

# Not As Easy As You Think—Experiences and Lessons Learnt from Creating a Visualization Image Typology

Jian Chen, Petra Isenberg, Robert S. Laramée, Tobias Isenberg, Michael Sedlmair, Torsten Möller, Han-Wei Shen

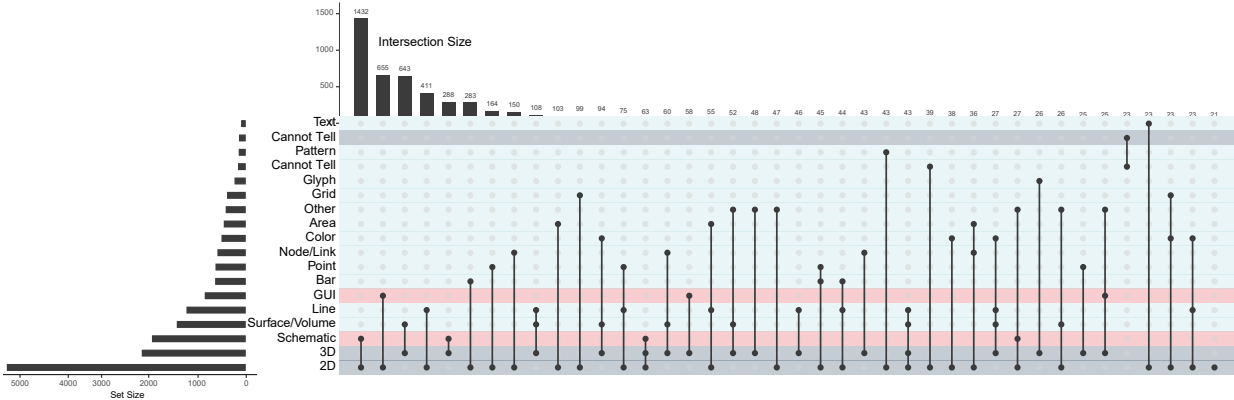


Fig. 1: Coding results from categorizing IEEE VIS paper images according to: visualization types (light blue), their dimensionality (dark blue), and additional image categories (red). 2D Schematics are the most common type of figure in IEEE VIS publications, followed by 3D surface/volume renderings.

**Abstract**—We present and discuss the results of a two-year qualitative analysis of images published in IEEE Visualization (VIS) papers. Specifically, we derive a typology of 13 visualization image types, coded to distinguish visual designs and several image characteristics. The categorization process required much more time and was more difficult than we anticipated. The resulting typology and image analysis may serve a number of purposes: to study the evolution of the community and its research output over time, to facilitate the categorization of visualization images for the purpose of research or teaching, to identify visual design styles, or to enable progress towards standardization in visualization. In addition to the typology and image characterization, we provide a dataset of 6,833 tagged images and an online tool that can be used to explore and analyze the large set of tagged images. The tool and data set enable a close examination of the diverse visualizations used and how they are published and communicated in our community.

**Index Terms**—Visualization, classification, images, typology

## 1 INTRODUCTION

The visualization research space has been studied from a variety of angles. Some started by considering specific data types (TimeVis [2], TreeVis [74]), others looked at keywords [47], evaluations [48, 56], topic modeling [50], interaction [108], or taxonomies of tasks and activities [3, 27]. Describing and classifying the challenges, artefacts, research methods, and theories in a research field is a difficult endeavor but this work is nevertheless very important for a variety of tasks. For example, a classification of visualization techniques can be extremely helpful when planning any research activity that requires a systematic coverage of the space of existing visualization. For example, when developing a new research method applicable to a variety of visualization techniques, it is important to test the method and ground the design on a broad set of techniques or variations of a specific technique [82]. Similarly, when writing an overview article, a textbook, or lecture series about visualization—classifications can illuminate coverage of techniques, to show the variety of approaches to a specific techniques, or to

identify aspects that need further attention. When trying to understand the historical evolution of the visualization field, it may be similarly useful to systematically consider the types of artefacts and research produced through the lens of a classification. In all three cases more broadly, a systematic organization can help to ensure coverage of a research space, identify outliers, structure discussions, and potentially even open up venues for future work [60, 77].

While many characterizations of visualization tasks, visualizations, keywords, or topics exist, what has not been systematically attempted so far is a bottom-up approach that starts with the visual artefacts published, communicated, and discussed in the community. We provide such a view on our research space and publication practices by systematically analyzing images published throughout the entire 30 year history of the IEEE Visualization conference—as the longest running venue for the publication of novel representation types, the evaluation of existing techniques, or the development of visualization systems (among other types of contributions). Specifically, we coded 6,833 figures from 695 papers published in IEEE VIS (VisWeek) 1990, 1995, 2000, 2005, 2010, 2015, and 2020, a subset of the VIS30K dataset [22]. Our initial goal was to study how visualizations were used to communicate research in the community. Throughout our two-year collaboration and lively discussions, this goal evolved toward establishing a broader typology of the visualizations we saw as well as a description of how these visualizations are used in visualization research publications. Our discussions focused on the visual appearance of these visualizations without consulting the captions and reading the authors’ intents in the papers. We discussed whether the images showcased a certain type of

- J. Chen and H.-W. Shen are with The Ohio State University, USA. E-mail: {chen.8028 | shen.94}@osu.edu.
- P. Isenberg and T. Isenberg are with Université Paris-Saclay, CNRS, Inria, LISN, France. E-mail: {petra.isenberg | tobias.isenberg}@inria.fr.
- R. S. Laramée is with the University of Nottingham, UK. E-mail: robert.laramée@nottingham.ac.uk.
- M. Sedlmair is with University of Stuttgart, Germany. E-mail: michael.sedlmair@visus.uni-stuttgart.de.
- T. Möller is with University of Vienna, Austria. E-mail: torsten.moeller@univie.ac.at.

visual design, a system or GUI element, or were schematics meant to explain workflows or processes. Our final code set describes 13 image types and their perceived dimensions (Fig. 1). We find that the largest categories were schematic representations, surface-based techniques & volumes, line-based-technique, and GUIs. Together, these top four categories account for 73% (4,986 out of 6,833) of the coded images.

By relying on our experience in visualization research, teaching, and practice we initially assumed the coding process would be relatively straightforward. Having studied visual encoding principles as put forward by Bertin [8], MacKinlay [62], and others, the design space of visualizations, in theory, seemed more or less clearly defined. However, it quickly became clear that visual designs “in the wild” (even with our constraint to the academic visualization community) display a great variety and demonstrate tremendous creativity, to the extent that the complexity of coding these charts is a major challenge of our work. There are, for instance, no standard definitions of many visual designs, e. g., glyphs and grid-based techniques, that are specific enough to foster a clear-cut coding process of visualization images. Given our shared background in visualization, we were also surprised about the large role that individual differences and interpretations played. And for some images, even after intense discussions, their categorization remained ambiguous and uncertain. In summary, we contribute:

- a novel typology of visualization images consisting of 13 categories,
- the coding dataset and quantitative analysis of 6,833 IEEE VIS (VisWeek) images based on the typology,
- a discussion of our process, failed attempts, and coding ambiguities in deriving the typology, and
- an open web-based tool to explore the image dataset.

## 2 RELATED WORK

Past work on visualization categorizations is related to our own, as well as work that analyzes research figures. We review these areas next.

### 2.1 Categorization as an Analogy of “What is it *like*?” Association: A Brief History

Categorization represents any grouping based on similarity [6], originated from Plato in philosophy [36]. Wittgenstein [105] argues that people are fluid about categories: different people may give different answers and the same person may give us at different time different answers. As a result, a top-down categorization that draws clear boundaries of shared properties to those of other categories, is too strict to represent how people understand categories. Rosch [80] later updated the notation of categorization using a data-driven solution where categories become *prototypes* from bottom-up clustering of similar instances. These prototypes can get clustered again to form a hierarchical categorization. Psychologists (e. g., Medin and Schaffer [67], Nosofsky [73], and Krushki [55]) have gone further and argue for what they call the *example-based theory* of categorization, where humans store *instances* and *examples* of these; similarities and differences among which let us see and learn *associations* between these examples. For things closer together, they *look like* a clustering of learned examples and belong to the same category. Human observations of similarity depend on high-order structure [98] (thus cannot be represented by low-level features [110]), and are context-dependent [66] (thus categorical boundaries can thus be fuzzy). In this work we also stepped away from the top-down categorization paradigm and tried to categorize visualizations as a bottom-up association of what we see from these images.

### 2.2 Visualization Categorizations

Images have been categorized based on how they are constructed, rather than how they are seen by viewers. Textbooks, in particular, often rely on categorizations to structure their content [78]. While early books such as Brinton’s [14] were collections of graphical representations in use, modern textbooks regularly use one of a few structures:

**Focus on construction rules for design purposes.** A first and highly influential approach to characterizing visual designs was Bertin’s visual semiotics [8]. He discussed the fundamental building blocks of images

that are modified by visual variables (visual channels), which encode data. Similar in spirit, several proposals have been made to describe visual designs through the lens of a visual language with a set of syntactic rules. Examples include Wilkinson’s Grammar of Graphics [104], Engelhardt’s Language of Graphics [94], or Mackinlay’s automatic design [62]. Applying rules formulated in a visual language can yield a broad range of visualization designs [69] and several visualization tools and libraries are based on them, e. g., Tableau [63], D3 [13], or Vega-lite [81]. Others, such as Tufte’s *Envisioning Information* [90] differentiate techniques by higher-level construction rules such as small multiples, or principles such as layering and separation.

What unites these approaches is that they attempt to describe *how* to construct a visualization but do not focus on *what* the end-product of the construction looks like. Different sets of rules may lead to images that *look* very similar. Intriguingly, these construction rules cannot name a category nor tell apart visualization categories. For example, we cannot use length, area, and orientation to differentiate a bar chart from a pie chart. In another words, humans cannot always rely on abstract definitions or shared drawing entities to predict categories [80, 105]. We used an inverse approach in that we took existing images and attempted to describe their visual appearance. By using this approach, our characterization incorporates our assessment of what is important in a visualization; this assessment is certainly related to encoding of marks and channels without necessarily considering what data is encoded. This approach enables us to generate high-level categories beyond marks and channels.

**Focus on data types.** Many researchers have categorized visualization techniques based on the type of data they show. This approach makes sense as, in a typical iterative visualization design process, data are systematically mapped, winnowed, and refined to visual encoding [82]. Ward et al. [101], e. g., categorize visualization techniques for spatial data, geospatial data, time-oriented data, multivariate data, trees, graphs, and networks, and text and document visualizations. Heer et al.’s visualization zoo [40] classifies time series, statistical data, maps, hierarchies, and networks. Brodlić [15, pp. 40ff] classifies techniques into those for point, scalar, vector, and tensor data. Similar to the first approach, these categorizations focus on how to construct a visualization and do not focus on how to describe the visual appearance of an image. Compared to these characterizations where data is input and visual images are the output, we attempt to characterize visualizations without necessarily knowing the characteristics of the data that led to the final image. For example, we make no distinction between a line chart that shows temporal data and one that shows, e. g., a physical measurement such as voltage that was sampled in some arbitrary sequence but plotted in a meaningful way from low to high values.

**Focus on task and analysis question types.** Another set of textbooks introduces visualizations by linking representation and analysis tasks/questions. Fisher and Meyer [33], for example, group techniques such as histograms and boxplots under the analysis question of “showing how data is distributed.” Maciejewski [61] also takes this approach in his grouping of techniques. Again, a focus on analysis questions considers a-priori criteria to choose and categorize visualization techniques in the same vein as data and construction rules do. As a consequence, visually similar techniques are considered in separate categories; for example, Fisher and Meyer [33] categorize bar charts under “visualizations that show how groups differ” and histograms under “visualizations that show how data is distributed.” Again, our approach attempts to uniquely identify images from the standpoint of having been created already, without necessarily knowing what data they show, how they have been constructed, or what tasks they were meant to serve. This allows us to group visually similar techniques together and only later to consider other aspects in which they differ.

**Most categories are functional.** We are certainly not the first to realize the feature differences between *what we design* and *what we see*. The computer vision community has realized that many descriptors (e. g., Canny edge detector [17], orientation map [64], and HOG algorithms [30]) use features that do not align with what humans see, and therefore cause some computer vision algorithms to fail. Recent artificial intelligence algorithms are better able to assign categories

to items because the categories are treated as a continuous space of related high-level concepts [20, 65]. In the space of artificial intelligence categorization, our results are thus useful for future grouping and classifying new visualization techniques (akin to ImageNet in computer vision [54]). Nonetheless, we also show the diversity of the data: even under a single category. Our collection combines various variable appearances, spatial arrangements, appearances (color), compositions, and viewpoints.

### 2.3 The Role of Graphs in Scientific Communication

Though our perspectives on how to categorize visualizations is different from that of many textbooks, our method of studying figures in evaluating scientific advances has been used before. Latour [57] laid out graph features that make them essentially a pervasive form of visualization and a specialized vocabulary for transforming and analyzing data to represent scientific findings. The pervasiveness and centrality of scientific figures led Latour to conclude that scientists exhibit a “graphical obsession” and indeed to suggest that the use of graphs distinguishes scientific domains. In the visualization domain, our work is most closely related to Borkin et al.’s [12] work towards studying what makes images memorable. In that work, the authors suggested a taxonomy of techniques that is a mix of encoding (area, bar, . . .), data (network, tree, . . .), and analysis-focused (distribution) categories. The authors asked students to annotate 2,070 single-panel visualizations using their taxonomy and derived an annotated set of images. Our work differs, however, in several aspects from Borkin et al.’s: we describe and discuss the process of deriving our categorization and the inherent difficulties, we included images with multiple visual encodings, we focus on visualization articles as a source and do not study memorability as a final goal of our work. Our approach is one of the few that focuses on images only. Visuals are one of the most essential outputs from the visualization community (as opposed to data or tasks). We consider them to be very important since they are at the center of our work. Our project thus facilitates the classification, exploration, and analysis of our own fundamental content through a new image-centric lens.

## 3 THE IMAGE CODING PROCESS

Our image coding is an ambitious project. We list a set of high-level goals we strive for and the process to reach these goals. We begin our discussion with a temporal overview of our process to classify images and our systematic methodology. Our process became one of open and axial coding [18] while we continuously updated, drew connections between, and refined our codes as we analyzed more data.

### 3.1 Goals of our Image Coding

To summarize our previous discussion, our categorization of images according to “what we see” has four concrete goals:

**Provide an alternative viewpoint:** Rather than categorizing visual designs from underlying data, construction rules, or functions we provide a categorization based on the visual content of images alone. This approach offers a new viewpoint that can serve to compare and complement other categorizations and puts the focus on the diversity of aesthetics and other visual properties within a single category.

**Collect experiences concerning the difficulties of categorizing visualization images:** We document our multi-stage process to derive a relatively high-level categorization of images and describe inherent uncertainty, failed attempts, and current limitations. We also describe how difficult it can be to understand images that have been taken out of the context of the text and captions.

**Provide a small set of broad categories:** We purposefully wanted to create a classification with only a few categories that would remain manageable given the detailed and often complex types of images produced in the Visualization community. These categories needed to capture the diversity of design approaches, rendering methods, algorithms, or view point selections within a category.

**Provide data and explore the use of images in the visualization community.** This exploration can give valuable insight in the changing practices of communication and research in our community.

### 3.2 Visualization Image Data Source

We developed our classification using the VIS30K [22] collection of images and its associated VisPubData [46] meta data. This dataset largely represents visualization as a field because it contains every visualization image published at IEEE VIS (including Visual Analytics, Information Visualization, and Scientific Visualization) since 1990. Based on our pilot studies (described below) it became clear that we would not be able to classify all 30,000 images. Thus we chose to code images in five-year intervals for our primary study, starting with 1990 and up until 2020 (inclusive). This led to a set of 6,833 figures from 695 IEEE VisWeek/VIS full papers (including case studies), which we analyzed. In each phase of the work, seven experienced coders, all co-authors of this manuscript, classified subsets of the images.

### 3.3 Image Classification Process

Included in our process description are the approximate start dates and duration of each phase.

**Phase 1—Initial image classification based on keywords** (circa Mar. 2020 start, approx. 1 month): We began our work with a focus on *visualization techniques* where we considered that technique names such as treemaps, parallel coordinates, etc. could well describe the content of the images we were analyzing. To improve objectivity and reduce bias, we wanted to tag images with the most common technique names used in the community as extracted from author keywords used for VIS papers. We ranked the author keywords extracted in prior work [47] to derive the initial top-21 keywords for specific techniques. In addition to the encoding techniques used in each image, we added two code categories that seemed important to be able to describe visualizations and communication practices in the community: the rendering dimensionality (i. e., 2D or 3D) and the functional purposes of creating the image (i. e., the reason why the authors created each image, for example: illustration of a visualization technique, experiment results, or screenshot of GUI.) See Appendix B for the initial keyword list and functions. The initial code set, thus, included 28 codes.

**Phase 2—Initial pilot coding** (circa Apr. 2020 start, approx. 2 months): To test our initial code set, each coder categorized visualization images from the year 2006. Visualization images from the year 2006 were used in our pilot study only. We subsequently introduced new technique codes by merging techniques that had been tagged “other,” and added “schematic diagram” to the list of image purposes, which resulted in 22 technique codes. We discussed the definition of these terms and gave all coders written instructions and example images from each category. Each figure was labeled initially by one coder in this stage and validated by a second coder. We based the validation assignment of the second coder on their respective expertise, to verify all images included and excluded in a specific category (e. g., volume rendering was coded and verified by experts with a sustained track record in volume graphics). With these steps we removed false positives, avoided false negatives, and ensured image classification consistency.

**Phase 3—Consolidation: Seeing by association and analogies** (circa Jun. 2020 start, approx. 5 months): In Phase 3, we discussed what worked well and what did not, and why. The codes focused on visualization techniques quickly became difficult to apply as the number of techniques grew increasingly large. We had difficulty defining when a technique should receive its own code or be covered under “other.” Also, some technique names were different but pointed to visually similar designs. Point clouds and 3D scatterplots, e. g., both render points according to underlying coordinates in 3D space, with the main exception that scatterplots typically include reference structures such as axes and gridlines. These conflicts in turn lead us to re-frame our code set using higher-level (more general) visualization type codes.

We decided to focus on describing the main perceptual (or visual) characteristic of a given visualization and to create codes that enable the viewer to distinguish graphical similarities and differences. We thus re-grouped and merged the codes sharing similar visual characteristics into a more general code. We put *isosurface*, e. g., into a more general *surface-based techniques* category and grouped *point clouds* and *scatterplots* into a more abstract *point-based techniques* category.

Table 1: The main visualization type, function, and dimensionality codes used in our review. Additional codes not listed here were “I cannot tell” to label images that had unclear techniques or dimensionalities.

Visualization Type Codes		Description	Examples
(1)	Generalized Bar Representations	Graphs that represent data with rectangular bars whose heights or lengths are proportional to the values they represent.	bar charts, stacked bar charts, histograms, box plots, sunburst diagrams.
(2)	Point-based Representations	Representations that use point marks to represent data samples	scatterplots (2D/3D), point clouds, dot plots, bubble charts.
(3)	Line-based Representations	Representations in which lines, edges, or curves represent data samples. Lines can depict surface features or data values.	line charts, parallel coordinates, contour lines, radar/spider charts, streamlines, or tensor field lines.
(4)	Node-link Trees/Graphs, Networks, Meshes	Representations using points for nodes/points and explicit connections to convey relationships between data values	node-link diagrams, node-link trees, node-link graphs, meshes, arc diagrams, sankey diagrams.
(5)	Generalized Area Representations	Representations with a focus on areas of 2D space or 2D surfaces including sub-sets of these surfaces. Areas can be geographical regions or polygons whose size or shape represents abstract data.	(stacked) area chart, area chart, streamgraph, ThemeRiver, violin plot, cartograms, rideline chart, voronoi diagram, treemaps, pie chart.
(6)	Surface-based Representations and Volumes	Representations of the inner and/or outer features and/or boundaries of a continuous spatial phenomenon or object in 3D physical space or 4D space-time, or slices thereof.	terrains, isosurfaces, stream surfaces, volume rendering using transfer functions, slices through a volume (e. g., X-ray, CT slice).
(7)	Generalized Matrix / Grid	Representations that position data in a <i>discrete</i> grid structure. The grid can vary in resolution, is typically rectilinear but can use other shapes such as hexagonal grids etc.,	network matrices, discrete density maps, scarf or strip plots.
(8)	Continuous Pattern-based Representations	Representations of continuous data along planes and surfaces, typically for vector and tensor fields. Representations frequently use texture-mapped imagery to describe variations in orientations, directions, or flow.	Line Integral Convolution (LIC), Spot Noise, Image-Space Advection (ISA), Image-Based Flow Visualization (IBFV)
(9)	Continuous color-based Representations	Representations that use a primary encoding where the hue or brightness or saturation encodes quantitative values on a continuous surface. Color-mapping is systematic, thus is not as a result of illumination or an author-chosen categorical representation.	pixel heatmaps, color-mapped surfaces.
(10)	Glyph-based Representations	Multiple small independent visual representations that depict multiple attributes (dimensions) of a data record. Placement is usually meaningful and typically multiple glyphs are displayed for comparison.	Star glyphs, 3D glyphs, Chernoff faces, vector field glyphs
(11)	Text-based Representations	Representations of data (usually text itself) that use varying properties of letters and words such as font size, color, width, style, type to encode data.	tag clouds, word trees, parallel tag clouds, typomaps.
Function Codes		Description	Examples
A.	GUI Screenshots or GUI Photos	Images that show a system or user interface.	a photograph of a person sitting in front of a given system, a figure containing GUI features such as windows, icons, cursor, and pointers (WIMP), or non-WIMP VR/AR interfaces.
B.	Schematic Representations, Concept Illustrations	Often simplified representations showing the appearance, structure, or logic of a process or concept.	workflow diagrams, algorithm diagrams, sketches.
Dimensionality Codes		Description	Examples
	2D	Flat representations, no specific depth codes added to renditions.	Most statistical charts, most maps, . . .
	3D	Representation with specific depth cues that achieve the perception of 3D (shading, perspective, lighting, . . .).	Most volume renderings, . . .

This consolidation process resulted in 12 high-level visualization type codes that ultimately became part of our final set shown in Table 1. We reduced our image function category into just two codes: GUI (Screenshots) and Schematics as these were visually identifiable without requiring knowledge of the underlying data. Both categories represent the codes from other encoding categories from Phase 1, which did not appear in our technique-focused codes. Given the challenges to code the visualization images we had encountered, we also decided to collect subjective ratings of difficulty (easy, neutral, hard).

**Phase 4—2<sup>nd</sup> Pilot Visualization Typology** (circa Dec. 2020 start, approx. 3 months): In the first two months after having arrived at the new code set, we discussed, debated, and coded two sets of 50 (i. e., 100 in total) randomly chosen images from our seven-year target image data set to calibrate our collective understanding. During this exercise we clarified code definitions and discussed ambiguities. We assigned these 100 images to all of us for quality control as well to align our decisions and discuss potential pitfalls with our new code set. We used a dedicated web-based coding tool (Appendix C) for the coding in this

phase. At the end of this phase, our project was already one year old. We had almost weekly meetings and discussions throughout this time and were now confident that we had largely reached consensus on the general typology shown as the top 13 codes (except color) of Table 1.

**Phase 5—Result Coding and Validation** (circa Apr. 2021 start, approx. 6 months): In this phase we coded all 6,833 images in our chosen dataset based on the refined definitions and characterizations, using largely the same coding tool as before (Appendix C). We used the 12 visualization type codes, 2 function codes (GUI, Schematics), 2 codes for the dimensionality (2D vs. 3D), and 3 difficulty codes (easy, neutral, hard) that meant to capture how hard the categorization had been for the coder. We first looked at the function of an image. If the image showed either a schematic or a GUI we assigned the respective code. If not, the function was implicitly considered to be a “visualization example” and we proceeded to assigning a visualization type code. Both categories shared an “I cannot tell” code that was assigned when neither one of the two explicit functions nor a visualization type could be assigned. In addition, coders could freely add new codes for visualization or

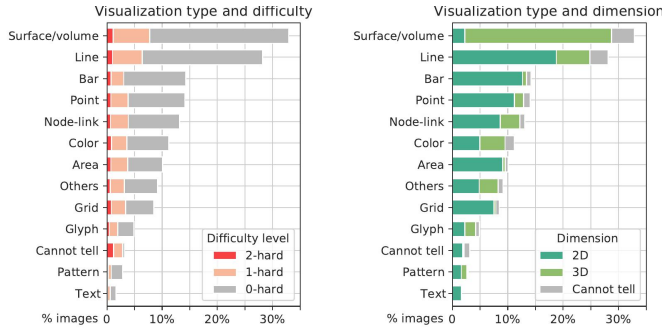


Fig. 2: The  $x$ -value represents the proportion of applied image codes in a category relative to the total 4,214 visualization images (after excluding the pure schematic and GUI images). We can see that the most common visualization types were “surface-based representations and volumes” and “line-based representations”.

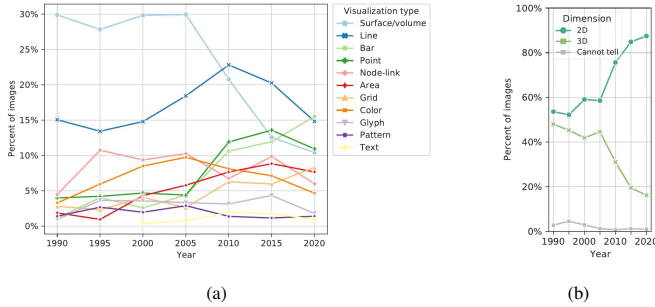


Fig. 3: Temporal overview of the proportions of visualization types (a) and dimensions (b). We can see that “surface-based representations and volumes” (Surface) become a lot less common after 2005 while at the same time “point-based representations”(Point) and “generalized bar representations” (Bar) rise.

image types when they found something new that was not covered by the existing codes. For the dimensionality we also added a code called “I cannot tell” which could be checked when coders were unsure whether the visualization was a 2D or 3D rendering. Coders could assign multiple types and dimensionalities to one image which was necessary because many images showed more than one visual design.

In the coding process, we also recoded the 100 images we had previously pilot-coded during Phase 4. To ensure high-quality results and more clearly capture potential difficulties in applying the codes two coders were randomly assigned to each image. This phase was lengthy and laborious due the number of images we classified. We met regularly to resolve further questions that arose during coding and to augment or clarify code descriptions further.

**Phase 6—Verification** (circa Oct. 2021 start, approx. 6 months): In the final phase, the two coders assigned to each image worked to reach an agreement when their applied codes did not match. For this purpose we developed two more web-based visual interfaces that focused on conflict resolution and giving and overview of applied codes (Appendix C). We then filtered the results such that only the inconsistently coded images were shown, and we used this process to resolve all disagreements. This verification was also a lengthy and laborious process requiring consistent discussions. We discussed difficult and ambiguous cases as a team every week until we could agree on a solution (we describe some of the most difficult classifications in more detail in Sect. 5). As part of this discussion we added a 13<sup>th</sup> visualization type, *continuous color-based encodings*, listed as point (9) in Table 1. We also refined our definitions further. Consequently, we went through all previously coded images again to check if they had to be re-coded for consistency, and resolved potential resulting disagreements as part of our discussion process.

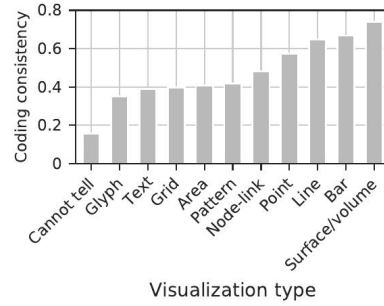


Fig. 4: The initial consistency of visualization type codes applied to images. We can see that the coders had least consistency related to “glyph-based representations”.

## 4 RESULTS

In this section, we describe the results of our work after the completion of Phase 6 discussed in the previous section. We summarize the coding results based on the codes in each coding category.

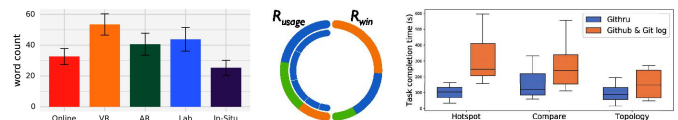
Our coding process applied 6,299 visualization type codes, 2,619 function codes, 7,181 2D/3D dimensionality codes, and 13,666 difficulty codes to the 6,833 images. Many images included multiple types and dimensionality codes. Fig. 2–left shows the distribution of visualization type in relation to the difficulty ratings given by coders for each image. The right shows the distribution of dimensionalities per visualization type. Fig. 3 provides a historic overview of the spread of visualization type and dimensions in a given year. After our final coding pass by the end of Phase 5, we calculated the consistency between the pair of coders that had coded each image. The resulting consistency depended strongly on the visualization type of the image (Fig. 4).

### 4.1 Visualization Types: Towards a Visualization Typology

In this section, we describe canonical examples for each visualization type. Later, in Sect. 5, we cover the main difficulties we encountered working with this typology. All images are referenced from left to right.

#### 4.1.1 Generalized Bar Representations

As generalized bar charts we coded visualizations that represent data with straight or curved bars whose heights or lengths seemed proportional to represented data values. Canonical examples of generalized bars are: (stacked/divided/regular) bar charts, histograms, radial bar charts, and donut charts. Generalized bar charts were the third most common visualization type among the images we coded and have gained in proportion in the later years we coded. Coders found them mostly easy to identify and the consistency between coders was among the highest at 67%. 3D generalized bar charts were extremely rare and only sometimes appeared either in the early years we coded or more recently to visualize data on 3D surfaces such as a globe. Canonical examples of generalized bar charts taken from [103], [107], [53]:



#### 4.1.2 Point-based Representations

Point-based representations typically use dots or circles with a small radius to encode data, however, we did not specify specific primitive shapes in our definition. Similar to Bertin [8], we considered point-based representation to encode point locations in a 2D or 3D space. Point marks could be small circles but also 3D spheres and sometimes other shapes like triangles, stars, etc. Canonical examples of visualization techniques of this type are scatterplots, (volumetric) point clouds, or dot plots. Point-based representations were the fourth most common visualization type according to our coding. Coders found identifying them slightly harder than length-based encodings and the overall consistency of coders was 57%. Surprisingly, we saw only a small percentage

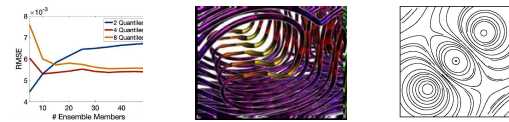
of 3D point-based representations, perhaps due to large amount of work on scatterplots or using scatterplot-like representations of, for example, dimensionality reduction or clustering results. Canonical examples of point-based representations taken from [76], [106], [91]:



#### 4.1.3 Line-based Representations

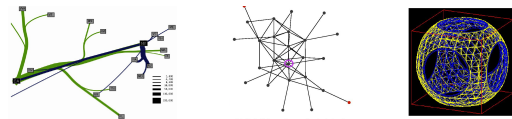
Lines, edges, and curves were the second most common representations of data according to our codes. Line-based visualization can depict surface features or data values. The lines and edges used in these visualizations could be straight or curved. Canonical line-based visualization techniques are line charts, parallel coordinates, contour lines, radar/spider charts, streamlines, or tensor field lines. We did not code lines that delineated areas as line-based representations.

About a quarter (28.1%) of line-based representations were rendered in 3D. There was also a sizable proportion of 12% of images where the coders could not tell whether a line chart was 3D or 2D due to a lack of clear depth cues. Most line charts, however, were the typical 2D line charts that most often represent temporal data. Images of canonical line-based representations taken from [4], [86], [93]:



#### 4.1.4 Node-link Trees/Graphs, Networks, Meshes

Representations of this type depict points and explicit connections to convey relationships between data values. Node-link relationships can be found in trees, graphs, networks, and meshes. Node positions can be given, e. g., geospatial locations, or be coded in the data (e. g., projections). Connections can be continuous, e. g., a Reeb graph, as the topological structure is given by showing continuous functions in space, or discrete, e. g., edges in a tree. Representations of this type were the 5<sup>th</sup> most common representation type and their representation has stayed relatively stable at roughly 5–10% of images per year. Most codes were applied to 2D images, but 27% of the codes in this category belonged to images in 3D. Canonical “Node-link trees/graphs, networks, meshes” representations taken from [75], [109], [9]:



#### 4.1.5 Generalized Area Representations

Generalized area charts are representations with a focus on areas of 2D space or 2D surfaces, including sub-sets of these surfaces. Areas can be geographical regions or polygons whose size or shape represents abstract data. Areas often feature explicit boundaries and, within, are filled with categorical colors or use contrast in luminance and shading to encode attributes of the areas. Common examples of generalized area charts are regular area charts, treemaps, cartograms, choropleth maps, pie charts, or violin plots. Generalized area charts were the 6<sup>th</sup> most common type of representation type in the images we coded. Most areas were part of surfaces rendered in 2D. Over the years the proportion of images with areas increased by 1–2% every 5 years up to just under 10% in recent years. Canonical examples of “generalized area representations” taken from [16], [83], [79]:



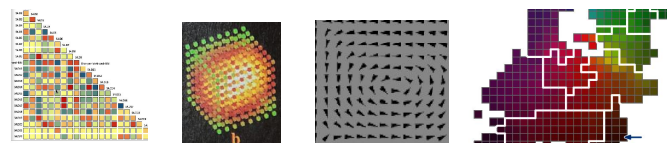
#### 4.1.6 Surface-based Representations and Volumes

Surfaces and volumes represent the inner and/or outer features and/or boundaries of a continuous spatial phenomenon or object in 3D physical space, 4D space-time, or slices thereof. Surfaces typically represent the inner and outer boundaries of a given 3D scalar field, e. g., isosurfaces [38], or integral surfaces in vector fields, e. g., stream ribbons and stream surfaces [28]. Volume rendering is a set of dedicated techniques that depict sub-sets of volume data, usually with an element of thickness (as opposed to infinitely thin surfaces) [39]. Frequent characteristics of images in this category include: the use of semi-transparency, the application of lighting and shading techniques, perspective or parallel projection, and the addition of other 3D and depth cues. Volume rendering has been one of the most important areas of Visualization in the early years of the conference [47]. As such, it is perhaps not surprising that surface and volume representations were the most common techniques in our dataset. It is also the only representation technique with primarily 3D renderings. The few 2D renderings we found included slices of volumes such as X-ray or CT slices. Similar to what we saw in prior work on keywords [47], the number of surface and volume renderings has decreased drastically since 2005 from around 25–30% of all images to currently around 15%. Canonical examples of surface and volume-based representations taken from [43], [88], [71]:



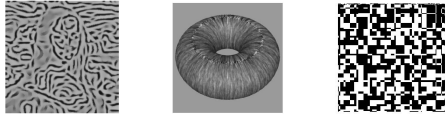
#### 4.1.7 Generalized Matrix or Grid

Generalized matrices and grids are representations that position data in a *discrete* grid structure. The grid can vary in resolution, is typically rectilinear but can use other shapes such as hexagonal grids etc. This representation type includes figures where the underlying grid is part of the data structure, but does not include figures where the underlying grid is merely used as a convenient arrangement of sub-sets of the data (as in small-multiples and scatterplot matrices). Other elements such as color or glyphs can appear at these discrete grid positions (e. g., grid-based vector field visualization). Under 10% of all images contained generalized matrices/grids and that consistently across the years. Of these, 89% were 2D grids/matrices and 4.2% were 3D (the rest was “I cannot tell”). Common visualization techniques in this representation type are discrete heatmaps, scarf/strip plots, space-time cubes, or matrix-based network visualizations. Canonical examples of this visualization type taken from [44], [72], [35], [68]:



#### 4.1.8 Continuous Pattern-based Representations

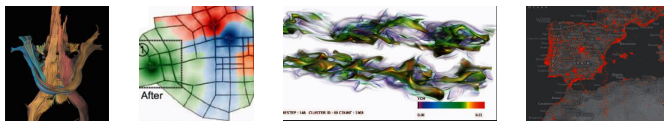
In general, continuous pattern-based representations incorporate images that focus on representing continuous data variation along planes and surfaces (akin to a “texture”). This category differentiates itself from “continuous color-based representations” (see next) by using repetitive patterns or structures in the texture mapped to data. Frequent characteristics of representations of this type are smooth, high-resolution, and highly detailed variations/changes across the data. Representations of simulated flow are common. Typical visualization techniques include: Line Integral Convolution (LIC), Spot Noise, Image-Space Advection (ISA), or Image-Based Flow Visualization (IBFV). Continuous pattern-based representations were among the most rare types in our coding with under 2.9% of all images in our dataset. Most patterns were used on surfaces represented in 2D, but some also applied to 3D. Canonical examples of this visualization type taken from [35], [92], [51]:



#### 4.1.9 Continuous Color-based Representations

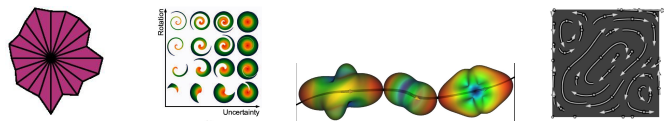
Continuous Color-based Representations use a primary color (hue, brightness, and/or saturation) encoding across a continuous surface or volume. We added this visualization type late during the verification phase (Phase 6 in Sect. 3) because many of our discussions arose around how to code heatmaps (e. g., right-most image below). Discrete heatmaps were clearly of Matrix/Grid type, but other similar color-based encodings applied in a continuous way were not. The main characteristics of continuous color-based representations are the prominent encoding of data through color at high resolution down to the individual pixel level and smooth transitions between varying colors.

A prominent technique in this category are representations featuring a transfer function. Conceptually, a transfer function is a colormap with an added opacity encoding. Other examples include continuous heatmaps and pixel-based encodings. Color-based encodings are less common in our data but this can be partly attributed to the fact that the code only appeared in the final verification phase and that some images might have been missed, this is why we do not provide a consistency score for this code. Most continuous color-based encodings were applied to surfaces rendered in 2D. Canonical examples of this visualization type from left to right taken from [26], [31], [95], [32]:



#### 4.1.10 Glyph-based Representations

We coded glyphs when we saw multiple small independent visual representations that depicted multiple attributes (dimensions) of a data record. These glyphs often used multiple geometric primitives to encode data. When multiple properties of a single mark encoded data, we also considered them as glyphs especially when we saw multiple glyphs displayed for comparison and/or with meaningful placement in space. For example, 3D glyphs, often were made up of one mark such as a small 3D cuboid where height, width, and depth could encode different data dimensions. In much of the visualization literature these marks would be named a glyph and we retained this usage of the term. Common glyph-based techniques in this category were star glyphs  $\star$ , 3D tensor glyphs  $\bullet$ , Chernoff faces  $\text{☹}$ , or vector field arrows  $\rightarrow$ . Overall, glyph-based encodings were not particularly frequent (<5% of all images contained glyphs) and we saw only slightly more glyphs rendered in 2D than 3D. Glyphs, however, were difficult to identify and our consistency for this visualization type was initially only 35%. Canonical examples of this visualization type taken from [29], [41], [42], [35]:



#### 4.1.11 Text-based Representations

Text-based representations encode data (usually text itself) using varying properties of letters and words such as font size, color, width, style, or type. Common visualization techniques for this representation type are tag clouds, word trees, or typomaps. We did not code images where text was used only for labeling and annotation or where text was the underlying data source but the representation did not use text properties to encode the text. Text-based representations were the most rare in our coding. All text-based representations were rendered in 2D. The initial coding consistency was low at 39% primarily because some coders initially also coded representation of text as a data source. Example images for this type taken from [59], [97], [1]:

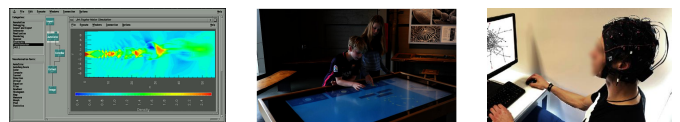


## 4.2 Visualization Functions

All images tagged with a visualization type code were implicitly assigned the function to “showcase a visualization technique.” This function was by far the most common. In addition to this implicit function we coded screenshots or images of graphical user interfaces (GUIs) and schematic representations. For images with both of these functions we did not assign additional individual visualization types.

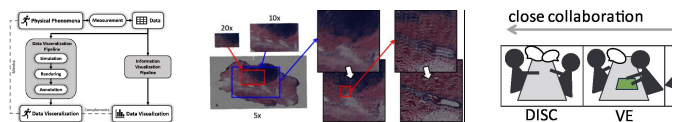
### 4.2.1 GUI (Screenshots)/User Interface Depiction

Images tagged with this code were either screenshots or photos of a system interface. GUIs required the presence of window components or other UI widgets, such as buttons, sliders, boxes, scroll bars, pointers (e. g., the hand cursor showing interaction), etc. Non-WIMP interfaces (e. g., for VR or touch-based applications) were indicated by, e. g., a hand/finger touching a surface or clearly visible interface hardware such as a tablet, a tabletop display, or other types of hardware. There are 825 GUI images in total representing about 12% of all images. The proportions of the GUI images over time are also relatively stable over the years. Only about 13% of these are 3D. Coders found identifying them was easy and the overall consistency of coders was 70%. Canonical examples for this type taken from [7], [25], [109]:



### 4.2.2 Schematic Representation and Concept Illustrations

A schematic or concept illustration is an often simplified representation showing the appearance, structure, or logic of a process or concept. Typical examples include flowcharts to illustrate an algorithm, process diagrams, or sketches. Schematics and illustrations are common in research papers, not just in visualization papers. We coded 1,919 schematic or concept illustration representations. This category is most common and among these, 79.3% were 2D. Canonical examples of this visualization type taken from [58], [49], [45]:



## 4.3 Dimensionality: 2D and 3D

Another category we coded concerned the spatial dimensionality of the rendering of visualizations, GUIs, and schematics. A flat representation on a 2D plane without perceived depth was labeled as 2D. Those images that appeared to be in 3D (or volumetric) space were classified as 3D. To classify an image as 3D we looked for depth cues such as occlusion, lighting and shading, parallel and perspective projection, rotation, or any other depth cues. We also required a continuous depth with smooth transitions between depth values. In other words, we did not generally code images as 3D when the content resided in just one or two 2D planes, e. g., an artificial discrete depth. As we shall see in Sect. 5, even this apparently simple exercise turned out to be non-trivial for many images due to the presence or absence of a mixture of depth cues. Two-dimensional representations became more common than 3D after 2005, perhaps due to the new VAST conference (Fig. 3b). It is also not surprising that 3D representations were used more broadly for “surface-based representations and volumes” (Fig. 2). For the visualization types “surface-based representations and volumes” and “line-based representations” we observed a decrease of 3D use over time.

## 5 CHALLENGES ON JUDGING VISUALIZATION TYPES

While the previous section might seem relatively straightforward to apply, ambiguous cases were much more common than we thought. In this section we focus on discussing our most important challenges during the coding process.

### 5.1 Choosing the Right Level & Members for our Typology

During our discussions we made several failed attempts at deriving a typology that we could apply to images without knowing details about the data they represented or what construction rules led to them.

**Bertin’s marks and channels as inspiration.** One of our first attempts was to use Bertin’s semiology of graphics and in particular his marks and visual variables for describing visualizations [8]. Using this approach resulted in numerous (low-level) codes per image that together did not allow us to meaningfully describe what we saw. For example, a scatterplot would be coded as point marks with a position encoding (visual variable). What we wanted to establish were instead higher-level categories that would include both the graphical primitives used and the coordinate system. Still, Bertin’s definition of visual variables and marks provided inspiration that we see in the naming of several of our visualization types. For example, we developed a more general “point-based representations” category that covers scatterplot-like images. In this category, the visual mark of a point would be dominant, perceptually.

**Visualization Techniques as a Typology?** Ward once said “I’ll Know it When I See it” [99]. We rely on our observations to help us derive knowledge about data even when we do not know what the data is in detail. Yet, our second failed attempt at arriving a typology began from trying to identify dedicated visualization techniques in the images we saw. We wanted to characterize the “output space”, the result space of encoding techniques. However, different data encoding techniques could result in similar visuals while the identification of some techniques required knowing the data that was being visualized. For instance, timeline visualizations could only be identified if the axis labels were clearly pointing out temporal data. We later completely abandoned the approach to use visualization techniques for building a typology so that we could focus on what the data representation actually looked like. What we retained from this failed attempt, however, was a focus on the central encoding technique. We intentionally chose not to code legends, labels, and embellishments etc.

**Is “continuous color-based encoding” a separate type?** Several of our visualization types make reference to Bertin’s visual channels position and size. For example grid-based encodings rely on specific positions for information layout. Point-, bar-, and area-based representations are distinguished in terms of how size encodes information. Point-based encodings primarily reference a position, bar- encodings a length, and area-based encodings a two-dimensional size. Despite the fact that properties of color such as hue, saturation, and luminance are frequently used to encode data as well, we did not have a dedicated visualization type for color-based encodings. Instead we initially coded most continuous and discrete heat-map type encodings under “generalized matrix / grid” with the argument that these encodings are applied on pixel grids. During Phase 6 of our codings, many discussions centered around the question of when a continuous color-based encoding should be coded as a grid - especially when it did not look at all like a grid. For example, when continuous colorscale were applied to 3D geometries such as streamlines, the matrix/grid encoding no longer seemed appropriate. Hence, after many discussions, we decided to create a distinct category to recognise the use of continuous colormaps.

### 5.2 General Coding Ambiguity

Many of the challenges with our typology of visualization types stemmed from ambiguities in how we should apply the code set we developed. Here, we discuss the most important of these challenges.

**Surfaces and volume rendering.** In an early visualization-type code set we had included surface and volume renderings as two separate codes. Several of us, however, found it very challenging to discern the differences between some volume and surface representation. For example, volume rendering may add thickness but it can also depict

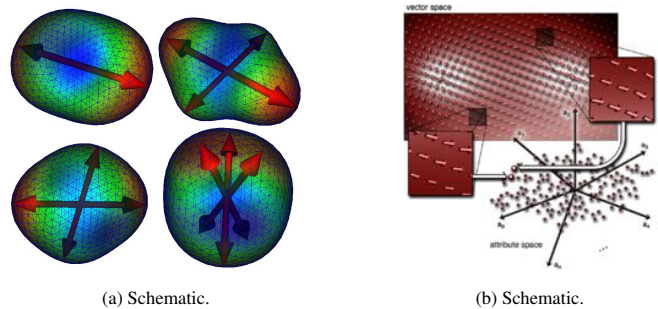


Fig. 5: Challenging cases of coding “schematic representations and concept illustrations”. (a) Is this a glyph-based or schematic representation? (b) We choose not to code the visualization type inside “schematic representations and concept illustrations”. (Images from (a) Hlawitschka & Scheuermann [42] and (b) Daniels et al. [24]. © IEEE.)

surfaces. We decided that “recognizing” an image type at this level of detail (e.g., whether surfaces are produced through volume rendering or surface construction) was not reliable and may not even be important in terms of describing a figure’s visual content as the underlying algorithmic techniques would be transparent to viewers. In machine learning, visualization, and computer graphics, it has been argued that high-level concepts directly contribute to reasoning [20, 96]. High-level concepts remove the specific details of a given technique and focuses on what all instances of that family of visualization type have in common. We thus chose a visualization type that is more general, i.e., surface and volume techniques combined.

**Ambiguous Area-based Images.** We tried to avoid using data types in our classification schema because we wanted the focus to be consistently on images. We originally had a cartographic map category and choose to remove it because it was both focus on a data type and a specific technique. Instead, we decided that the depiction of areas and their relationships was the underlying principle for cartographic maps but also other related techniques such as area charts, stream graphs, etc. However, there were exceptions: route maps, for example, where lines indicate a direct route, were coded separately as networks because the routes encoded topological relationships. One difficulty we encountered was the distinction between a “map” (cartographical map) and a “terrain” (wireframe or surface rendering). Conceptually, these were very similar. However, using visual appearance as our guide, map images generally appeared to delineate distinct areas while terrains showed smooth surfaces with evaluations. Another difficulty related to maps arose when maps were used as a reference structure for data representations layered on top, akin to how gridlines are used on scatterplots. In these cases we had to derive elaborate procedures to be consistent in our treatment of reference structures. We decided only to code the underlying maps if the visualization would seem to change in meaning or message. However, these decisions were very subjective and resulted in several coding inconsistencies.

**Ambiguous Schematic Images.** A large number of figures were schematic representations or concept illustrations. It was often challenging to differentiate between schematics and a demonstration of a visual encoding technique. We often had to abandon our initial goal to ignore what data was encoded to be able to say whether the representation showed a toy dataset. Toy datasets are common in schematic representations, however, the frequent absence of contextual information in images, such as coordinate axis, labels, or scales made the identification of toy datasets difficult. We found that many figures simply did not depict scales which aligns with observations by Cleveland in his review of graphics in other scientific journals [23].

We also struggled with the use of annotations in figures as an identification criteria for schematics. For example, Fig. 5a can be coded either as glyph-based in that it shows a mathematical tensor or as schematic to illustrate the authors’ design idea or mathematical function. A majority of the team members chose to code “schematic representations and concept illustrations”. Schematic image are often meant to be particularly pedagogical and, thus, include a number of labels and arrows or



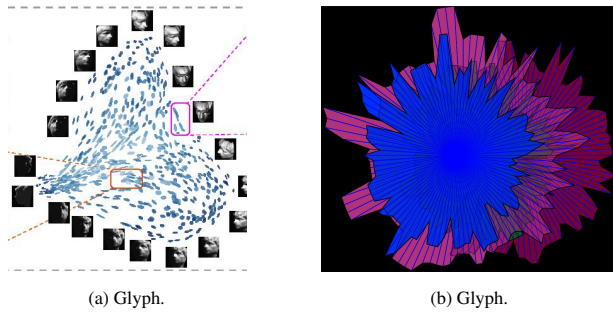



Fig. 6: Challenging cases of coding “glyph-based representations”. (a) The ellipsoids in the middle could be high-dimensional with two axes and thus orientation and magnitude. (b) features starGlyph-like dimensional comparisons and thus is a type of “glyph-based representations”. (Images from (a) Bian et al. [10] and (b) Fanea et al. [29]. © IEEE.)

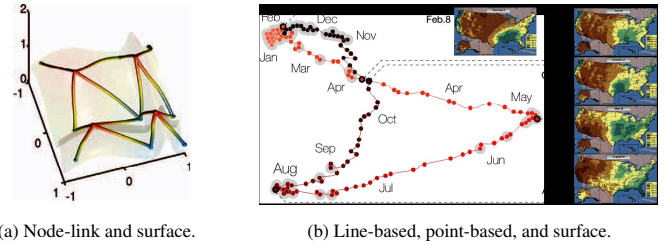
other annotations. We experimented with a specific coding guidelines in which we considered whether after the removal of annotations we were still seeing an example of a visual encoding type or not—in which case the image would have been a schematic. As such, we agreed that the appearance of labels and annotations to explain an image did not automatically mean the image was a schematic. Our general heuristic for schematics involved establishing if we saw 1) a well-known (or toy) dataset, 2) a pedagogical purpose, and 3) the illustration of a concept.

**Ambiguous Glyph Cases.** Glyphs are notoriously difficult to define. Recent definitions emphasize different aspects to delineate a glyph from other encodings. Fuchs et al. defined data glyphs as “data-driven visual entities, which make use of different visual channels to encode multiple attribute dimensions” [34]. Borgo et al. [11] followed Ward to define glyphs as “a visual representation of a piece of data where the attributes of a graphical entity are depicted by one or more attributes of a data record” [100]. Munzner’s glyph definition is broad and requires a data encoding to be assembled out of multiple marks that encode data [69]. For example, each single bar in a stacked bar chart  would be a “microglyph” according to Munzner because it is a composite object from multiple length-encoding marks. Throughout our coding we used a mix of the given definitions.

Coding “glyph-based representations” was challenging as glyphs were often associated with a placement strategy. For example, a focus on identifying a specific position is a property they share with point-based techniques. However, most commonly glyphs have been defined as representations of multi-variate data which by itself would not help to distinguish glyphs from other visualization types in our typology. A challenge is deciding when a point or other graphic primitive becomes a glyph. There is no standard definition to decide after how many data dimensions a single mark becomes a glyph and when a glyph becomes a chart. There seems to be, however, a general consensus that a glyph requires to reach a certain level of complexity to be categorized this way. However, reaching consensus on the level of complexity is challenging. We chose to label an image as “glyph-based representations” if there were multiple representation of data points that represented both position and additional data dimensions using color, shape, or other geometric primitives. Fig. 6 illustrates some difficult cases, when both coders scored the difficulty as “hard”.

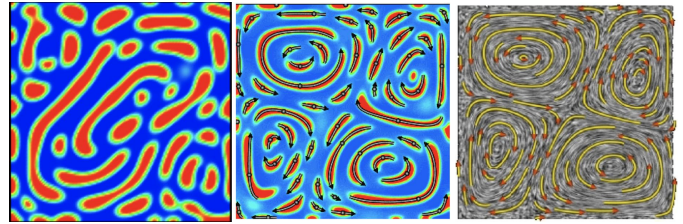
### 5.3 Multiple Encoding Ambiguity

Many images showed multiple visual encodings which is one of the primary reasons for coding inconsistencies we encountered. We agreed to code multiple visualization types, if these types were distinctive and could be perceived clearly. For example, if there were multiple visual designs layered or nested, that could be distinguished from one another, we tagged more than one encoding, e.g., node-link + area for Fig. 7a. We also decided when not to check multiple encodings. For example, the most frequent representation of a confidence interval includes a bar whose length represents the interval and a dot to represent the average. In this case we chose to code the bar as a primary encoding to which the average was considered and annotation. However, these decisions



(a) Node-link and surface. (b) Line-based, point-based, and surface.

Fig. 7: Challenging multiple coding ambiguity. Here, the two code signifies different aspects of the data and can be separated to stand alone. As a result, multiple codes apply. (Images from (a) Suits et al. [87] and (b) Bach et al. [5]. © IEEE.)



(a) Continuous-color. (b) Continuous-color & glyph. (c) Pattern-based & glyph.

Fig. 8: Challenging multiple coding ambiguity. Here, we did not code “surface-based representations and volumes” and only chose the primary code, which differentiates these visualization techniques (Images from (a) Weiskopf et al. [102] and (b,c) Garcke et al. [35]. © IEEE.)

were difficult, often inconsistent, and error-prone (for example, the dot for the average is not merely an annotation when confidence intervals are not symmetric).

**Influence of Coder Expertise** Though we chose to classify images without precise knowledge of the underlying data nor the data type (we purposefully ignored the figure captions), coder expertise often still played a major role in resolving ambiguities. For some images, we were familiar with the original papers, but for most cases of ambiguity, experience may have helped tremendously in understanding the intention behind images; especially for schematics. In a sense, our decision of coding an image depended not only on the perceived structure but also the intended function of the image. For example, scarf plots [84] were categorized as matrices rather than point-based encodings because we considered space for lines to be samples along a grid.

## 6 DISCUSSION

This section presents our reflection on our two-year coding process, limitations, and future work.

### 6.1 From Specific Techniques to A General Typology

Essentially, our coding experiment evaluated if visualizations (in academic publications) can be easily understood and categorized by experienced researchers. While we started with the intuition of finding categories based on data, tasks, and low-level encoding principles (characterization of the “input space”), we ended up scraping this and came up with a typology of the result or output space of images. Some of the original categories survived (point-based, line-based, and generalized area; surface-based and text-based; as well as node-link and glyph) and further new ones have emerged (bar representations, matrix/grid, continuous patterns, continuous color). Our typology also combines constructive features (low-level design space construction elements) with functional characteristics from the output space, especially for images that contained schematics and GUIs.

The design space elements are perhaps close to psychophysics—evaluations are well posed and hypotheses can be tested with an (often-lab-controlled) experiment. Resulting images, on the other hand, have much more to do with the viewers’ knowledge and context, and has its footing in vision science and even machine learning. What-we-see

could heavily influence how we act to choose to see next, to provide another angle to understand visualization effectiveness in the future.

Furthermore, implicit in teaching and learning tasks such as “show me the node-link diagrams” are much deeper issues involving the notion of what is meant by “node-link diagrams”. This meaning would vary with spatial (e.g. topological connectivity) and non-spatial data (e.g., social networks), context of use, and observers. We found that student coders involved in the earlier stage of this project had dramatically different understandings of author keywords (often with misconceptions). Also, our initial exercise (which lasted for more than a year) of measuring *visual design terms of authors or of low-level features* has largely been challenged by low-level details of naming techniques rather than what the visualizations show us.

## 6.2 What We See and Speed of Human Reasoning

The most significant contribution of our work is the derivation of a small set of categories that attempt to cover all techniques (completely). By merely seeing a figure, we purposely avoid focus on the data but rather focus on interpretation of visual design alone.

Our typology might be the first typology easily accessible to draw clear boundaries between visualization image types. Intuitively, a bar chart can be encoded as position or height, but users may not relate them to low-level taxonomy. Describing a bar chart as position and length may not be as accessible to the general audience as it is for a visual design expert. Similarly, it is easier to describe a histogram as a more typical “generalized bar representations” than a “distribution plot”. Murphy calls effects like this ‘typicality’ [70] for any task requiring relating an item to its categorical concepts, using typical terms encourages learning and usefulness for inference tasks. These categorical concepts further explain how we can understand objects we have never seen before. and extrapolate new categories from a few given examples. This current work may partially reflect this, since the expert coders could not agree on detailed categories when specific techniques were used. Expert coders also could not always visually distinguish between techniques. Some reasoning, such as volume rendering is fuzzier than surface rendering, does not fit all cases and coders did not feel confident in categorizing these. Even volume graphics experts made frequent mistakes except for those images they knew a priori. Coding accuracy is heavily influenced by the coder’s own expertise and experience in a given area. Such a disagreement suggests that visualization techniques may not be as easily accessible as we think.

## 6.3 Limitations and Future Work

Describing visual images at this what-we-see level—matching human intuition has not been studied in-depth previously. Tufte mentioned that “when principles of design replicate principles of thought, the act of arranging information becomes an act of insight” (189, p. 9)). The difficult question is whether we can ask visualization scholars to imitate this way of reasoning: to interpret an image in connecting a visualization to the designers’ intent. The potential next step is to invite the community to use our typology to study / refine categories.

The evaluation of our typology is done through its methodology (initial coding pass, multiple coders resolve conflicts, codes being iterated and discussed over the course of approximately two years). Additional validation, e.g. through external researchers that were not part of the coding team, would be a good next step, but is outside the scope of this paper. Our framework might potentially be used to compare and analyze what humans and machines see differently from these visualization types.

## 7 CONCLUSION

This article reflects on our journey to define a new visualization typology using high-level categories. The journey began with community-defined visualization keywords. Two failed attempts to use technique keywords and low-level encodings finally results in our typology of visualization types that focuses on describing our community’s “output”. Our visualization types emerged from both structural and functional similarities of the images. Indeterminacy of hard cases reflects perceptual uncertainty but similarity in each category—this looks like

that—gives insight into the understanding of visualization techniques. The typology developed here could provide a potentially powerful framework for studying topics of interests in visualization, such as image retrieval.

## REFERENCES

- [1] S. Afzal, R. Maciejewski, Y. Jang, N. Elmqvist, and D. S. Ebert. Spatial text visualization using automatic typographic maps. *IEEE Trans Vis Comput Graph*, 18(12):2556–2564, 2012. doi: 10.1109/TVCG.2012.264
- [2] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-oriented Data*. Springer, London, 2011. doi: 10.1007/978-0-85729-079-3
- [3] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *Proc. InfoVis*, pp. 111–117. IEEE Computer Society, Los Alamitos, 2005. doi: 10.1109/INFVIS.2005.1532136
- [4] T. M. Athawale, B. Ma, E. Sakhaee, C. R. Johnson, and A. Entezari. Direct volume rendering with nonparametric models of uncertainty. *IEEE Trans Vis Comput Graph*, 27(2):1797–1807, 2021. doi: 10.1109/TVCG.2020.3030394
- [5] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Trans Vis Comput Graph*, 22(1):559–568, 2016. doi: 10.1109/TVCG.2015.2467851
- [6] M. Bar. The proactive brain: Using analogies and associations to generate predictions. *Trends Cognit Sci*, 11(7):280–289, 2007. doi: 10.1016/j.tics.2007.05.005
- [7] L. D. Bergman, B. E. Rogowitz, and L. A. Treinish. A rule-based tool for assisting colormap selection. In *Proc. Visualization*, pp. 118–125. IEEE Computer Society, Los Alamitos, 1995. doi: 10.1109/VISUAL.1995.480803
- [8] J. Bertin. *Semiology of Graphics: Diagrams, Networks, Maps*. Esri Press, 1983.
- [9] P. Bhaniramka, R. Wenger, and R. Crawfis. Isosurfacing in higher dimensions. In *Proc. Visualization*, pp. 267–273. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/VISUAL.2000.885704
- [10] R. Bian, Y. Xue, L. Zhou, J. Zhang, B. Chen, D. Weiskopf, and Y. Wang. Implicit multidimensional projection of local subspaces. *IEEE Trans Vis Comput Graph*, 27(2):1558–1568, 2021. doi: 10.1109/TVCG.2020.3030368
- [11] R. Borgo, J. Kehrler, D. H. Chung, E. Maguire, R. S. Laramée, H. Hauser, M. Ward, and M. Chen. Glyph-based visualization: Foundations, design guidelines, techniques and applications. In *Eurographics State of the Art Reports*, pp. 39–63. The Eurographics Association, Goslar, 2013. doi: 10.2312/conf/EG2013/stars/039-063
- [12] M. A. Borkin, A. A. Vo, Z. Bylinskii, P. Isola, S. Sunkavalli, A. Oliva, and H. Pfister. What makes a visualization memorable? *IEEE Trans Vis Comput Graph*, 19(12):2306–2315, 2013. doi: 10.1109/TVCG.2013.234
- [13] M. Bostock, V. Ogievetsky, and J. Heer. D<sup>3</sup>: Data-driven documents. *IEEE Trans Vis Comput Graph*, 17(12):2301–2309, 2011. doi: 10.1109/TVCG.2011.185
- [14] W. C. Brinton. *Graphic Presentation*. Brinton Assoc., New York, 1939.
- [15] K. W. Brodlie. Visualization techniques. In *Scientific Visualization: Techniques and Applications*, chap. 3, pp. 37–85. Springer, Berlin, 2012. doi: 10.1007/978-3-642-76942-9\_3
- [16] C. Bu, Q. Zhang, Q. Wang, J. Zhang, M. Sedlmair, O. Deussen, and Y. Wang. Sinestream: Improving the readability of streamgraphs by minimizing sine illusion effects. *IEEE Trans Vis Comput Graph*, 27(2):1634–1643, 2021. doi: 10.1109/TVCG.2020.3030404
- [17] J. Canny. A computational approach to edge detection. *IEEE Trans Pattern Anal Mach Intell*, 8(6):679–698, 1986. doi: 10.1109/TPAMI.1986.4767851
- [18] K. Charmaz. *Constructing Grounded Theory: A Practical Guide through Qualitative Analysis*. Sage, 2006.
- [19] C. Chen, F. Ibeke-SanJuan, E. SanJuan, and C. Weaver. Visual analysis of conflicting opinions. In *Proc. VAST*, pp. 59–66. IEEE Computer Society, Los Alamitos, 2006. doi: 10.1109/VAST.2006.261431
- [20] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su. This looks like that: Deep learning for interpretable image recognition. In *Proc. NeurIPS*, pp. 8930–8941. Curran Associates, Inc., Red Hook, NY, 2019. doi: 10.48550/arXiv.1806.10574
- [21] J. Chen, M. Ling, R. Li, P. Isenberg, T. Isenberg, M. Sedlmair, T. Möller, R. Laramée, H.-W. Shen, K. Wünsche, and Q. Wang. IEEE VIS figures and tables image dataset. Dataset and online search, visimagenavigator.github.io, 2020. doi: 10.21227/4hy6-vh52

- [22] J. Chen, M. Ling, R. Li, P. Isenberg, T. Isenberg, M. Sedlmair, T. Möller, R. S. Laramée, H.-W. Shen, K. Wünsche, and Q. Wang. VIS30K: A collection of figures and tables from IEEE visualization conference publications. *IEEE Trans Vis Comput Graph*, 27(9):3826–3833, 2021. doi: 10.1109/TVCG.2021.3054916
- [23] W. S. Cleveland and R. McGill. Graphical perception and graphical methods for analyzing scientific data. *Science*, 229(4716):828–833, 1985. doi: 10.1126/science.229.4716.828
- [24] J. Daniels, II, E. W. Anderson, L. G. Nonato, and C. T. Silva. Interactive vector field feature identification. *IEEE Trans Vis Comput Graph*, 16(6):1560–1568, 2010. doi: 10.1109/TVCG.2010.170
- [25] K. Dasu, K.-L. Ma, J. Ma, and J. Frazier. Sea of Genes: A reflection on visualising metagenomic data for museums. *IEEE Trans Vis Comput Graph*, 27(2):935–945, 2021. doi: 10.1109/TVCG.2020.3030412
- [26] Ç. Demiralp, J. F. Hughes, and D. H. Laidlaw. Coloring 3D line fields using Boy’s real projective plane immersion. *IEEE Trans Vis Comput Graph*, 15(6):1457–1464, 2009. doi: 10.1109/TVCG.2009.125
- [27] E. Dimara and J. Stasko. A critical reflection on visualization research: Where do decision making tasks hide? *IEEE Trans Vis Comput Graph*, 28(1):1128–1138, 2022. doi: 10.1109/TVCG.2021.3114813
- [28] M. Edmunds, R. S. Laramée, G. Chen, N. Max, E. Zhang, and C. Ware. Surface-based flow visualization. *Comput Graph*, 36(8):974–990, 2012. doi: 10.1016/j.cag.2012.07.006
- [29] E. Fanea, S. Carpendale, and T. Isenberg. An interactive 3D integration of parallel coordinates and star glyphs. In *Proc. InfoVis*, pp. 149–156. IEEE Computer Society, Los Alamitos, 2005. doi: 10.1109/INFOVIS.2005.5
- [30] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans Pattern Anal Mach Intell*, 32(9):1627–1645, 2010. doi: 10.1109/TPAMI.2009.167
- [31] Z. Feng, H. Li, W. Zeng, S.-H. Yang, and H. Qu. Topology density map for urban data visualization and analysis. *IEEE Trans Vis Comput Graph*, 27(2):828–838, 2021. doi: 10.1109/TVCG.2020.3030469
- [32] D. Fisher. Hotmap: Looking at geographic attention. *IEEE Trans Vis Comput Graph*, 13(6):1184–1191, 2007. doi: 10.1109/TVCG.2007.70561
- [33] D. Fisher and M. Meyer. *Making Data Visual*. O’Reilly Media, 2018.
- [34] J. Fuchs, F. Fischer, F. Mansmann, E. Bertini, and P. Isenberg. Evaluation of alternative glyph designs for time series data in a small multiple setting. In *Proc. CHI*, pp. 3237–3246. ACM, New York, 2013. doi: 10.1145/2470654.2466443
- [35] H. Garcke, T. Preußner, M. Rumpf, A. Telea, U. Weikard, and J. van Wijk. A continuous clustering method for vector fields. In *Proc. Visualization*, pp. 351–358. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/VISUAL2000.885715
- [36] T. Givón. Prototypes: Between Plato and Wittgenstein. In C. Craig, ed., *Noun Classes and Categorization*, pp. 77–102. John Benjamins Publishing, Amsterdam, 1986.
- [37] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. *Inf Vis*, 10(4):289–309, 2011. doi: 10.1177/1473871611416549
- [38] B. F. Gregorski, J. G. Senecal, M. A. Duchaineau, and K. I. Joy. Adaptive extraction of time-varying isosurfaces. *IEEE Trans Vis Comput Graph*, 10(6):683–694, 2004. doi: 10.1109/TVCG.2004.35
- [39] C. D. Hansen and C. R. Johnson, eds. *The Visualization Handbook*. Elsevier, Oxford, 2005. doi: 10.1016/B978-0-12-387582-2.500514
- [40] J. Heer, M. Bostock, and V. Ogievetsky. A tour through the visualization zoo: A survey of powerful visualization techniques, from the obvious to the obscure. *Queue*, 8(5):20–30, 2010. doi: 10.1145/1794514.1805128
- [41] M. Hlawatsch, P. Leube, W. Nowak, and D. Weiskopf. Flow radar glyphs—Static visualization of unsteady flow with uncertainty. *IEEE Trans Vis Comput Graph*, 17(12):1949–1958, 2011. doi: 10.1109/TVCG.2011.203
- [42] M. Hlawatschka and G. Scheuermann. HOT-lines: Tracking lines in higher order tensor fields. In *Proc. Visualization*, pp. 27–34. IEEE Computer Society, Los Alamitos, 2005. doi: 10.1109/VISUAL2005.1532773
- [43] M. Hummel, C. Garth, B. Hamann, H. Hagen, and K. I. Joy. Iris: Illustrative rendering for integral surfaces. *IEEE Trans Vis Comput Graph*, 16(6):1319–1328, 2010. doi: 10.1109/TVCG.2010.173
- [44] S. Ingram, T. Munzner, V. Irvine, M. Tory, S. Bergner, and T. Möller. DimStiller: Workflows for dimensional analysis and reduction. In *Proc. VAST*, pp. 3–10. IEEE Computer Society, Los Alamitos, 2010. doi: 10.1109/VAST.2010.5652392
- [45] P. Isenberg, D. Fisher, M. R. Morris, K. Inkpen, and M. Czerwinski. An exploratory study of co-located collaborative visual analytics around a tabletop display. In *Proc. VAST*, pp. 179–186. IEEE Computer Society, Los Alamitos, 2010. doi: 10.1109/VAST.2010.5652880
- [46] P. Isenberg, F. Heimerl, S. Koch, T. Isenberg, P. Xu, C. D. Stolper, M. Sedlmair, J. Chen, T. Möller, and J. Stasko. Vispubdata.org: A metadata collection about IEEE visualization (VIS) publications. *IEEE Trans Vis Comput Graph*, 23(9):2199–2206, 2017. doi: 10.1109/TVCG.2016.2615308
- [47] P. Isenberg, T. Isenberg, M. Sedlmair, J. Chen, and T. Möller. Visualization as seen through its research paper keywords. *IEEE Trans Vis Comput Graph*, 23(1):771–780, 2017. doi: 10.1109/TVCG.2016.2598827
- [48] T. Isenberg, P. Isenberg, J. Chen, M. Sedlmair, and T. Möller. A systematic review on the practice of evaluating visualization. *IEEE Trans Vis Comput Graph*, 19(12):2818–2827, 2013. doi: 10.1109/TVCG.2013.126
- [49] W.-K. Jeong, J. Schneider, S. Turney, B. E. Faulkner-Jones, D. Meyer, R. Westermann, R. C. Reid, J. Lichtman, and H. Pfister. Interactive histology of large-scale biomedical image stacks. *IEEE Trans Vis Comput Graph*, 16(6):1386–1395, 2010. doi: 10.1109/TVCG.2010.168
- [50] X. Jiang and J. Zhang. A text visualization method for cross-domain research topic mining. *J Vis*, 19(3):561–576, 2016. doi: 10.1007/s12650-015-0323-9
- [51] B. Jobard, G. Erlebacher, and M. Y. Hussaini. Hardware-accelerated texture advection for unsteady flow visualization. In *Proc. Visualization*, pp. 155–162. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/VISUAL2000.885689
- [52] S. Kim, I. Woo, R. Maciejewski, D. S. Ebert, T. D. Ropp, and K. Thomas. Evaluating the effectiveness of visualization techniques for schematic diagrams in maintenance tasks. In *Proc. APGV*, pp. 33–40. ACM, New York, 2010. doi: 10.1145/1836248.1836254
- [53] Y. Kim, J. Kim, H. Jeon, Y.-H. Kim, H. Song, B. Kim, and J. Seo. Github: Visual analytics for understanding software development history through Git metadata analysis. *IEEE Trans Vis Comput Graph*, 27(2):656–666, 2021. doi: 10.1109/TVCG.2020.3030414
- [54] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun ACM*, 60(6):84–90, 2017. doi: 10.1145/3065386
- [55] J. K. Kruschke. ALCOVE: An exemplar-based connectionist model of category learning. *Psychol Rev*, 99(1):22–44, 1992. doi: 10.1037/0033-295x.99.1.22
- [56] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Trans Vis Comput Graph*, 18(9):1520–1536, 2012. doi: 10.1109/TVCG.2011.279
- [57] B. Latour. Wizualizacja i poznanie: zryswywanie rzeczy razem (Visualisation and cognition: Drawing things together). *Avant Trends Interdiscip Stud*, 3(T):207–257, 2012.
- [58] B. Lee, D. Brown, B. Lee, C. Hurter, S. Drucker, and T. Dwyer. Data visceralization: Enabling deeper understanding of data using virtual reality. *IEEE Trans Vis Comput Graph*, 27(2):1095–1105, 2021. doi: 10.1109/TVCG.2020.3030435
- [59] B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale. Sparkclouds: Visualizing trends in tag clouds. *IEEE Trans Vis Comput Graph*, 16(6):1182–1189, 2010. doi: 10.1109/TVCG.2010.194
- [60] G. L. Lohse, K. Biolsi, N. Walker, and H. H. Rueter. A classification of visual representations. *Commun ACM*, 37(12):36–50, 1994. doi: 10.1145/198366.198376
- [61] R. Maciejewski. Data representations, transformations, and statistics for visual reasoning. *Synth Lect Vis*, 2(1):1–85, 2011. doi: 10.2200/S00357ED1V01Y201105VIS002
- [62] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans Graph*, 5(2):110–141, 1986. doi: 10.1145/22949.22950
- [63] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Trans Vis Comput Graph*, 13(6):1137–1144, 2007. doi: 10.1109/TVCG.2007.70594
- [64] J. Malik and R. Rosenholtz. Computing local surface orientation and shape from texture for curved surfaces. *Int J Comput Vision*, 23(2):149–168, 1997. doi: 10.1023/A:1007958829620
- [65] T. Malisiewicz and A. A. Efros. Recognition by association via learning per-exemplar distances. In *Proc. CVPR*, pp. 42:1–42:8. IEEE Computer Society, Los Alamitos, 2008. doi: 10.1109/CVPR.2008.4587462
- [66] D. L. Medin, R. L. Goldstone, and D. Gentner. Respects for similarity. *Psychol Rev*, 100(2):254–278, 1993. doi: 10.1037/0033-295X.100.2.254
- [67] D. L. Medin and M. M. Schaffer. Context theory of classification learning.

- Psychol Rev*, 85(3):207–238, 1978. doi: 10.1037/0033-295X.85.3.207
- [68] W. Meulemans, M. Sondag, and B. Speckmann. A simple pipeline for coherent grid maps. *IEEE Trans Vis Comput Graph*, 27(2):1236–1246, 2021. doi: 10.1109/TVCG.2020.3028953
- [69] T. Munzner. *Visualization Analysis and Design*. CRC Press, New York, 2014. doi: 10.1201/b17511
- [70] G. Murphy. *The Big Book of Concepts*. MIT Press, Cambridge, 2004.
- [71] N. Nguyen, O. Strnad, T. Klein, D. Luo, R. Alharbi, P. Wonka, M. Maritan, P. Mindek, L. Autin, D. S. Goodsell, and I. Viola. Modeling in the time of COVID-19: Statistical and rule-based mesoscale models. *IEEE Trans Vis Comput Graph*, 27(2):722–732, 2021. doi: 10.1109/TVCG.2020.3030415
- [72] G. M. Nielson and B. Hamann. The asymptotic decider: Removing the ambiguity in marching cubes. In *Proc. Visualization*, pp. 83–91. IEEE Computer Society, Los Alamitos, 1991. doi: 10.1109/VISUAL.1991.175782
- [73] R. M. Nosofsky. Attention, similarity, and the identification-categorization relationship. *J Exp Psychol Anim Learn Cognit: Gener*, 115(1):39–57, 1986. doi: 10.1037/0096-3445.115.1.39
- [74] A. Pandey, U. H. Syeda, C. Shah, J. A. Guerra-Gomez, and M. Borkin. A state-of-the-art survey of tasks for tree design and evaluation with a curated task dataset. *IEEE Trans Vis Comput Graph*, 28(10):3563–3584, 2022. doi: 10.1109/TVCG.2021.3064037
- [75] D. Phan, L. Xiao, R. Yeh, and P. Hanrahan. Flow map layout. In *Proc. InfoVis*, pp. 219–224. IEEE Computer Society, Los Alamitos, 2005. doi: 10.1109/INFVIS.2005.1532150
- [76] P. Pu and D. Lalanne. Interactive problem solving via algorithm visualization. In *Proc. InfoVis*, pp. 145–153. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/INFVIS.2000.885103
- [77] H. C. Purchase, K. Isaacs, T. Buetti, B. Hastings, A. Kassam, A. Kim, and S. v. Hoesen. A classification of infographics. In *Proc. Diagrams*, pp. 210–218. Springer, Cham, 2018. doi: 10.1007/978-3-319-91376-6\_21
- [78] D. Rees and R. Laramee. A survey of information visualization books. *Comput Graph Forum*, 38(1):610–646, 2019. doi: 10.1111/cgf.13595
- [79] P. Rheingans, M. Marietta, and J. Nichols. Interactive 3D visualization of actual anatomy and simulated chemical time-course data for fish. In *Proc. Visualization*, pp. 393–396. IEEE Computer Society, Los Alamitos, 1995. doi: 10.1109/VISUAL.1995.485169
- [80] E. H. Rosch. Natural categories. *Cognit Psychol*, 4(3):328–350, 1973. doi: 10.1016/0010-0285(73)90017-0
- [81] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-Lite: A grammar of interactive graphics. *IEEE Trans Vis Comput Graph*, 23(1):341–350, 2017. doi: 10.1109/TVCG.2016.2599030
- [82] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Trans Vis Comput Graph*, 18(12):2431–2440, 2012. doi: 10.1109/TVCG.2012.213
- [83] B. Shneiderman and M. Wattenberg. Ordered treemap layouts. In *Proc. InfoVis*, pp. 73–78. IEEE Computer Society, Los Alamitos, 2001. doi: 10.1109/INFVIS.2001.963283
- [84] A. Somarakis, M. E. Ijsselstein, S. J. Luk, B. Kenkhuis, N. F. de Miranda, B. P. Lelieveldt, and T. Höllt. Visual cohort comparison for spatial single-cell omics-data. *IEEE Trans Vis Comput Graph*, 27(2):733–743, 2021. doi: 10.1109/TVCG.2020.3030336
- [85] Y. Song, J. Ye, N. Svakhine, S. Lasher-Trapp, M. Baldwin, and D. Ebert. An atmospheric visual analysis and exploration system. *IEEE Trans Vis Comput Graph*, 12(5):1157–1164, 2006. doi: 10.1109/TVCG.2006.117
- [86] C. Stoll, S. Gumhold, and H.-P. Seidel. Visualization with stylized line primitives. In *Proc. Visualization*, pp. 695–702. IEEE Computer Society, Los Alamitos, 2005. doi: 10.1109/VISUAL.2005.1532859
- [87] F. Suits, J. T. Klosowski, W. P. Horn, and G. Lecina. Simplification of surface annotations. In *Proc. Visualization*, pp. 235–242. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/VISUAL.2000.885700
- [88] A. Tikhonova, C. D. Correa, and K.-L. Ma. Visualization by proxy: A novel framework for deferred interaction with volume data. *IEEE Trans Vis Comput Graph*, 16(6):1551–1559, 2010. doi: 10.1109/TVCG.2010.215
- [89] E. R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press, Cheshire, 1998.
- [90] E. R. Tufte. *Envisioning Information*. Graphics Press, Cheshire, 2001.
- [91] D. Ushizima, D. Morozov, G. H. Weber, A. G. C. Bianchi, J. A. Sethian, and E. W. Bethel. Augmented topological descriptors of pore networks for material science. *IEEE Trans Vis Comput Graph*, 18(12):2041–2050, 2012. doi: 10.1109/TVCG.2012.200
- [92] J. J. van Wijk. Image based flow visualization for curved surfaces. In *Proc. Visualization*, pp. 123–130. IEEE Computer Society, Los Alamitos, 2003. doi: 10.1109/VISUAL.2003.1250363
- [93] V. Verma, D. Kao, and A. Pang. A flow-guided streamline seeding strategy. In *Proc. Visualization*, pp. 163–170. IEEE Computer Society, Los Alamitos, 2000. doi: 10.1109/VISUAL.2000.885690
- [94] J. von Engelhardt. *The Language of Graphics*. PhD thesis, University of Amsterdam, 2002. doi: 11245/1.208097
- [95] C. Wang, H. Yu, and K.-L. Ma. Importance-driven time-varying data visualization. *IEEE Trans Vis Comput Graph*, 14(6):1547–1554, 2008. doi: 10.1109/TVCG.2008.140
- [96] X. Wang, Z. Bylinskii, A. Hertzmann, and R. Pepperell. Toward quantifying ambiguities in artistic images. *ACM Trans Appl Percept*, 17(4):13:1–13:10, 2020. doi: 10.1145/3418054
- [97] Y. Wang, X. Chu, C. Bao, L. Zhu, O. Deussen, B. Chen, and M. Sedlmair. EdWordle: Consistency-preserving word cloud editing. *IEEE Trans Vis Comput Graph*, 24(1):647–656, 2017. doi: 10.1109/TVCG.2017.2745859
- [98] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans Image Process*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861
- [99] M. O. Ward. A taxonomy of glyph placement strategies for multidimensional data visualization. *Inf Vis*, 1(3–4):194–210, 2002. doi: 10.1057/palgrave.ivs.9500025
- [100] M. O. Ward. Multivariate data glyphs: Principles and practice. In *Handbook of Data Visualization*, pp. 179–198. Springer, Berlin, 2008. doi: 10.1007/978-3-540-33037-0\_8
- [101] M. O. Ward, G. Grinstein, and D. Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications*. A K Peters/CRC Press, New York, 2<sup>nd</sup> ed., 2015. doi: 10.1201/b18379
- [102] D. Weiskopf and G. Erlebacher. Overview of flow visualization. In Hansen and Johnson [39], chap. 12, pp. 261–278. doi: 10.1016/B978-012387582-2/50014-9
- [103] M. Weiß, K. Angerbauer, A. Voit, M. Schwarzl, M. Sedlmair, and S. Mayer. Revisited: Comparison of empirical methods to evaluate visualizations supporting crafting and assembly purposes. *IEEE Trans Vis Comput Graph*, 27(2):1204–1213, 2021. doi: 10.1109/TVCG.2020.3030400
- [104] L. Wilkinson. *The Grammar of Graphics*. Springer, New York, 2<sup>nd</sup> ed., 2005. doi: 10.1007/0-387-28695-0
- [105] L. Wittgenstein. *Philosophical Investigations*. John Wiley & Sons, 1953.
- [106] J. Xia, T. Chen, L. Zhang, W. Chen, Y. Chen, X. Zhang, C. Xie, and T. Schreck. SMAP: A joint dimensionality reduction scheme for secure multi-party visualization. In *Proc. VAST*, pp. 107–118. IEEE Computer Society, Los Alamitos, 2020. doi: 10.1109/VAST50239.2020.00015
- [107] S. Ye, Z. Chen, X. Chu, Y. Wang, S. Fu, L. Shen, K. Zhou, and Y. Wu. ShuttleSpace: Exploring and analyzing movement trajectory in immersive visualization. *IEEE Trans Vis Comput Graph*, 27(2):860–869, 2021. doi: 10.1109/TVCG.2020.3030392
- [108] J. S. Yi, Y. ah Kang, J. Stasko, and J. A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Trans Vis Comput Graph*, 13(6):1224–1231, 2007. doi: 10.1109/TVCG.2007.70515
- [109] V. Yoghoudjian, Y. Yang, T. Dwyer, L. Lawrence, M. Wybrow, and K. Marriott. Scalability of network visualisation from a cognitive load perspective. *IEEE Trans Vis Comput Graph*, 27(2):1677–1687, 2021. doi: 10.1109/TVCG.2020.3030459
- [110] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, pp. 586–595. IEEE Computer Society, Los Alamitos, 2018. doi: 10.1109/CVPR.2018.00068

# Not As Easy As You Think—Experiences and Lessons Learnt from Creating a Visualization Image Typology

## Additional material

While the main document contains the main aspects of employed techniques and results, this supplemental material aims at providing exhaustive and reproducible experimental details.

### A INPUT FROM THIS WORK: FIGURES

The first six years of the seven year’s data come from the collection of figure and table data from IEEE VIS publications from 1990–2019. We added the 2020 dataset to VIS30K data collection [21] using their model. In doing this, we reused the authors’ approach and their meta-data to remove all tables (unless part of a figure) and their open-source model and tools to extract and clean the new figure data of 2020, after scraping all papers from the IEEE site. The first author collected and cleaned the image data manually afterward. The final image set was checked by other VIS30K co-authors. After collecting these source data, we determined a criterion for visualization characterization that underlies our quantitative analysis of figure content analysis. We coded, refined, recoded, validated, and analyzed the figure content (see Table 1 for the final types).

### B ON THE ANALYSIS OF JUDGING VISUALIZATION TYPES: REPRESENTATION TRANSITIONS AND HEURISTICS (ADDITIONAL RESULTS)

**The top-21 keywords by authors and the four functional purposes of the images.** The top-21 keywords are both *frequently occurring* and *distinguishable* from one another. This initial process resulted in the following list. *bar chart, cartographic map, circular node-link tree, flow chart, flowline, glyphs, heatmap, isosurface rendering, line chart, matrix, node-link diagram, parallel coordinate plot, pie chart, point cloud, scatter plot, tag cloud, timeline, treemap, volume rendering, Voronoi diagram, and wireframe rendering.* An “other” tag is added to collect any other techniques that are not on the list.

Our initial purpose classification featured four categories: *F1. rendering illustrating a visualization technique, F2. result presentation from a quantitative evaluation, and F3. screenshot of system or graphical user interface (GUI). F4. photo of a real-world physical scene.* These functions are chosen to mainly identify quantitative charts (F1) which have a long history of basic research development, (F1) algorithm results largely from scientific visualization communities (F3) techniques from VAST interface papers, and (F4) photo used to show real-world imagery. Images not in these categories were added to the “other” category.

#### B.1 Code Changes Between Phases

The birth and death of these techniques and the transition to typology are shown in other supplemental materials. We have shared Instructions used by coders in other supplemental materials.

#### B.2 Additional Coding Choices: Concept evolution, rationale, self-correction, and agree to disagree: Early Coding 2006

**Not repeatable when characterizing by Authors’ keywords.** There is also a scalability issue, in that when a new technique is created, a new keyword would be added. It is not repeatable or even findable of these new solutions. In addition, focusing on specific cases could only tell us which specific instances of visualization techniques were used, but it did not identify cases to which the parts papers used were similar (e.g., both subburst and traditional bar chart ask people to reason with bar height.) Taxonomy is meant to be high-level describing a category of work but this does not exist. A good classification must be complete (to cover all designs possible) and relatively clean in that many designs fall into one category [37]. Organizing techniques into a hierarchical form to fit new techniques can facilitate learning and searching for popular techniques as well comparing the number of distinct techniques and related uses.

**We merge some categories if they contain common visual channels.** For example, flowlines, parallel coordinate plot, line chart all contain line drawings. *We merged them into the “line-based representations” category.* The flow chart category was prevalent and represent the largest number of figures in the other category from our 2006 coding, was thus added and later further expanded to “schematic representations and concept illustrations”. The original flow chart has only captured limited number of wiring diagrams. Many charts and diagrams in visualization papers are visually rich containing techniques to illustrate concepts. *We expanded flow chart to “General Schematic Representation, schematic images, schematic concept illustration”.*

**We ignored the drawing media used to visualize the data.** Equally challenging is whether or not views are hand-drawn or computer-generated. For example, schematic ones can be drawn by algorithms [52] and there used to be a research field known as “illustrative visualization” which developed algorithms that would render images in a style that appeared to be hand-drawn. The illustrations in the classical book of Bertin’s semiotics were also drawn by hand. Hence, we chose to ignore the media in the subsequent coding phase. Instead, we emphasize the elements in the figure rather than the drawing media. For example, a photograph of an environment (say a VR installation) would be coded as “3D”.

**We managed annotation, legend, and context.** One of the challenges was the treatment of context information, such as annotation marks, or color legends. Here, we simply focused on the primary visual encoding and agreed to not code such context information separately. Color legend is in “other” category. Context unless is relevant to the data is not coded. Some contextual data, such as geometry models or boundary conditions are important to understand the visualization techniques and were thus coded.

**We avoided including data types in the type names.** For example, scalar, vector, and tensor field visualization techniques can be defined using our typology without specifically mentioning flow fields or tensors. Thus, we removed this category. Text-based encodings is the only exemption to our premise of not coding data-type. However, it is such an esoteric type, that we felt it was justified in that case. We also believe that, in general, users of our typology will want to see this as a distinct category.

**Additional Results on 2D and 3D Uses for “surface-based representations and volumes” and “line-based representations”** are in Fig. 9. We see a decline of 3D use over time.

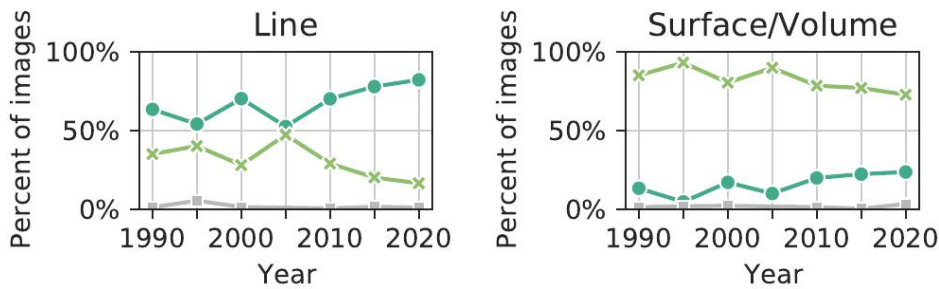


Fig. 9: Temporal overview of the proportions of 2D and 3D images for “surface-based representations and volumes” and “line-based representations”.

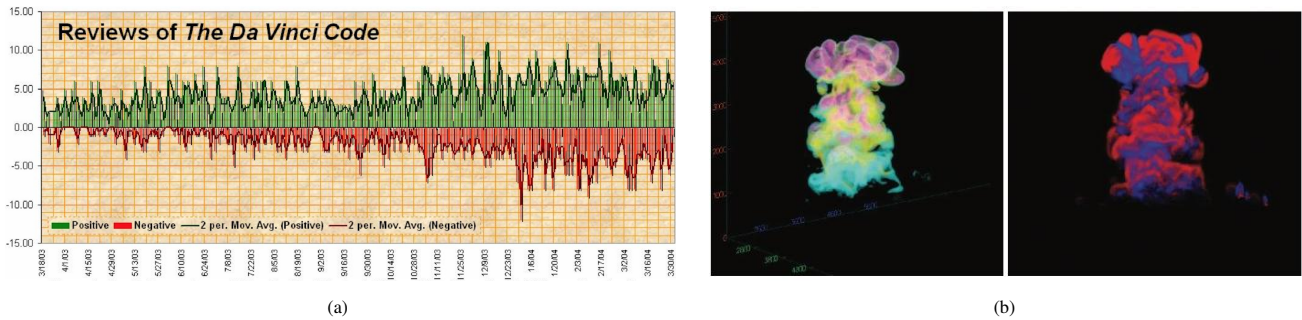


Fig. 10: **Rationale to remove timeline from our typology schema.** Is there a timeline in the figures? (a) yes because of the dates on the x-axis and (b) yes but not so obvious and the judgement counts on the coders’ knowledge. (Image courtesy of (a) Chen et al. [19] and (b) Song et al. [85].) © IEEE.

### C WEB INTERFACES FOR ANNOTATING FIGURES

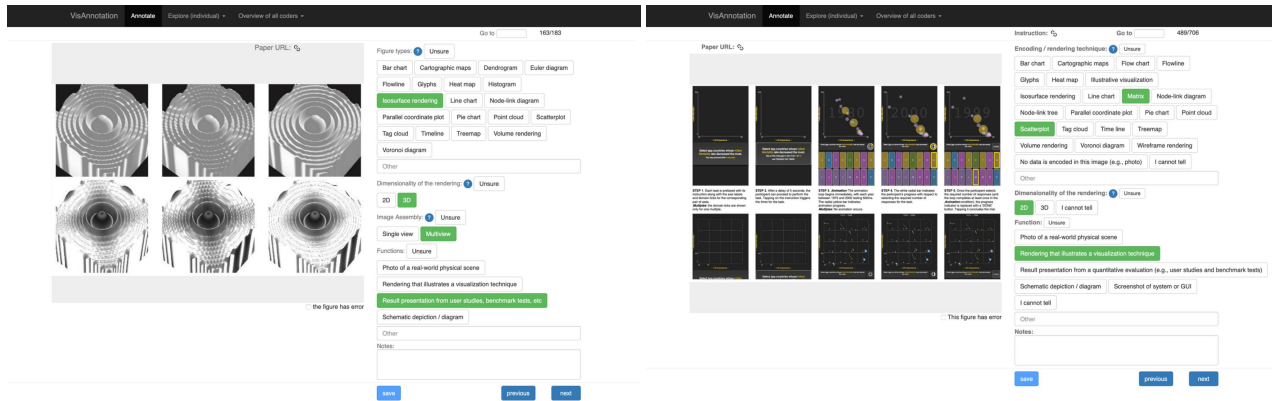
We annotated all images and discussed and compared our codings via our own web-based interfaces to support our collaborative work (Fig. 11 (a)-(d)). Our web-based tool automatically loads images and authorize the uses. The uses can tag the given image according to the terms, either keyword-based or type-based typology. On the back-end of our coding tool, we recorded every button click from each participant during the coding process for post-hoc analyses. For resolving code-consistency, we also implemented a comparative interface (Fig. 12) for us to resolve all coding conflicts. Again, all coders’ button clicks were recorded.

### D A VISUALIZATION CHARACTERIZATION TOOL

We built an exploration tool as an extension of VisImageNavigator (<https://visimagenavigator.github.io/>) (Fig. 13) for viewers to lookup, learn, and re-evaluate our 13 encoding categories and their dimensionality. We have augmented the exploration experience with image similarity, year, authors, and terms from paper abstract and keywords.

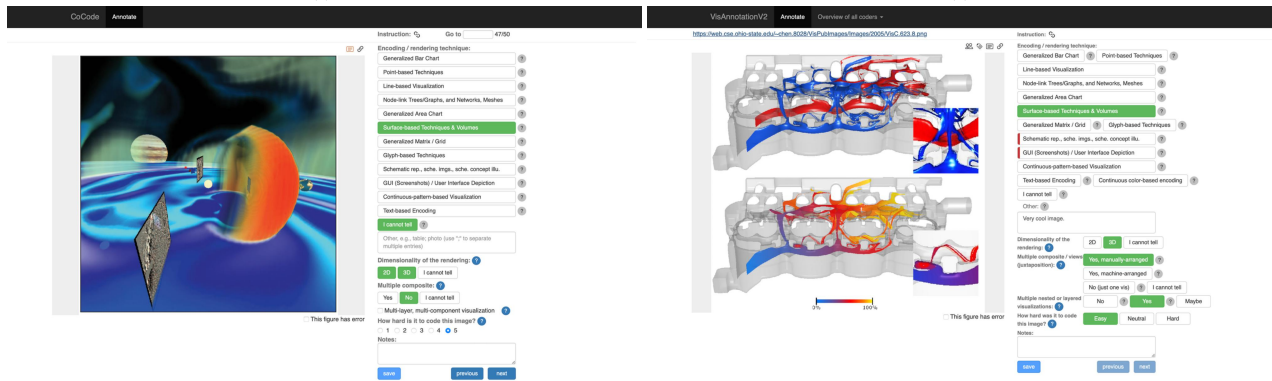
**Explore Morphological Structures and Form to Design New Techniques.** A figure may contain many pieces of information. To understand and design one’s own, attention must be directed to the key uses of techniques and visual details. Within a category of our high-level encoding, we may explore morphological variations in structure and form we purposely avoided in this work.

**Advance Education and Improve Visual literacy.** Since our categories are high-level, they should be easy to learn and memorize, thus a very exciting use is to dedicate it to educational uses. We can also use it to explore techniques in specific research areas. Some images and techniques, e.g., photos, might be more meaningful.



(a)

(b)



(c)

(d)

Fig. 11: Screenshot of Phase-1-2-3 and in Phases 4–5 Web interfaces. Phase 1 focused on techniques derived from authors’ keywords: The coders use the interfaces to code each image. The coding labels and categories were updated reflecting the results of weekly discussions. These label interfaces show that code revolution over time: from **Keyword-based** to **Type-based** updates.

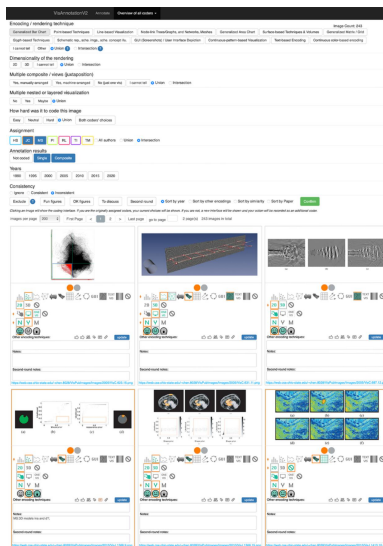


Fig. 12: Screenshots of Web Interfaces for the Coding in Phase 6 and for Resolving Coders’ Inconsistency through Pair-wise Comparison. Two coders results are shown together. For complex images, coders can also look up the images in the same paper coded by other coders through paper-based or image similarity-based search.

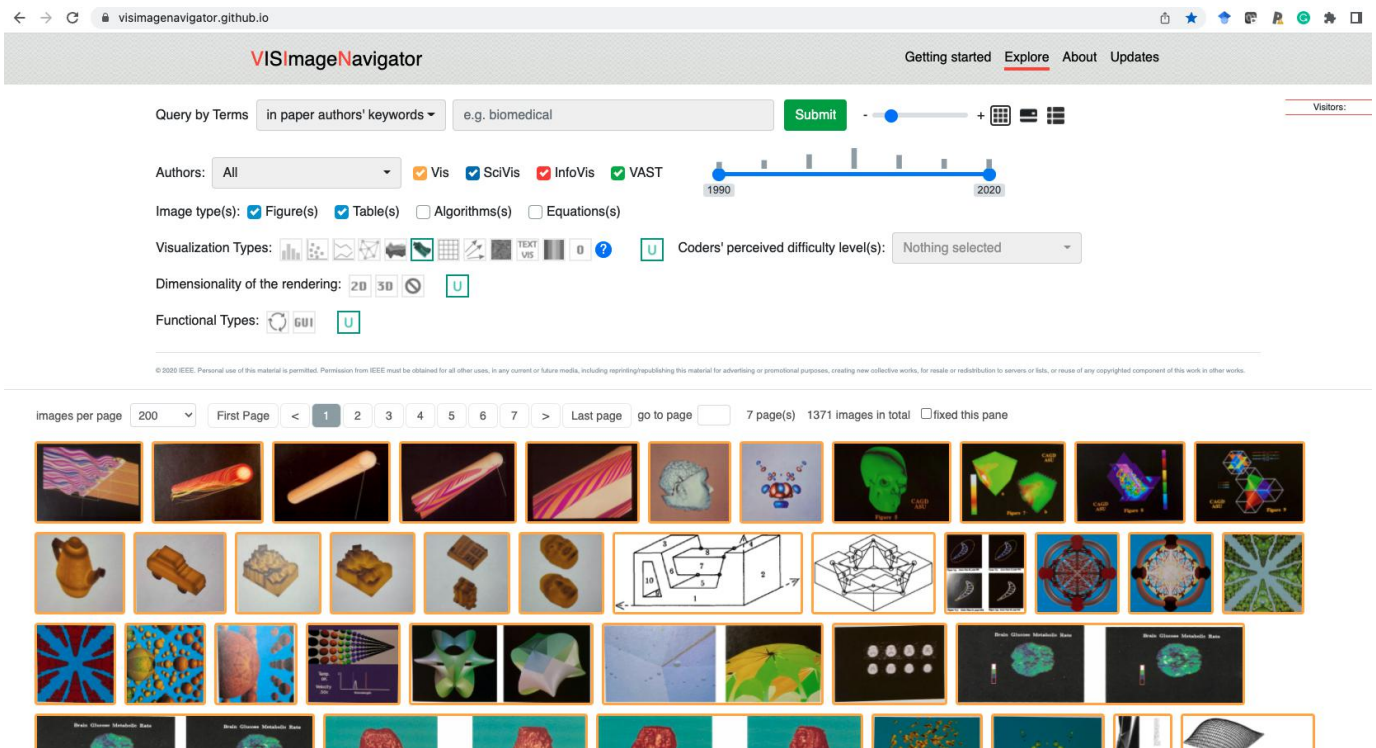


Fig. 13: Our exploratory tool is integrated with the VisImageNavigator (<https://visimagenavigator.github.io/>). Viewers can explore our codes of 9,039 labels by visualization types, functional types, dimensionality, and difficulty levels.