Ways of Visualizing Data on Curves

Benjamin Bach, Charles Perin, Qiuyuan Ren, Pierre Dragicevic

Transimage 2018

Proceedings of the 5th Biennial Transdisciplinary Imaging Conference 2018

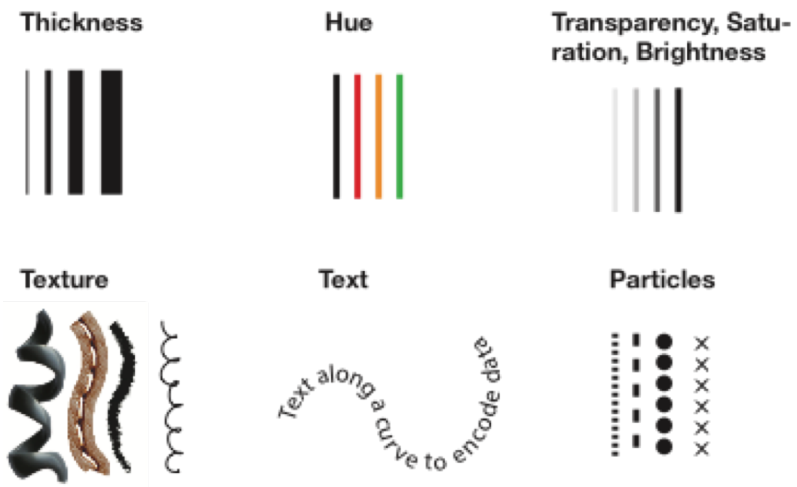
Ways of Visualizing Data on Curves

Benjamin Bach bbach@inf.ed.ac.uk
University of Edinburgh, Edinburgh, UK

Charles Perin charles.perin@free.fr
City University, London, UK

Qiuyuan Ren qiuyuanren1001@gmail.com
University of Edinburgh, Edinburgh, UK

Pierre Dragicevic pierre.dragicevic@inria.fr
Inria, Saclay, France



Visual variables used to encode data on the curve in the enrichment stage. Image: Qiuyuan Ren.

Bach, B., Ren, Q., Perin, C., Dragicevic, P. 2018. Ways of Visualizing Curves. In: Proceedings of the 5th Biennial Transdisciplinary Imaging Conference, TI2018, 18-20 April 2018, Edinburgh, UK. DOI: 10.6084/m9.figshare.6104705

Ways of Visualizing Data with Curves

Benjamin Bach
University of Edinburgh
Edinburgh, UK
bbach@inf.ed.ac.uk

Qiuyuan Ren
University of Edinburgh
Edinburgh, UK
qiuyuanren1001@gmail.com

Charles Perin
City University
London, UK
charles.perin@free.fr

Pierre Dragicevic
Inria
Saclay, France
pierre.dragicevic@inria.fr

Abstract
This paper reviews the many ways curves are used to encode data in information visualization. As part of our review, we introduce a curve-based visualization framework where data can be encoded in two major ways: *i)* through a curve's shape (a process we call *embedding*) and *ii)* through a curve's local visual attributes (a process we call *enrichment*). Our framework helps describing and organizing the rich design space of curve-based data visualizations, and offer inspiration for novel data visualizations.

Author Keywords
Information visualization, geographic visualization, temporal visualization, curves.

Introduction
For centuries, artists and scientists have exploited and commented on the rich expressive power offered by curves. For Kandinsky [17], a curve results from a point moving on a paper's surface; at any given point in time, the point can change its direction, speed, and appearance, thereby influencing the shape and final appearance of the curve. Moving from paper and canvases to computer-generated graphics opened up an even wider array of possibilities, creating a vast design space to tap into for communicating emotions, ideas, and facts. In this article, we focus on the use of computer-generated curves for communicating data.

Lines and other types of curves are pervasive in statistical charts and in information visualization (InfoVis), especially when visualizing temporal data [8]. Curves form the basis of elementary visualizations such as line charts [33], trajectory visualizations [2, 35], and connected scatterplots [15]. Meanwhile, new curve-based visualizations are routinely proposed to convey information such as network relations [3], patterns of narration in movies [18], or the sonic topology of poems [22].

Despite the pervasive use of curves, InfoVis researchers have only started to chart and study their design space in a systematic fashion [16, 25]. There is still a lack of general overviews of the many ways curves haven been used to (and can potentially be used to) encode data.

For Jacques Bertin [6], there are three major ways to visually represent data elements: through *points*, *lines* (i.e., curves), or *areas*. These are called *visual marks*. Once visual marks have been decided, *visual variables* such as size, luminance or texture can be used to encode data attributes. Some pairings between visual variables and visual marks are more sensible than others. For example, texture is best applied to areas, while size is best applied to points. Curves can accommodate a rich set of visual variables, and some visual variables have been specifically created for curves. These include beginning and ending symbols (e.g. arrow-heads), particles [25, 18], and parallel lines [18]. Curves also support many creative variations of classical visual encodings which can be either applied to the whole curve or to specific curve segments. As a result, there is a large design space for encoding data through curves.

In this article, we review how curves are used to convey information in InfoVis. Our review offers a catalog of prominent examples of information effectively communicated through curves, as well as a structured description of the different ways curves can be used to encode data. After introducing the general outline of our framework, we discuss how information can be conveyed through a curve's shape (*embedding*), and then discuss how information can be conveyed through a curve's local visual attributes (*enrichment*).

Methodology

To derive our framework, we collected visualizations and infographics that convey data through curves. Our search was initially guided by our general knowledge of InfoVis stemming from years of attending major visualization conferences, as well as teaching InfoVis classes to Master students. Additional resources we employed include Best American Infographics [12], Graphics Semiology [6], Dear Data [21], Information is Beautiful Awards, Visual Complexity [20], Brehmer's et al. survey on timelines [8], the Space-time cube survey [2], Pinterest and general Google Image search.

Our collection features classical visualizations such as line charts, node-link diagrams, and trajectory visualizations, as well as less common designs such as connected scatterplots [15] and time curves [4]. In an attempt to identify visual encodings that are unknown or underused in InfoVis, we also collected examples of curve depictions from graphic design and art.

In total, we collected over 60 images of visualizations, infographics, and artwork. This number being too large to fit a single article, we only present a selection of prominent and representative examples. An extensive list of our examples can be found online: http://dataoncurves.wordpress.com.

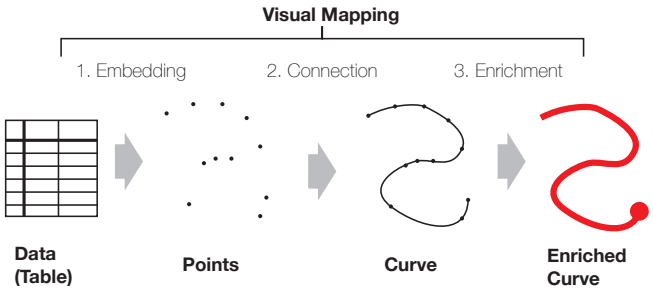


Figure 1: Pipeline for visualizing data through a curve (© B.Bach).

Framework for Curve Visualizations

For simplicity, we focus on single-curve visualizations although our framework easily expands to capture multi-curve visualizations. In such visualizations, data can be encoded in two major ways: *i)* using spatial embedding (i.e., through the curve's shape); and *ii)* using enrichment (i.e., enrich the style through the curve's visual attributes). The two methods can be combined.

Figure 1 illustrates the general process of encoding data with a curve. This process is data-agnostic as long as the dataset consists of an *ordered list of data elements.* A **data element** refers to an entity in the dataset (e.g., a person or a location), defined along an arbitrary number of **attributes** (e.g., age, position, time). Data elements are typically stored as rows in a data table, while attributes are stored as columns. Data elements need to be **ordered**. Their ordering can be either *a)* intrinsic (e.g., defined by the order of rows in a data table), *b)* based on values of an attribute (e.g., timestamp), or *c)* derived from a variety of attributes (e.g., minimization of pairwise distances between data elements [5]).

We define the curve encoding pipeline illustrated in Figure 1 as the following three-stage process:

1. In the **embedding** stage, each data element is mapped to a point with a position in space. Positions can be calculated through a variety of algorithms and visualization techniques, which will be discussed in the next section.
2. In the **connection** stage, the points are joined in order to produce a continuous curve. A curve segment is added between each pair of consecutive points, according to the ordering of the data elements. Although the curve segments can be straight, other interpolation techniques can be used (Bezier, splines, etc.). Taken together, we refer to the embedding + connection stages as **curve embedding**.
3. In the **enrichment** stage, data attributes are mapped to local visual attributes of the curve, such as thickness, texture, or color. Since there is a wide design space for enrichment, much of this article will focus on this stage.

We now describe the *embedding* stage in more detail. We will then turn to the *enrichment* stage.

EMBEDDING: Encoding Data with a Curve's Shape

Curve embedding provides rich opportunities for conveying data visually. A curve can take on many forms varying in complexity, from straight lines to smooth curves to zig-zag patterns. Global and local geometric features of a curve (e.g., sharp turns, loops, crossings) immediately stand out, and can tell captivating stories about the data [4, 15].

Existing curve embedding techniques can be categorized according to *i)* the dimensionality of the encoded data and *ii)* the dimensionality of the curve itself. **Data dimensionality** defines how many data attributes dictate the curve's

shape, i.e., how many of these attributes are used to position the points in the embedding stage of Figure 1. It can vary from zero to an arbitrary number of dimensions. In contrast, **curve dimensionality** refers to the dimensionality of the resulting curve. It is generally either 1D (i.e., a straight line), 2D (i.e., a planar curve), or 3D.

Here is an overview of curve embedding approaches classified according to their *data dimensionality*:

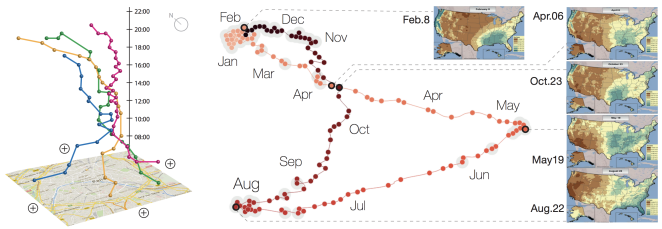
0D—In zero-dimensional approaches, the shape of the curve is not dictated by data but is decided in advance and fixed. Examples include *timelines* whose layout is data-independent and where data is encoded through the enriching process exclusively. In terms of curve dimensionality, while many timelines are 1D (i.e., straight lines [26, 8]), timelines can also take on complex shapes on the 2D plane (spirals, boustrophedons, etc. [8]), and even possibly in 3D space.

1D—Several curve embedding techniques exist that encode a single data attribute. Some *line charts* fall in this category, when the visualized dataset has no explicit time attribute (e.g., in time-series data where time intervals are constant, or in event sequences without time stamps). In such line charts, points are typically evenly spaced out on the horizontal axis, while their vertical position is decided by a single data attribute. Other examples include curves whose layout is fixed in advance but where points are positioned along the curve depending on some data attribute (e.g., time). Finally, a less common technique is the *turtlewalk*, where curves are constructed by adding each new data point at a fixed distance from the previous one, and at an angle that depends on the value of some data attribute. This technique has been used to visualize digits of π [31] but can also be used with continuous quantitative attributes.

2D—Many classical curve-based visualization techniques map two data attributes onto the two dimensions of the plane. These include all *line charts* that encode an explicit time attribute (typically on the x-axis), plus another data attribute (typically on the y-axis). A related technique is the *connected scatterplot*, where one data attribute is mapped to x while another one is mapped to y [15]. In contrast to the line chart, the data attribute mapped to the x-axis does not need to be monotonically increasing. While connected scatterplots can be used to encode abstract data, they can also be used to encode spatial data, in which case they simply become *2D trajectory visualizations*. While such Cartesian mappings are the most common, non-Cartesian mappings of two data attributes onto the 2D plane are also possible. For example, one could draw a line chart in polar coordinates, or use a modified turtlewalk technique where both point spacing and point angle encode data.

3D—Some curve-based visualization techniques use three data attributes in the embedding process. A common approach is the *space-time cube* curve, where two data attributes are mapped to x and y respectively, while a time attribute is mapped to z [2] (Figure 2(a)). This technique can be seen as an extension of *line charts* to two non-temporal attributes, and it is especially useful for visualizing two-dimensional trajectories [19]. *Connected scatterplots* and *trajectory visualizations* can also be generalized to three dimensions. All such visualizations use 3D embeddings, meaning that the produced curves are generally not planar. Such curves can be either be displayed without any loss of information through manual or digital fabrication[1], or they can be projected on a 2D medium [4].

[1] For two examples, see dataphys.org/list/wire-models-of-factory-worker-movements/ and dataphys.org/list/physical-space-time-cubes/



(a) (b)

Figure 2: Examples of higher-dimensional curves: a) 3D curve in a space-time cube (https://geotime.com) (© B.Bach), b) nD curve projected into 2D space: time curve [4] (© B.Bach).

nD—Through the use of *dimension reduction* techniques [14], it is possible to produce 2D or 3D curves based on an arbitrarily large number of data dimensions. Although dimension reduction techniques have been mostly used to produce point-based visualizations (e.g., scatterplots), the *time curve* approach [4, 30] (Figure 2(b)) illustrates how multidimensional scaling (MDS) can be used to produce curves that encode an arbitrarily large number of dimensions, while remaining relatively easy to interpret even for a non-technical audience. While such techniques can be powerful, they necessarily discard information in the data.

This overview helps classify curve-based visualizations according to the data and the encoding that are used to create the basic shape of the curve. In the next section, we present the last stage of the curve encoding pipeline shown in Figure 3: curve enrichment.

ENRICHMENT: Encoding Data with a Curve's Attributes

Enrichment is a stage where additional data attributes are mapped to visual attributes of the curve. In other words, it is

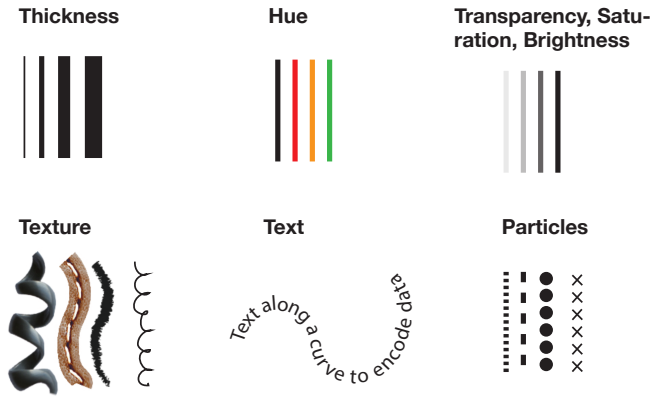


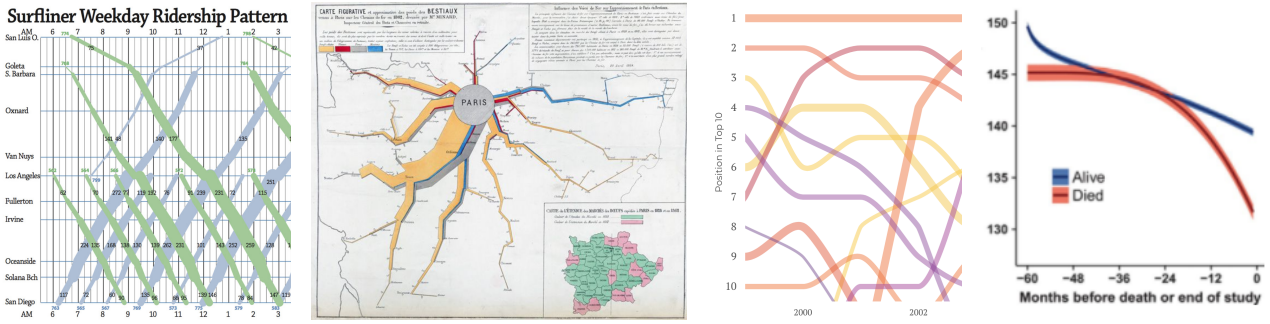
Figure 3: Visual variables used to encode data on the curve in the enrichment stage (© Q.Ren).

the stylizing process of the spatially-embedded curve.

In the following, we classify enrichment approaches by the visual variable employed (e.g., thickness, hue, etc). While Bertin [6] uses the same visual variables across all his visual marks (points, lines, areas), our classification is specific to lines. It includes both classical visual variables (covered first) and less conventional visual variables (covered later). To illustrate each visual variable, we give concrete examples taken from our survey and explain which data attributes are conveyed.

The visual variables covered in this section are illustrated in Figure 3.

Thickness refers to the width of the curve's stroke (Figure 3(a)). It is one of the most common visual variables used to enrich curves. For example, thickness is commonly used to represent the strength of traffic flows, such as the



(a) (b) (c) (d)

Figure 4: Examples for the use of thickness and color: a) strength of traffic (train passangers) [24] (© L.Nguyemn), b) transported units (animals) by Minard, (© C-J.Minard), c) Baby names (©N.Bremer), d) uncertainty in line charts [27] (© R.Ravindrarajah).

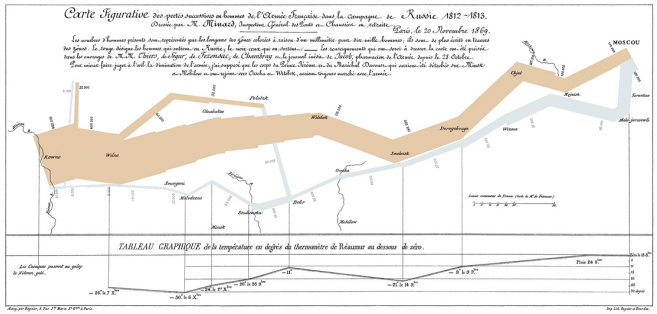
number of cars or people traveling through a road per unit of time [24, 11] (Figures 4(a) and 4(b)). Mapping traffic flow to curve thickness is an old practice that dates back from at least Charles Joseph Minard, with his *Napoleon's March* [23] (Figure 5(a)) and other flow maps.[2] GIS[3] maps line thickness to road throughput such that traffic jams become visible as narrow curve segments, while fluent traffic appears as thick segments. Finally, most road maps and street atlases use curve thickness to convey road importance.

Curve thickness has also been used to convey temporal information—for example, time and speed in trajectory visualizations [25], relative time intervals in time curves [4], and absolute time in connected scatterplot visualizations[4] (Figure 8(a)). Other usages include the communication of link strength in graphs [7], elevation in trail maps [34] (Figure 5(b)), variance in line charts [27] (Figure 4(d)), and highest position in ranking charts [9] (Figure 4(c)).

Hue is the 'type' of color that is used to color a curve and is often referred to simply as 'color'. This section focuses on hue only; we describe the other components of 'color' (transparency, saturation, and brightness) in the next subsection. Hue is commonly used to represent different objects, and to help visual grouping of curves of the same type. Examples include types of traffic flows (Figure 4(b)), type of roadss.

Other than for grouping, hue has been to encode data values at specific positions along the curve, for example, to show intensity of traffic flows and related speed of moving objects [11, 10]. An interesting use of hue is found in Minard's work *Napoleon's march* (Figure 5(a)) which uses hue to indicate both time of travel and direction; orange curve segments indicate the first part of Napoleaon's campaign with the army marching from west (left) to east (right). The army's retreat in the opposite direction is colored gray.

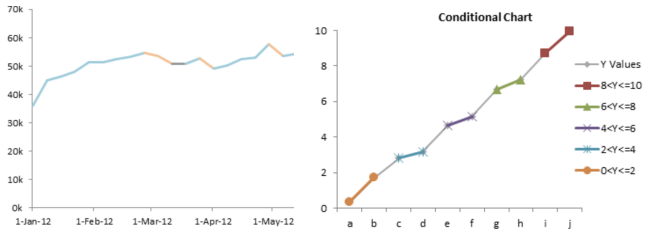
[2] https://sandrarendgen.wordpress.com/2013/06/22/the-forgotten-maps-of-minard/
[3] http://www.esri.com/news/arcwatch/0709/freeway-traffic.html
[4] http://truth-and-beauty.net/projects/remixing-rosling



(a)

(b)

Figure 5: Exampels of thickness encoding on maps: a) Napoleon's march (©C-J.Minard), b) Bike elevation map [33] (©J.Wood).



(a) (b)

Figure 6: Examples for the use of hue in linecharts (MS Excel) (© J.Peltier): a) Upwards, downwards, and stable periods, b) value intervals on y-axis ("conditional cart").

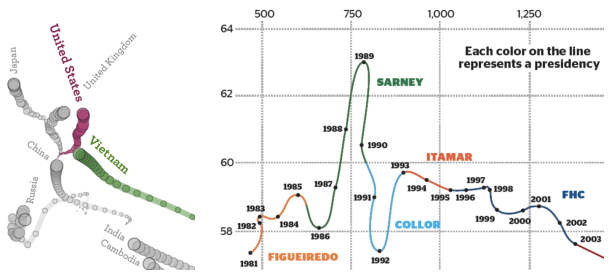
(a) (b)

Figure 7: Examples for the use of hue to show direction in 2D embeddings: a) Notabilia: Visualizing Deletion Discussions on Wikipedia. Each thread represents an article; green segments lean left and signify acceptance of a change, red segments lean right and mean rejection (© M.Stefaner), b) Exploring the Art hidden in Pi. Each digit (0-9) is encoded through both color and angle (direction). Visualizing the array of digits in the number Pi results in that fractal curve (© N.Bremer).

In line charts, hue has been used to encode upwards, downwards, and stable periods (Figure 6(a)) as well as value intervals (Figure 6(b)).[5] In these examples, hue is used redundantly to curve direction (up, down, unchanged) and position of the curve segment on the y-axis respectively. In other designs with a 2D embedding, hue has been used to indicate bivariate orientation: left/right[6] (Figure 7(a)) as well as angles (Figure 7(b)).[7]

Eventually, hue has been used with connected scatterplots (Figure 8) to differentiate between elements and their tra-

[5] https://peltiertech.com/conditional-formatting-of-lines-in-an-excel-line-chart-using-vba
[6] http://notabilia.net
[7] https://www.visualcinnamon.com/2015/01/exploring-art-hidden-in-pi.html



(a) (b)

Figure 8: Examples for encoding time and reading direction in connected scatterplots: a) Remixing Rosling (© M.Stefaner), b) Inequality and GDP (© Epoca Magazine, Brazil).

jectory in time (Figure 8(a)) and to highlight segments (periods) of time along a curve (Figure 8(b)).[8]

Transparency, **saturation**, and **brightness** mean the intensity of the curve and its transparency. As transparency and saturation are often used to achieve this same affect, here we review examples of both, though technically they remain two independent visual variables. Transparency is frequently used to discrimiate overlapping or clustered objects (e.g., in Figure 4(c)), yet less to directly encode data. One example that uses transparency to encode data is Robertson et al. [28] who use transparency to encode time in connected scatterplots, with more transparent point signaling points further in the past. This mimics the effect of fading out. A similar effect is achieve in time curves [4] (Figure 2(b)), using a mixture of brightness, saturation, and hue (a color range from lighter shinier to a darker orange) to indicate reading direction (start to end). This is a prominent example because it is one of the very few that encode the reading direction on a curve using a visual attribute rather than text labels such as years.

Texture refers to the local structure of curve. Other than Bertin, we differentiate between *particles* (e.g., dashed lines and different types of strokes) and *textures*. Particles are explained in the next subsection and refer to designs where the curve is represented through a set of individual visual elements. Textures, in our sense, are complex surface structures which would require shaders and/or photographs to be mapped onto the curve in both 2D and 3D visualizations of curves.

Bertin [6] offers many examples of textures for curves, e.g., to represent rivers, roads, and other curves on maps (Figure 9(a)). In these examples, the original curve shape is altered through specific renderings such as meanderings, small turns, double lines, pointlisim, etc. A similar example from Dear Data [21] (Figure 9(b)) shows example textures—zigzag lines of different frequence and almost curly lines—aside examples for particles (curves consisting of strokes and points) as explained in the next subsection.

A specific instance of texture is **text**, a technique inspired by calligrams. Text means words and text resembling apped onto the trajectory of the curve and its shape. Text curves are used to show spatial data where text alone forms the graphical features (Figure 9(c)) [1]. Every curve consisted of text representing the road's name. Beyond curves as regarded in this survey, words have been used in node-link diagrams to depict link types [32].

Particles can be seen as a specific type of texture, e.g., a dashed line. In this article, particles are *individual* visual marks, laid out along the curve's trajectory and resulting in a specific texture according to the Gestalt law of Continuation. A particle's shape can be anything as simple as a dot

[8] http://www.thefunctionalart.com/2012/09/in-praise-of-connected-scatosingter-plots.html

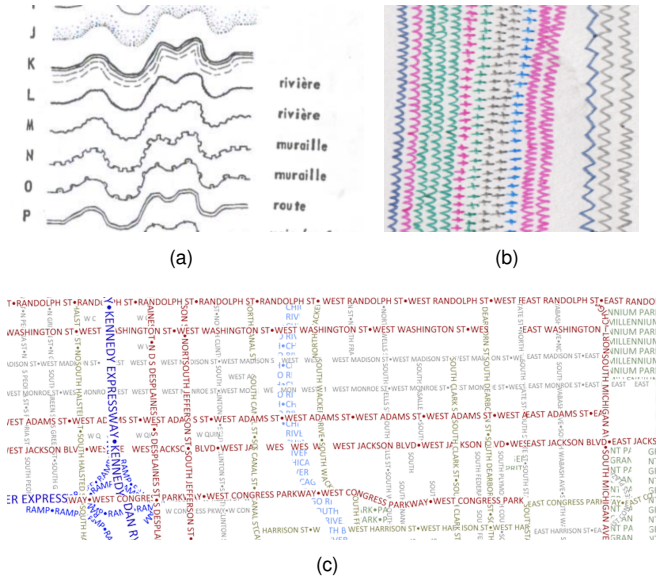


Figure 9: Examples of texture on curves: a) curves on maps showing rivers, river banks, streets, walls (© J.Bertin), b) particles and zig-zag curves in Dear Data (©S.Posavec [21]), c) text depicting roads [1] (© S.Afzal).

(resulting in a dotted line), a stroke segment (resulting in a dashed line), strokes perpendicular to the direction of the curve, symbols (small crosses in Fig 9(b)) and complexer visual elements such as 3D [13] (Figure 10).

Particles can be applied independently from any encoding of an underlying curve, such as using tickmarks on top of an existing trajectory. For example, Figure 11(a) uses tick marks along trajectories to indicate the number of travel days: the space between two tickmarks shows the space traveled in one day [6]. Bertin provides many examples for trajectories on maps where different particle shapes, alter-

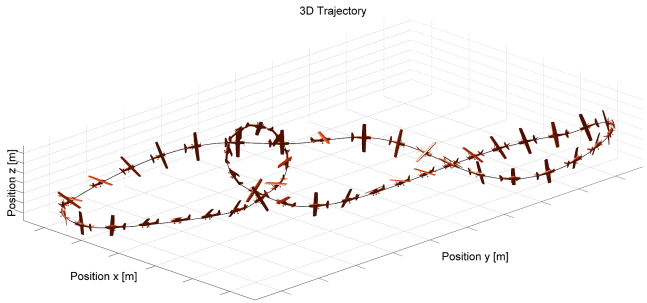


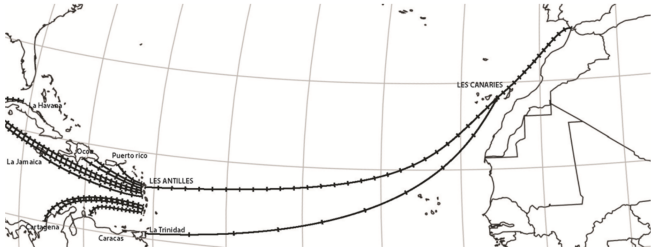
Figure 10: Visualization of a flight-route, including the inclination and facing-direction of an airplane ([13], © F.Fisch).

nations, and spacing along the curve, differentiate trajectories (Figure 11(b)). Shapes in this example include arrowheads, arrows with tail, and alternations between points and arrows.

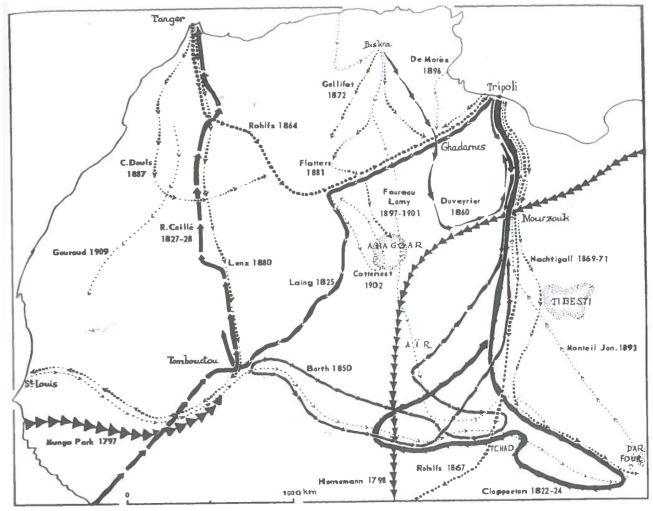
Particles have been eventually studied in more detail by Romat et al. [29] for conveying different information on links in networks. Alongside the common visual attributes (size, color, and shape) their framework defines parameters such as *particle pattern* (a pattern of particles similar to a Morse code), *pattern frequency* (visual spacing of a pattern), and *particle speed*. Each of these parameters can be used to encode different data [9].

Finally, some animations[10] use particles to show data elements traveling along the trajectory. Number, density and position of the particles represent information in the data. Other examples (e.g. [29]) use parameters of a particle system to encode data, and animating particles can be used to indicate direction of flow along a curve [3, 29].

[9] http://ilda.saclay.inria.fr/flownet
[10] https://www.youtube.com/watch?v=v1kfW1d3zsA



(a)



(b)

Figure 11: Examples for (static) particles along trajectories [6] (© J.Bertin): a) marks indicating days of travel (one mark=1 day), b) particle types indicate different expeditions.

Conclusion

We have started describing the richness of curve designs in information visualization by introducing the combination of curve embedding and curve enrichment to describe existing curve designs. While our framework is preliminary, it helps in organizing and comparing existing curve designs used in information visualization. We anticipate that it can inform the design of novel visualizations that rely on curves to encode information. For example, all of our curve-specific visual variables can be combined and hence be used to independently encode a set of data attributes for each segment of the curve. Eventually, such a systematic review can inform the evaluation of individual or combinations of curve designs for specific applications, tasks, and datasets.

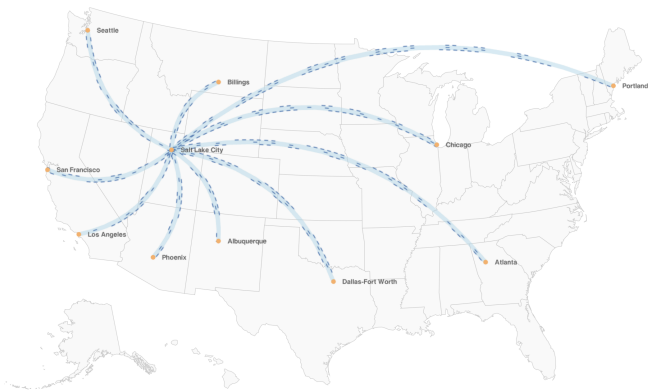


Figure 12: Example of the use of (moving) particles to repict direction, load, and delay of flights [29](© R.Hugo).

REFERENCES

1. Shehzad Afzal, Ross Maciejewski, Yun Jang, Niklas Elmqvist, and David S Ebert. 2012. Spatial text visualization using automatic typographic maps. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 18, 12 (2012), 2556–2564.

2. Benjamin Bach, Pierre Dragicevic, Daniel Archambault, Christophe Hurter, and Sheelagh Carpendale. 2017a. A Descriptive Framework for Temporal Data Visualizations Based on Generalized Space-Time Cubes. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 36–61.

3. Benjamin Bach, Nathalie Henry Riche, Christophe Hurter, Kim Marriott, and Tim Dwyer. 2017b. Towards unambiguous edge bundling: Investigating confluent drawings for network visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 23, 1 (2017), 541–550.

4. Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, and Pierre Dragicevic. 2016. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 22, 1 (2016), 559–568.

5. Michael Behrisch, Benjamin Bach, Nathalie Henry Riche, Tobias Schreck, and Jean-Daniel Fekete. 2016. Matrix reordering methods for table and network visualization. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 693–716.

6. Jacques Bertin. 1973. Sémiologie graphique. *Flammarion, Paris* (1973).

7. U. Brandes and B. Nick. 2011. Asymmetric Relations in Longitudinal Social Networks. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 17, 12 (Dec 2011), 2283–2290.

8. Matthew Brehmer, Bongshin Lee, Benjamin Bach, Nathalie Henry Riche, and Tamara Munzner. 2017. Timelines revisited: A design space and considerations for expressive storytelling. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 23, 9 (2017), 2151–2164.

9. Nadieh Bremer. 2015. The 10 most popular baby names per year since 1880. `http://nbremer.github.io/babynames/`. (2015). [online, last accessed Jan 31, 2018].

10. Stefan Buschmann, Matthias Trapp, and Jürgen Döllner. 2014. Real-time animated visualization of massive air-traffic trajectories. In *Proc. of IEEE Cyberworlds (CW)*. 174–181.

11. Grace C. Chung. 2009. Visualizing Freeway Traffic in the San Diego Region with GIS. online: `http://www.esri.com/news/arcwatch/0709/freeway-traffic.html`. (2009). [last accessed: Jan 31, 2018].

12. Gareth Cook and Robert Krulwich. 2016. *The Best American Infographics 2016*. Houghton Mifflin Harcourt.

13. Florian Fisch. 2011. *Development of a framework for the solution of high-fidelity trajectory optimization problems and bilevel optimal control problems*. Ph.D. Dissertation. Technische Universität München.

14. Imola K Fodor. 2002. *A survey of dimension reduction techniques*. Technical Report. Lawrence Livermore National Lab., CA (US).

15. Steve Haroz, Robert Kosara, and Steven L Franconeri. 2016. The connected scatterplot for presenting paired time series. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 22, 9 (2016), 2174–2186.

16. Danny Holten, Petra Isenberg, Jarke J Van Wijk, and Jean-Daniel Fekete. 2011. An extended evaluation of the readability of tapered, animated, and textured

directed-edge representations in node-link graphs. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE*. IEEE, 195–202.

17. Wassily Kandinsky. 1927. *Punkt und Linie zu Fläche: Beitrag zur Analyse der malerischen Elemente*. Vol. 9.
18. Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus Gross, and Hanspeter Pfister. 2017. Visualizing Nonlinear Narratives with Story Curves. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* (2017).
19. Menno-Jan Kraak. 2003. The space-time cube revisited from a geovisualization perspective. In *Proc. 21st International Cartographic Conference*. 1988–1996.
20. Manual Lima and others. 2011. Visual complexity. *source: http://www. visualcomplexity. com* (2011).
21. G Lupi and S Posavec. 2016. Dear data: The story of a friendship in fifty-two postcards. (2016).
22. N. McCurdy, J. Lein, K. Coles, and M. Meyer. 2016. Poemage: Visualizing the Sonic Topology of a Poem. *IEEE Transactions on Visualization and Computer Graphics (TVCG))* 22, 1 (Jan 2016), 439–448.
23. Charles-Joseph Minard. 1869. Carte figurative des pertes successives en hommes de l'armée francaise dans la Campagne de Russie 1812-13 (comparées à celle d'Hannibal durant la 2ème Guerre Punique). Bibliothèque nationale de France. Available at `http://gallica.bnf.fr/ark:/12148/btv1b52504201x`.
24. Lam Nguyen. 2009. Surfliner Weekday Ridership Pattern. Memorandum to Chairs and Commissioners regarding Follow up on September and December 2008 Rail Items. State of California Department of Transportation. (2009).
25. Charles Perin, Tiffany Wun, Richard Pusch, and Sheelagh Carpendale. 2017. Assessing the Graphical Perception of Time and Speed on 2D+ Time Trajectories. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* (2017).
26. Catherine Plaisant, Brett Milash, Anne Rose, Seth Widoff, and Ben Shneiderman. 1996. LifeLines: visualizing personal histories. In *Proceedings of the SIGCHI conference on Human factors in computing systems(CHI)*. ACM, 221–227.
27. Rathi Ravindrarajah, Nisha C Hazra, Shota Hamada, Judith Charlton, Stephen HD Jackson, Alex Dregan, and Martin C Gulliford. 2017. Systolic Blood Pressure Trajectory, Frailty, and All-Cause Mortality> 80 Years of AgeClinical Perspective: Cohort Study Using Electronic Health Records. *Circulation* 135, 24 (2017), 2357–2368.
28. George Robertson, Roland Fernandez, Danyel Fisher, Bongshin Lee, and John Stasko. 2008. Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 14, 6 (2008).
29. Hugo Romat, Caroline Appert, Benjamin Bach, Nathalie Henry-Riche, and Emmanuel Pietriga. 2018. Animated Edge Textures in Node-Link Diagrams: a Design Space and Initial Evaluation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. ACM.
30. Stef van den Elzen, Danny Holten, Jorik Blaas, and Jarke J van Wijk. 2016. Reducing snapshots to points: A visual analytics approach to dynamic network exploration. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 22, 1 (2016), 1–10.

31. John Venn. 2006. *The logic of chance*. Courier Corporation.

32. Pak Chung Wong, Patrick Mackey, Ken Perrine, James Eagan, Harlan Foote, and Jim Thomas. 2005. Dynamic visualization of graphs with extended labels. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*. IEEE, 73–80.

33. J. Wood. 2015a. Visualizing Personal Progress in Participatory Sports Cycling Events. *IEEE Computer Graphics and Applications* 35, 4 (July 2015), 73–81.

34. Jo Wood. 2015b. Visualizing personal progress in participatory sports cycling events. *IEEE Computer Graphics and Applications* 35, 4 (2015), 73–81.

35. Jo Wood, Aidan Slingsby, and Jason Dykes. 2011. Visualizing the Dynamics of London's Bicycle-Hire Scheme. *Cartographica: The International Journal for Geographic Information and Geovisualization* 46, 4 (2011), 239–251.