Topological Equivariant Artist Model for Visualization Library Architecture

Hannah Aizenman, Thomas Caswell, and Michael Grossberg, *Member, IEEE*,

Abstract—The abstract goes here.

Index Terms—

1 Introduction

This paper uses methods from topology and category theory to develop a model of the transformation from data to graphical representation. This model provides a language to specify how data is structured and how this structure is carried through in the visualization, and serves as the basis for a functional approach to implementing visualization library components. Topology allows us to describe the structure of the data and graphics in a generalizable, scalable, and trackable way. Category theory provides a framework for separating the transformations implemented by visualization libraries from the various stages of visualization and therefore can be used to describe the constraints imposed on the library components [1], [2]. Well constrained modular components are inherently functional [3], and a functional framework yields a library implementation that is likely to be shorter, clearer, and more suitd to distributed, concurrent, and on demand tasks [4]. Using this functional approach, this paper contributes a practical framework for decoupling data processing from visualization generation in a way that allows for modular visualization components that are applicable to a variety of data sets in different formats. is it OK that this is something reviewer 4 wrote

2 Related Work

This work aims to develop a model for describing visualization transformations that can serve as guidance for how to architecture a general purpose visualization library. We define a general purpose visualization library as one that provides non domain specific building block components [5] for building visualizations, for example functions for converting data to color or encoding data as dots. In this section, we describe how visualization libraries attempt this goal and discuss work that formally describes what properties of data should be preserved in a visualization.

- *H. Aizenman and M. Grossberg are with the department of Computer Science, City College of New York.*
  E-mail: haizenman@ccny.cuny.edu, mgrossberg@ccny.cuny.edu
- *Thomas Caswell is with National Synchrotron Light Source II, Brookhaven National Lab*
  E-mail: tcaswell@bnl.gov

Manuscript received X XX, XXXX; revised X XX, XXXX.

We restrict the properties of data that should be preserved to

continuity how elements in a dataset are connect to each other, e.g. discrete rows in a table, networked nodes, pixels in an image, points on a line

equivariance functions on data that have an equivalent effect on the graphical representation, e.g. rotating a matrix has a matching rotation of the image, translating the points on a line has a matching visual shift in the line plot

2.1 Continuity

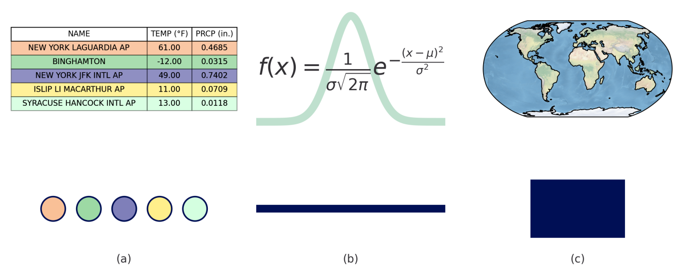


Fig. 1: Continuity is how elements in a data set are connected to each other, which is distinct from how the data is structured. The rows in (a) are discrete, therefore they have discrete continuity as illustrated by the discrete dots. The gaussian in (b) is a 1D continuous function, therefore the continuity of the elements of the gaussian can be represented as a line on an interval (0,1). In (c), every element of the globe is connected to its nearest neighbors, which yields a 2D continuous continuity as illustrated by the square.

Continuity is a representation of how the elements in a dataset are connected to each other. For example, in Figure 1, each station record in the table is independent of the others; therefore, the continuity of the table is discrete. The data provided by the gaussian are points sampled along the curve, therefore the continuity of the points on the line is 1D continuous. Every point on the globe is connected to its 6 nearest cardinal neighboring points (NW, N, NE, E, SE, S, SW, W).

Often continuity is expressed in the choice of visual algorithm (visualization type), as explored in taxonomies

Fig. 2: Continuity is implicit in choice of visualization rather than explicitly in choice of data container. The line plots in (b) are generated by a 2D table (a). Structurally this table can be identical to the 2D matrix (a) that generates the image in (c).

by Tory and Möller [6] and Chi [7]. For example, in Figure 2 the same table can be interpreted as a set of 1D continuous curves when visualized as a collection of line plots or as a 2D surface when visualized as an image. This means that often there is no way to express data continuity independent of visualization type, meaning most visualization libraries will allow, for example, visualizing discrete data as a line plot or an image. General purpose visualization libraries-such as Matplotlib [8], Vtk [9], [10], and D3 [11]-carry distinct data models as part of the implementation of each visual algorithm. The lack of unified data model means that each plot in a linked [12], [13] visualization is treated as independent, as are the transforms converting each field in the data to a visual equivalent.

Domain specific libraries can guarantee consistency because they have a single model of the data in their software design, as discussed in Heer and Agarwal [14]'s survey of visualization software design patterns. For example, the relational database is core to tools influenced by APT, such as Tableau [15], [16], [17] and the Grammar of Graphics [18] inspired ggplot [19], Vega [20] and Altair [21]. Images underpin scientific visualization tools such as Napari [22] and ImageJ [23] and the digital humanities oriented ImagePlot [24] macro; the need to visualize and manipulate graphs has spawned tools like Gephi [25], Graphviz [26], and Networkx [27].

2.1.1 Fiber Bundles

A model that allows for expressing data continuity in a general flexible way allows us to express, and therefore be faithful to, the multivariate consistency constraints described by Qu and Hullman [28]. A general data model also allows for consistent adaptation to modern data needs, such as complex, metadata rich, distributed, or streaming data. The mathematical theory of fiber bundles provides one such abstraction that can express complicated dimensionality and continuity without being tied to any one data container

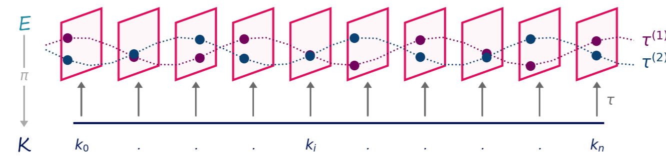


Fig. 3: A fiber bundle is mathematical construct that allows us to express the relationship between data and continuity. The total space Eis the topological space in which the data is embedded. The fiber space F is embedded in E and is the set of all possible values that any add big rectangle E

type, as proposed by Butler, Bryson, and Pendley [29], [30]. In this paper, we build on their work that proposes using topological spaces to represent different properties of data. Here we present a brief summary of topology, for more information see Hatcher **[?]**, Munkres **[?]**, and Bradley et. al. **[?]**.

A topological space a topological space (X, T) is a set X with a topology T. Topologies are collections of open sets

such that the empty set and X are in the collection of open sets T, the union of elements in T

Specifically, Butler, Bryson, and Pendley suggest that fiber bundles be the basis of an abstract data model. Fiber bundles are a collection (E, K, F, π) of topological spaces

Total Space E
Fiber Space F
Base Space K

with a projection map $\pi : E \to K$ that connects every point in E to a point in K.

$$F \hookrightarrow E \xrightarrow{\pi} K \tag{1}$$

As indicated by $\hookrightarrow$, the fiber space F is embedded inside the total space EThis is illustrated in Figure 3, wherein the data lives in the manifold E. Each data point has two dimensions, represented as the square fiber F. While the data is embedded in E, it's continuity is that each point lies along a line, represented by the base space K.

2.2 Sheaf Maps $\mathcal{O}$

We can use sheafs to ensure continuity even when the data is broken up. Is the glue rules -¿ we can haz parallism.

$$A : \mathcal{O}(E) \to \mathcal{O}(H) \tag{2}$$

2.3 Equivariance

What is it?

$$\begin{tikzcd}
\text{data} \arrow[r] \arrow[d, "\text{function}"] & \text{representation} \arrow[r] & \text{visual stimulus} \arrow[d, "\text{visual equivalent to fu}"] \\
\text{data} \arrow[r] & \text{representation} \arrow[r] & \text{visual stimulus}
\end{tikzcd}$$

2.3.1 Category Theory

In this work, we propose that equivariance constraints can be expressed using category theory. Vickers et. al provide a brief introduction to category theory for visualization practitioners [31], but their work focuses on data, representation, and evocation, while this paper is aims to provide guidance on how the map from data to representation should be implemented.

Fig. 4

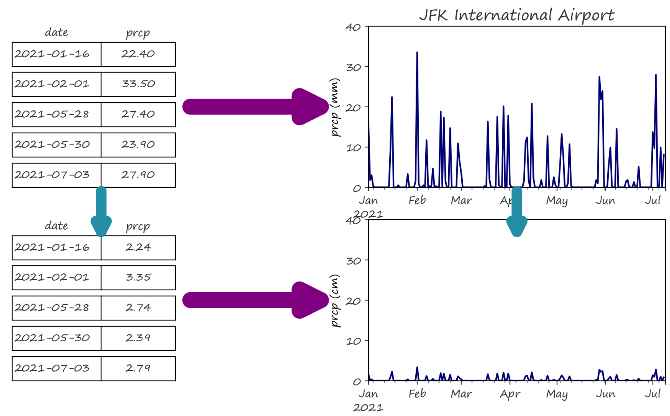


Fig. 5: Equivariance is that a transformation on the data has a corresponding transformation in the graphical representation. For example, in this figure the data is scaled by a factor 10. Equivalently the line plot is scaled by factor of 10, resulting in a shrunken line plot. Either a transformation on the data side can induce a transformation on the visual side, or a transformation on the visual side indicates that there is also a transformation on the data side.

3 Artist

The Artist $\mathcal{A}$ is a transformation from a

$$\mathcal{A} : E \to H \tag{3}$$

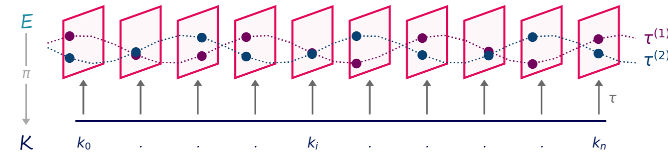
3.1 Data Bundle E

We use topology to model

The The continuity of the data is encoded in the base space K.

The properties of the variables are encoded in the fiber space F. The fiber is a topological space

Spivak provides notation for describing the set of all possible values U_σ of a single column with name c and type T. This set of values is the fiber "F" $F = U_\sigma(c)$. When data is multivariate, the fiber F is the cartesian product of each columns fiber space (5)



replace with more concrete
replace with more concrete

Fig. 6:
replace with more concrete

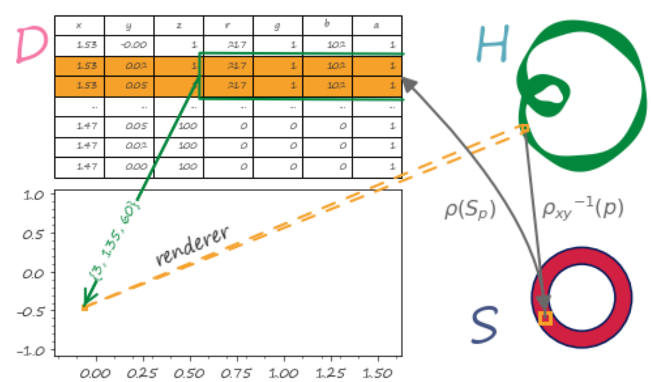
3.1.1 Structured Data: Section τ

$$\begin{tikzcd} F \arrow[r, hook] & E \arrow[d, shift right, "\pi"'] \\ & K \arrow[u, bend right, "\tau"'] \end{tikzcd} \tag{4}$$

3.1.2 Structure: Continuity and Equivariant Actions

$$\begin{tikzcd} F_i \arrow[d, "m_j"'] \arrow[dr, "m_j \circ m_k"] & \\ F_i \arrow[r, "m_k"'] & F_i \end{tikzcd} \tag{5}$$

3.2 Graphic Bundle H



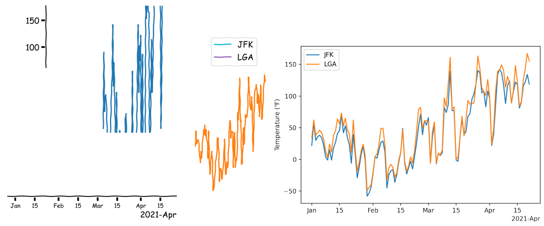
$$\begin{tikzcd} D \arrow[r, hook] & H \arrow[d, shift right, "\pi"'] \\ & S \arrow[u, bend right, "\rho"'] \end{tikzcd} \tag{6}$$

3.2.0.1 Continuity: Base Space S:
3.2.0.2 Equivariance: Fiber D:
3.2.0.3 Structured Data: Visual ρ:

3.3 Union of Artists

4 Constraints: Construction and formal properties of Artists

Identify the constraints, category theory makes it easier to spell out these constraints, this map has to be equivariant ordered data data on trees, there is a monoid that acts on them is not clear that if your map acts on them there will be a nice map on ordered why category theory? - F-¿V-¿D -¿ F-¿V is 1:1 equivariance -¿ V-¿D is glyph equivariance



(a) Union of H (b) Artist with shared/harmonized μ

Fig. 7: Simulation results for the network.

What are the formal properties and constraints on xi, nu, and qhat

nu = any function f:E$\rightarrow$ E, $equivV \rightarrow Vf : F_i \rightarrow F_i, \nu_i : F_i P_i,$

define category of structure preserving transformations on F - valid maps F-¿F - user in setting up fibers, sets up what are valid transforms/morphisms - nu is a functor

$$\begin{tikzcd} E \arrow[r, "\nu"] \arrow[dr, "\pi"'] & V \arrow[d, "\pi"] & \xi^*V \arrow[l, "\xi^*"'] \arrow[r, "Q"] \arrow[d, "\xi^*\pi"'] & H \arrow[dl, "\pi"] \\ & K & S \arrow[l, "\xi"] & \end{tikzcd} \tag{7}$$

4.1 Graphic to Data: ξ

$$\begin{tikzcd} E \arrow[d, "\pi"'] & H \arrow[d, "\pi"'] \\ K & S \arrow[l, "\xi"'] \end{tikzcd} \tag{8}$$

4.2 Visual Bundle V

$$\begin{tikzcd} P \arrow[r, hook] & V \arrow[d, "\pi"'] \\ & K \arrow[u, bend right, "\mu"'] \end{tikzcd} \tag{9}$$

4.3 Data to Visual Encodings: ν

Visual Channel Encodings

$$\{\nu_0, \ldots, \nu_n\} : \{\tau_0, \ldots, \tau_n\} \mapsto \{\mu_0, \ldots, \mu_n\} \tag{10}$$

We enforce the equivariance constraint

$$\begin{tikzcd} E_i \arrow[r, "\nu_i"] \arrow[d, "m_r"'] & V_i \arrow[d, "m_\nu"] \\ E_i \arrow[r, "\nu_i"] & V_i \end{tikzcd} \tag{11}$$

4.3.1 *Visual to Graphic:* Q

Visual encodings to something like marks

5 Case Study

We implement the **arrows** in Figure 10. `axesArtist` is a parent artist that acts as a screen. This allows for the composition described in subsection 3.3

5.1 A

```
for local_tau in axesArtist.artist.data.query(screen_bounds, dpi):
    mu = axesArtist.artist.graphic.mu(local_tau)
    rho = axesArtist.artist.graphic.qhat(**mu)
    H = rho(renderer)
```

where the artist is already parameterized with the ξ functions and which fibers they are associated to:

5.1.1 ξ

5.1.2 ν

5.1.3 $\hat{Q}$

6 Discussion

6.1 Limitations

6.2 future work

7 Conclusion

The conclusion goes here.

Appendix A
Rendering: ρ

Appendix B
Manufacturing $\hat{Q} \leftarrow Q$

$$\begin{tikzcd} E \arrow[r, "\nu"] \arrow[dr, "\pi"'] & V \arrow[d, "\pi"'] & \xi^*V \arrow[l, "\xi^*"'] \arrow[r, "Q"] \arrow[d, "\xi^*\pi"'] & H \arrow[dl, "\pi"] \\ & K \arrow[u, bend right, "\mu"'] & S \arrow[l, "\xi"] \arrow[u, bend right, "\xi^*\mu"'] & \end{tikzcd} \tag{12}$$

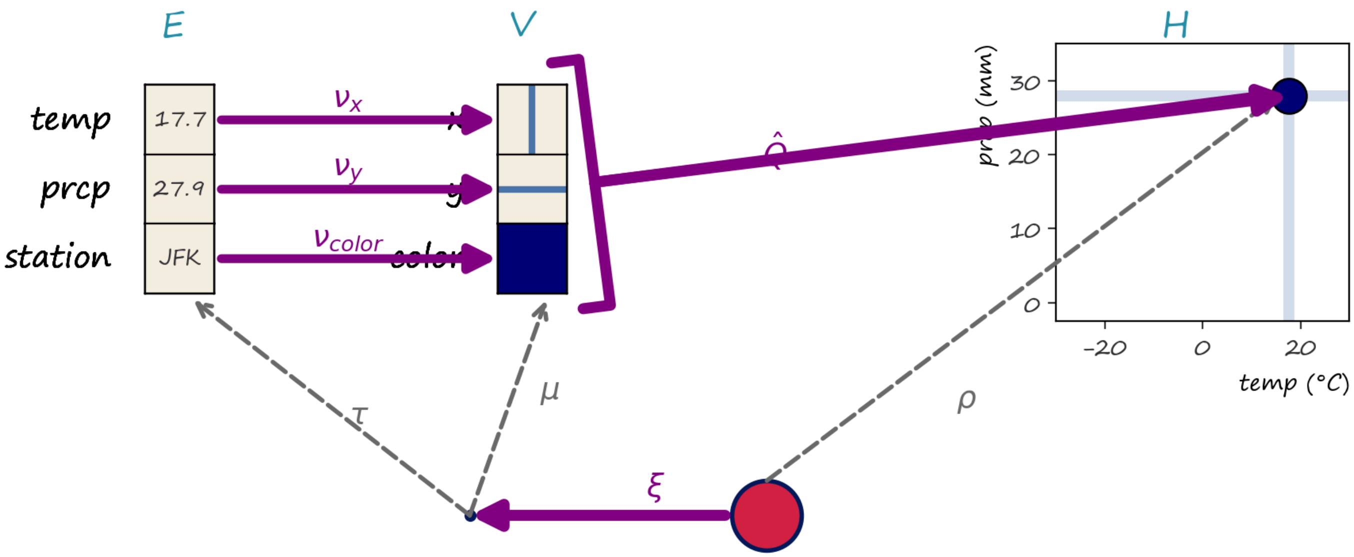
Acknowledgments

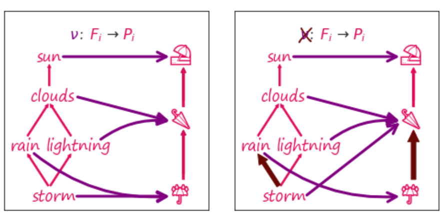
The authors would like to thank...

Hannah Aizenman Biography text here.

Thomas Caswell Biography text here.

Michael Grossberg Biography text here.





(a) Artists with shared μ_i renderered correctly

(b) Artists without shared μ_i

Fig. 8: Simulation results for the network.

Fig. 9: rework this as a commutative box w/ the r in E row associated w/ this qhat(k)

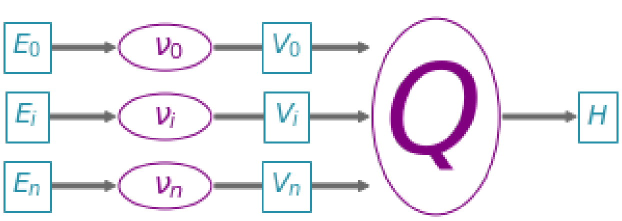


Fig. 10: add in xi!