply adds the component visualizations together, the visual clutter may become significant, and it is also likely to cause conflicts arising from one visualization occluding another.

### 5.1 Mapgets

Mapgets [38] is a geographic visualization system that allows users to interactively perform map editing and querying of geographical datasets. The maps generated using Mapgets are built on an underlying presentation stack that superimposes multiple dataset layers on top of each other. The users can dynamically select the dataset

to use for each layer and the total number of layers to compose. Different layers in the presentation stack allow users to independently interact with each of the associated visualization and control the layer attributes. The technique also allows the users to reorder layers in the presentation stack to achieve the desirable map result.

Figure 6 shows an example of a European map generated in Mapgets. The presentation stack associated with this map consists of three layers: the bottom layer visualizes rivers, the center layer is used to depict the country borders, and the topmost layer is used to display the country labels.

### 5.2 GeoSpace

GeoSpace [22] allows users to interactively explore complex visual spaces using superimposed views. It permits progressively overlaying different datasets, based on the user queries, in a single view. Beyond allowing users to explore datasets through dynamic queries, GeoSpace also supports pan and zoom operations for navigation.

Figure 7 shows GeoSpace system being used for exploring crime around the Cambridge, MA area. The figure shows a 2D view of the visualization, where red dots that are spatially coupled to the underlying layer show the reported crime cases in the region.

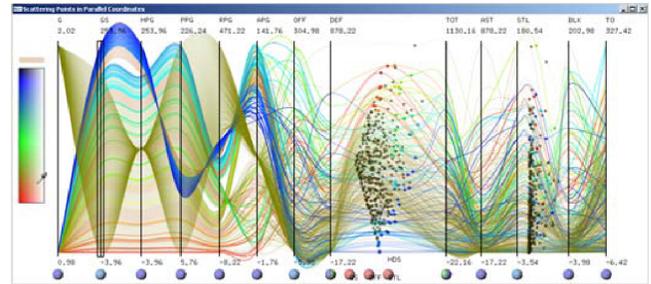


Figure 8: **SPPC** [45] (Overloaded Views). This tool overloads points into the region bounded by two axes in the parallel coordinate plot.

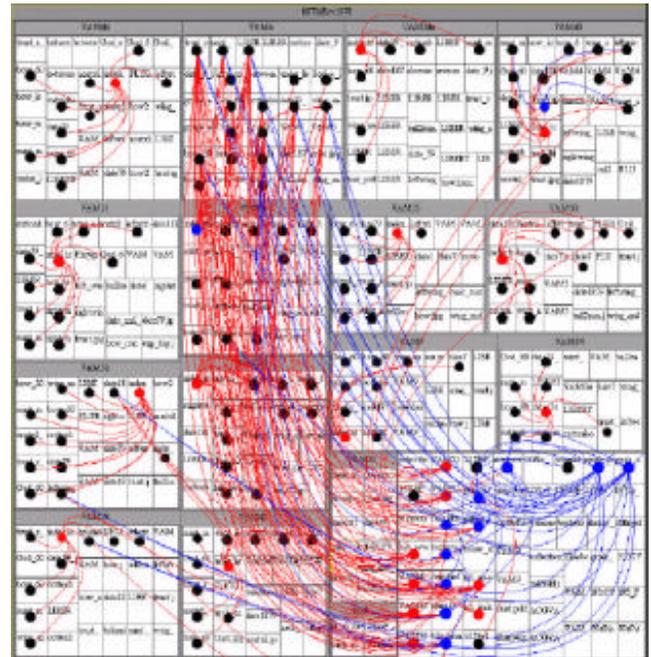


Figure 9: **Links on treemaps** [14] (Overloaded Views). The tool identifies a tree structure in a graph and visualizes it using a treemap.

## 6 OVERLOADING → OVERLOADED VIEWS

This design pattern characterizes compositions where one visualization, called the *client* visualization, is rendered inside another visualization, called the *host*, using the same spatial mapping as the host [26]. Overloaded views (Figures 8 and 9) are similar to superimposed views, but with some important differences. Like superimposition, the client visualization in this design pattern is overlaid on the host. However, unlike Superimposed Views, there exists no one-to-one spatial linking between the two visualizations [12].

While previous design patterns have all operated on specific *views* of component visualizations, overloaded views (and also the next pattern, Nested Views) operate on the visual structure themselves. In other words, it is no longer possible to merely use visual layout operations to organize the views together, but the visual structures themselves must be modified to combine the components. We will see examples of this below.

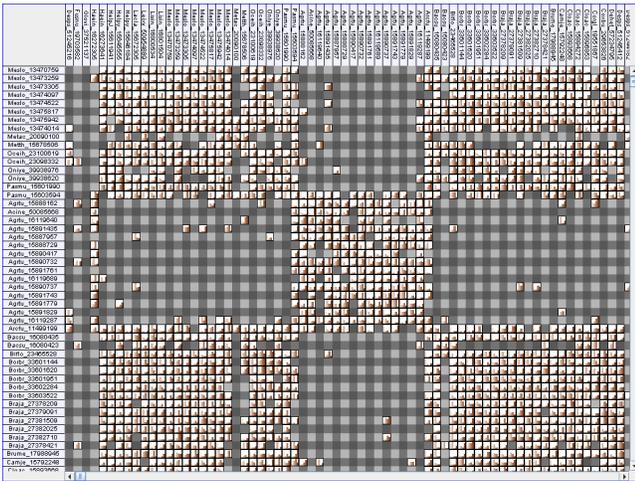


Figure 10: **ZAME** [13] (Nested Views). Visual exploration of a protein-protein interaction dataset in ZAME.

### 6.1 Scatter Plots in Parallel Coordinates (SPCC)

Yuan et al. [45] presented a system that allows overloading of 2D scatterplots on a parallel coordinates visualization [18] (Figure 8). The technique is based on converting the space between pairs of selected coordinate dimensions in a parallel coordinate plot into scatterplots through multidimensional scaling [42]. The technique takes advantage of the fact that parallel coordinate plots do not really use the space **between** the parallel dimensional axes, which means that this space is open for being overloaded.

SPCC is also an example of combining two techniques to compensate for their individual shortcomings. Parallel coordinates are efficient for visualizing multiple dimensions in a compact 2D visual representation. However, they make it hard to correlate trends across multiple dimensions due to their inherent visual clutter. Scatterplots, on the other hand, provide an effective way of correlating trends in any dimension of a dataset [10]. Combining both techniques allows for sharing their advantages.

### 6.2 Graph Links on Treemaps

Fekete et al. [14] proposed a technique for rendering graphs using a treemap [20] with overloaded graph links. The idea is based on the fact that it is possible to decompose a graph into a tree structure and a set of remaining graph edges that are not included in the tree. This graph decomposition allows for using a treemap to visualize the tree structure, and then overload links corresponding to the remaining graph edges on the treemap visualization. Even though Fekete et al.

call this “overlying”, the technique is an example of overloading in our terminology because the graph links are not just a separate layer on top of the treemap, but they are embedded into the visual structure of the treemap and use the node positions as anchors.

Figure 9 shows the technique being used to visualize a website. Here, the directory structure, inherent in any website, is visualized through an underlying treemap and external links are visualized through overlaid edges. The overlaid edges are not straight lines, but are curved to highlight source and target locations. The edges are curved more near the source, hence making it easy to visually recognize the direction of the link. The tool also supports controlling the visibility of various edges to reduce visual clutter, and coloring edges based on their attributes.

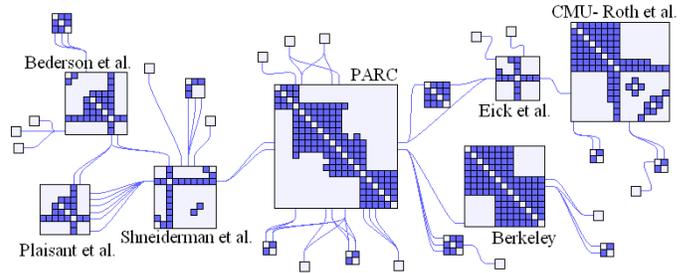


Figure 11: **NodeTrix** [17] (Nested Views). This example shows a visualization of the InfoVis co-authorship network.

## 7 NESTING → NESTED VIEWS

Nested views, like overloaded views, are also based on the notion of host and client visualizations. However, in this design pattern, one or more client visualizations are nested **inside** the visual marks of the host visualizations, based on the relational linking between the points. Most often, the nesting is performed simply by replacing the visual marks in the host visualization by nested instances of the client visualization (Figures 10 and 11). An example of this would be a scatterplot where the individual marks are barchart glyphs [25].

The nested views pattern provides an effective way of relating data points in the host visualization to the data visualized through the client visualizations. Again the users need not divide their attention between multiple views, and the host visualization is allowed to use the full available space. However, since the design pattern embeds one or more visualizations inside a visual mark, the client visualizations are allocated only a small portion of the host visualization’s visual space, and zooming and panning may be required to see details. Furthermore, just like overloading, nested views compose the actual visual structures of the components, which typically requires a more careful design.

One issue to discuss here is the difference between overloading and nesting. These are different design patterns because nesting simply replaces the visual marks of the host with the visual structure of the client, whereas overloading requires a much more integrated composition of the visual structures of the host and the client.

### 7.1 ZAME

Nested views are becoming increasingly prominent for visualizing large-scale datasets using glyph-based methods. ZAME [13], a visualization system designed to explore large-scale adjacency matrix graph visualization, uses this approach. The base matrix representation used in ZAME is a hierarchical aggregation of the underlying dataset. The tool allows the user to zoom in data space, which amounts to drilling-down and rolling-up in the aggregation hierarchy to see more or less details. Abstract glyphs representing aggregated data for each cell in the matrix are nested inside the visual marks of the matrix to convey information about the aggregation.

Figure 10 shows ZAME being used to navigate a bioinformatics dataset. The tool supports navigation in both data space—drill-down and roll-up—as well as in geometric space—zoom and pan. In the figure, a bar graph visualization, nested inside the matrix visual marks, is used to show an overview of the aggregated data.

## 7.2 NodeTriX

NodeTriX [17] was designed to visualize large social networks, and takes advantage of the fact that such networks have a small-world nature of being globally sparse but locally dense. The technique achieves this by combining adjacency matrices inside a traditional node-link diagram, using the matrix for locally dense cliques, and visualizing the sparse connections between the cliques in the node-link representation. The combination also addresses the fact that matrix representations on a whole are less familiar to users than node-link representations. Figure 11 shows an example of the technique being used for studying co-authorship data.

NodeTriX is an instance of nested views because it composes two visualizations inside a single visual representation. This is also another example of how combining visual representations provides holistic advantages that the components never possessed. However, it is also clear that a significant amount of design is needed to integrate the two visual representations efficiently.

## 8 DESIGN SPACE

Finally, using the above review of existing work, we here define a unified design space for composite visualization based on the visualizations involved, the spatial composition, and the data relations between items in the individual visualizations. We use this design space to classify existing composite visualizations in Table 1.

**Visualizations** Virtually any type of visualization can form a component in a composite visualization. The choice depends on the dataset being visualized. Some composite visualizations may even consist of more than two component visualizations.

**Spatial Relation** For a visual composition, spatial relation describes how two or more visual structures are related in the physical display space or in the view. We use the same visual compositions defined in Section 2.2 for this attribute.

**Data Relation** This factor describes the visual relation between the data items in one visual structure to the data items in another. Based on our literature review, different values used for this factor are as follows:

- **None:** No visual relation between items of different views;
- **Item-item:** One-to-one visual relation between the items of different views;
- **Item-group:** An item in one views is related to multiple items in another; and
- **Item-dimension** An item in one view acts as a dimension or scale for another.

### 8.1 Guidelines

Any design pattern framework needs a higher level discussion where the designers take a step back and discuss the framework as a whole. Below we will try to do that for each of the five patterns.

**Juxtaposed views** support implicit visual linking and hence add very little additional visual clutter to the resulting display.

- **Benefits:** The component visualizations are independent and can be composed without interference. Easy to implement.
- **Drawbacks:** Implicit visual linking is not always easy to see, particularly when multiple objects are selected. Space is divided between the views, yielding less space for each view.

- **Applications:** Use for heterogeneous datasets consisting of many different types of data, or for where different independent visualizations need to be combined.

Figure 12(a) shows an example of juxtaposed views realized through two underlying visualizations: a scatterplot and a bar graph. The two visualizations are coordinated through brushing that allows selection in one view (purple bins in the bar graph) to be coordinated in the other view (purple squares in the scatterplot). The available space is equally divided between the two visualizations but it is also possible to have a non-symmetrical division. The two visualizations are independent, have different scale, and are displaying different, but related, datasets.

**Integrated views** use explicit visual linking between multiple component visualizations.

- **Benefits:** Easy to perceive one-to-one and one-to-many relations between items in components. Visualizations are less independent compared to juxtaposed views, but still separate.
- **Drawbacks:** Extra visual clutter added to the overall view. Display space is split between the views. Some dependencies exist between views to allow for the visual linking.
- **Applications:** Use for heterogeneous datasets where correlation and comparisons between views is particularly important.

Figure 12(b) depicts an example of integrated views based on scatterplot and bar graph visualizations. Like the example for juxtaposed views, discussed previously, here also each of these visualizations is displaying different but related datasets. The relational linking among the data values is shown with the help of straight lines from one view to another that add extra visual clutter.

**Superimposed views** overlay one component visualization on another and hence are useful in performing direct comparison among data values associated with these visualizations.

- **Benefits:** Allows direct comparison in the same visual space.
- **Drawbacks:** May cause occlusion and high visual clutter. The client visualization must share the same spatial mapping as the host visualization.
- **Applications:** In settings where comparison is common, or where the component visualization views need to be as large as possible (potentially the entire available space).

Continuing the example of combining a scatterplot and a bar graph, Figure 12(c) shows an example of superimposed views based on these two visualizations. Unlike the examples for juxtaposed and integrated views, the datasets associated with the two visualizations must share the same horizontal and vertical axes.

**Overloaded views**, like superimposed views, overload the space of one visual representation with another visual representation.

- **Benefits:** The client visualization does not have to share the same coordinate space as the host visualization. This also yield more flexibility and control over visual clutter.
- **Drawbacks:** Visual clutter is increased, and the visual design dependencies between components are significant.
- **Applications:** Situations where one visualization can be folded into another to yield a compact (and complex) result.

Figure 12(d) depicts an example composition of scatterplot and bar graph visualizations based on this design pattern. In the figure, the scatterplot is acting as a host for the bar graph visualization. Here, we assume that the dataset is visualizing something related to two adjacent items in the dataset (items 3 and 4), such as number of samples, so we have simply folded the bar graph between those items in the scatterplot. This not only allows us to visualize the two

Technique	Visualization A	Visualization B	Spatial Relation	Data Relation
ComVis [24] (Figure 2)	any	any	juxtapose	none
Improvise [39] (Figure 3)	any	any	juxtapose	none
Jigsaw [36]	any	any	juxtapose	none
Snap-Together [30]	any	any	juxtapose	none
semantic substrates [34] (Figure 4)	node-link	node-link	juxtapose	item-item
VisLink [11] (Figure 5)	radial graph	node-link	juxtapose	item-item
Napoleon’s March on Moscow [37]	time line view	area visualization	juxtapose	item-item
Mapgets [38] (Figure 6)	map	text	superimpose	item-item
GeoSpace [22] (Figure 7)	map	bar graph	superimpose	item-item
3D GIS [8]	map	glyphs	superimpose	item-item
Scatter Plots in Parallel Coordinates [45] (Figure 8)	parallel coordinate	scatterplot	overload	item-dimension
Graph links on treemaps [14] (Figure 9)	treemap	node-link	overload	item-item
SparkClouds [21]	tag cloud	line graph	overload	item-item
ZAME [13] (Figure 10)	matrix	glyphs	nested	item-group
NodeTrix [17] (Figure 11)	node-link	matrix	nested	item-group
TimeMatrix [44]	matrix	glyphs	nested	item-group
GPUVis [25]	Scatterplot	glyphs	nested	item-group

Table 1: Classification of common composite visualization techniques using our design space.

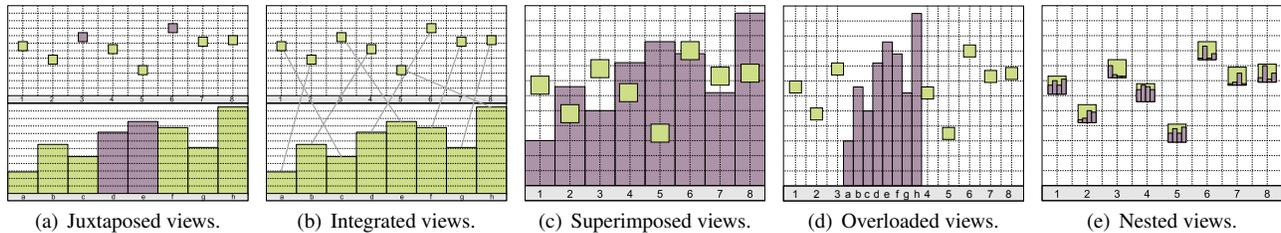


Figure 12: Example of composing a scatterplot and bar graph using different methods.

datasets in the same space and using different visualizations, but also highlights the relational linking between the two datasets.

**Nested views** provide an efficient approach to link each of the data values, visualized through the host visualization, to its related dataset, visualized through client visualizations. This is achieved by nesting clients inside the visual marks in the host.

- **Benefits:** Very compact representation, easy correlation.
- **Drawbacks:** Limited space for the client visualizations, clutter is high, and visual design dependencies are high.
- **Applications:** Again, situations that call for augmenting a particular visual representation with additional mapping.

Figure 12(e) shows an example composition of scatterplot and bar graph visualizations based on this design pattern. In the figure, the scatterplot visualization is acting as a host and bar graph visualizations are nested inside its visual marks.

There is probably not a clear winner among different design patterns while designing an information visualization tool. The correct choice of design pattern to use for a particular implementation depends on different conditions, such as the available view space, user knowledge, and the complexity of the underlying dataset. Ideally speaking, designers should be able to combine any existing visualizations to generate a composite visualization view.

## 8.2 Delimitations

While our above CVV design patterns are general in nature, they are based solely on the spatial layout of component visualizations. However, it is possible to envision other ways to combine two or more visualizations, for example using interaction or animation. One such example is the use of interactive hyperlinking [6, 43] (or *wormholing*) to navigate between different visualization views.

## 8.3 Discussion

There are several direct benefits to structuring the design space of composite visualization views in this manner. Classifying existing techniques into patterns not only helps in understanding these techniques, but also in evaluating their strengths and weaknesses.

However, the design patterns presented in this paper are all based on evidence from the literature of how existing visualization tools and techniques use composite views. Therefore, our framework is inherently limited to current designs, and more descriptive than generative in nature. Furthermore, this list of patterns is not necessarily exhaustive, and we certainly foresee additional design patterns for composite views to emerge with progress in information visualization. It is also not always straightforward to separate what is a composite visualization and what is an “atomic” (or component) visualization, particularly when the compositions on the visual structures—which is the case for overloaded and nested views—as opposed to merely on the views. Our approach in the above text has been to treat as components any technique has been presented in the literature as a standalone technique.

## 9 CONCLUSION

We have proposed a novel framework for specifying, designing, and evaluating compositions of multiple visualizations in the same visual space that we call *composite visualization views*. The benefit of the framework is not only to provide a way to unify a large collection of existing work where visual representations are combined in various ways, but also to suggest new combinations of visual representations that may significantly advance the state of the art.

## REFERENCES

- [1] C. Ahlberg and B. Shneiderman. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *Proceed-*

- ings of the ACM Conference on Human Factors in Computing Systems, pages 313–317, 1994.
- [2] C. Alexander, S. Ishakawa, and M. Silverstein. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, New York, 1977.
  - [3] G. Andrienko and N. V. Andrienko. Making a GIS intelligent: Common GIS project view. In *AISB Quarterly*, pages 15–17, 1999.
  - [4] M. Q. W. Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 110–119, 2000.
  - [5] R. A. Becker and W. S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.
  - [6] B. B. Bederson and J. D. Hollan. Pad++: A zooming graphical interface for exploring alternate interface physics. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 17–26, 1994.
  - [7] N. Boukhelifa, J. C. Roberts, and P. Rodgers. A coordination model for exploratory multi-view visualization. In *Proceedings of the International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pages 76–85, 2003.
  - [8] S. Brooks and J. L. Whalley. Multilayer hybrid visualizations to support 3D GIS. *Computers, Environment and Urban Systems*, 32(4):278–292, 2008.
  - [9] S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization — Using Vision to Think*. Morgan Kaufmann, 1999.
  - [10] W. S. Cleveland. *Visualizing Data*. Hobart Press, 1994.
  - [11] C. Collins and M. S. T. Carpendale. VisLink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1192–1199, 2007.
  - [12] C. Collins, G. Penn, and M. S. T. Carpendale. Bubble Sets: revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1009–1016, 2009.
  - [13] N. Elmqvist, T.-N. Do, H. Goodell, N. Henry, and J.-D. Fekete. ZAME: Interactive large-scale graph visualization. In *Proceedings of the IEEE Pacific Symposium on Visualization*, pages 215–222, 2008.
  - [14] J.-D. Fekete, D. Wang, N. Dang, A. Aris, and C. Plaisant. Overlaying graph links on treemaps. In *Poster Proceedings of the IEEE Symposium on Information Visualization*, 2003.
  - [15] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
  - [16] J. Heer and M. Agrawala. Software design patterns for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):853–860, 2006.
  - [17] N. Henry, J.-D. Fekete, and M. J. McGuffin. NodeTrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007.
  - [18] A. Inselberg. The plane with parallel coordinates. *Visual Computing*, 1:69–91, 1985.
  - [19] W. Javed, B. McDonnell, and N. Elmqvist. Graphical perception of multiple time series. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):927–934, 2010.
  - [20] B. Johnson and B. Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the IEEE Conference on Visualization*, pages 284–291, 1991.
  - [21] B. Lee, N. H. Riche, A. K. Karlson, and M. S. T. Carpendale. SparkClouds: Visualizing trends in tag clouds. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1182–1189, 2010.
  - [22] I. Lokuge and S. Ishizaki. GeoSpace: An interactive visualization system for exploring complex information spaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 409–414, 1995.
  - [23] P. Lucas, S. F. Roth, and C. C. Gombert. Visage: Dynamic information exploration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 19–20, 1996.
  - [24] K. Matkovic, W. Freiler, D. Gracanic, and H. Hauser. ComVis: A coordinated multiple views system for prototyping new visualization technology. In *Proceedings of the International Conference on Information Visualization*, pages 215–220, 2008.
  - [25] B. McDonnell and N. Elmqvist. Towards utilizing gpus in information visualization: A model and implementation of image-space operations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1105–1112, 2009.
  - [26] P. Neumann, D. S. Schlechtweg, and S. Carpendale. ArcTrees: visualizing relations in hierarchical data. In *Proceedings of the IEEE European Symposium on Visualization*, pages 53–60, 2005.
  - [27] K. L. Norman, L. J. Weldon, and B. Shneiderman. Cognitive layouts of windows and multiple screens for user interfaces. *International Journal of Man-Machine Studies*, 25(2):229–248, 1986.
  - [28] C. North. Robust, end-user programmable, multiple-window coordination. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 60–61, 1998.
  - [29] C. North and B. Shneiderman. A taxonomy of multiple window coordination. Technical report, University of Maryland, 1997.
  - [30] C. North and B. Shneiderman. Snap-together visualization: A user interface for coordinating visualizations via relational schemata. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 128–135, 2000.
  - [31] J. C. Roberts. On encouraging multiple views for visualization. In *Proceedings of the International Conference on Information Visualization*, pages 8–15, 1998.
  - [32] J. C. Roberts. State of the art: Coordinated and multiple views in exploratory visualization. In *Proceedings of the International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pages 67–71, 2007.
  - [33] B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1992.
  - [34] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006.
  - [35] J. Stasko, R. Catrambone, M. Guzdial, and K. McDonald. An evaluation of space-filling information visualizations for depicting hierarchical structures. *International Journal of Human-Computer Studies*, 53(5):663–694, 2000.
  - [36] J. T. Stasko, C. Görg, and Z. Liu. Jigsaw: supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132, 2008.
  - [37] E. R. Tufte. *The visual display of quantitative information*. Graphics Press, 1983.
  - [38] A. Voisard. Mapgets: A tool for visualizing and querying geographic information. *Journal of Visual Languages and Computing*, 6(4):367–384, 1995.
  - [39] C. Weaver. Building highly-coordinated visualizations in improvise. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 159–166, 2004.
  - [40] C. Weaver. Patterns of coordination in improvise visualizations. In *Proceedings of the SPIE Conference on Visualization and Data Analysis*, 2007.
  - [41] C. Weaver. Cross-filtered views for multidimensional visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):192–204, 2010.
  - [42] P. C. Wong and R. D. Bergeron. Multivariate visualization using metric scaling. In *Proceedings of the IEEE Conference on Visualization*, pages 111–118, 1997.
  - [43] A. Woodruff, P. Wisnovsky, C. Taylor, M. Stonebraker, C. Paxson, J. Chen, and A. Aiken. Zooming and tunneling in tioga: Supporting navigation in multidimensional space. In *Proceedings of the Symposium on Visual Languages*, pages 191–195, 1994.
  - [44] J. S. Yi, N. Elmqvist, and S. Lee. Timematrix: Visualizing temporal social networks using interactive matrix-based visualizations. *International Journal of Human-Computer Interaction*, 26(11-12):1031–1051, 2010.
  - [45] X. Yuan, P. Guo, H. Xiao, H. Zhou, and H. Qu. Scattering points in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1001–1008, 2009.