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 [?], [?]. Well constrained modular components are inherently functional [?], and a functional framework yields a library implementation that is likely to be shorter, clearer, and more suited to distributed, concurrent, and on demand tasks [?]. 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

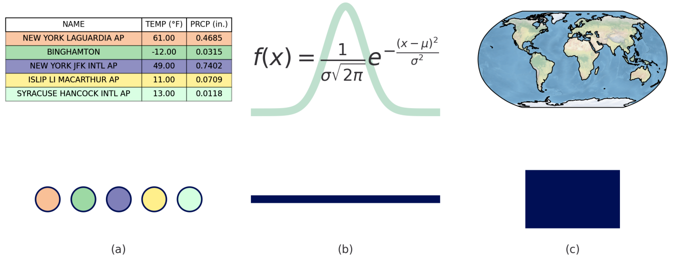
2 Related Work

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 [?] 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. We restrict the properties of data that should be preserved to

continuity how elements in a dataset are organized, 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



NAME	TEMP (°F)	PRCP (in.)
NEW YORK LAGUARDIA AP	61.00	0.4685
BINGHAMTON	-12.00	0.0315
NEW YORK JFK INTL AP	49.00	0.7402
ISLIP LI MACARTHUR AP	11.00	0.0709
SYRACUSE HANCOCK INTL AP	13.00	0.0118

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{\sigma^2}}$$

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 describes elements in a data set are organized; this concept is termed topological properties by Wilkinson [?]. Wilkinson provides the examples of values that are isolated from each other, and therefore discrete, and values lying on a continuum or in a compact region. For example, in **??**, 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). We propose that a robust model of continuity provides a way to develop library components that can

---

- *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.

work on the data in small pieces in a manner where the overall continuity of the data is preserved in the visual transformation.

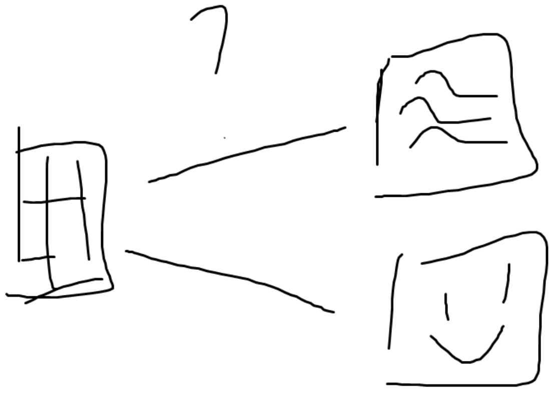


Fig. 2: Continuity is implicit in choice of visualization rather than explicitly in choice of data container. The line plots in (b) are generated by a 2D table (a). Structurally this table can be identical to the 2D matrix (a) that generates the image in (c).

The preservation of continuity can be made explicit, as in the transformation of table to parallel coordinates in Ruchikachorn and Mueller [?], but is often expressed implicitly in the choice of visual algorithm (visualization type), as explored in taxonomies by Tory and Möller [?] and Chi [?].

For example, in **??** 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 [?], Vtk [?], [?], and D3 [?]-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 [?], [?] visualization is treated as independent, as are the transforms converting each field in the data to a visual equivalent.

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

2.1.1 *Fiber Bundles*

The model described in this work provides a modol for expressing data sets with different topological properties.

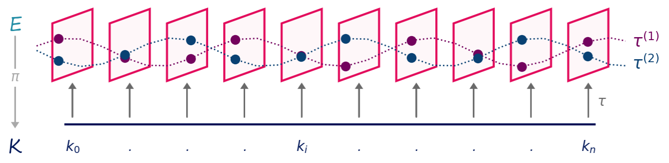


Fig. 3: A fiber bundle is mathematical construct that allows us to express the relationship between data and continuity. The total space Eis the topological space in which the data is embedded. The fiber space F is embedded in E and is the set of all possible values that any add big rectangle E

We obtain this generality by using the mathematical theory of fiber bundles as the basis of our abstraction, as proposed by Butler, Bryson, and Pendley [?], [?]. In this paper, we build on their work that proposes using topological spaces to represent different properties of data.

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

Definition 2.1. A topological space $(X, \mathscr{T})$ is a set X with a topology $\mathscr{T}$. Topologies are collections of open sets U such that the empty set and X are in the collection of open sets $\mathscr{T}$, the union of elements in $\mathscr{T}$ finish out this definition

Here we present a very brief summary of topics, for more information Hatcher [?], Munkres [?], and Spanier [?].

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 **??**, wherein values lives in the fiber $F \subseteq E$. In this example, the fiber F is the cartesian product of two sets $F_0 \times F_1$ where each fiber F_i is ....

$$formalequationofthefiber? \tag{2}$$

While the values are embedded in F, the continuity of the data is modeled as the base space K, which is the quotient space [?]

$$formalmathofthebasespaceasequivalenceclassforF_[k] \tag{3}$$

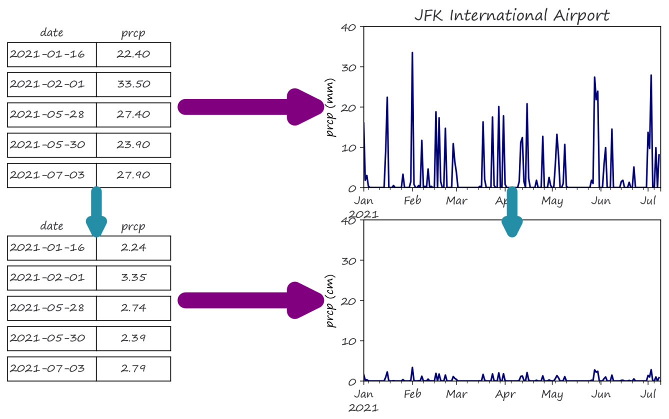
which means that every point in F_k maps to a point $k \subset K$.

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

Throughout this paper, we denote fiber bundles with the tuple $(totalspace, basespace, \pi, fiberspace)$

2.2 Equivariance

When introducing the retinal variables, Bertin informally specifies that continuity is preserved in the mark and defines equivariance constraints in terms of data and visual variables being selective, associative, ordered, or quantitative [?]. In the *A Presentation Tool*(APT) model, Mackinlay embeds the continuity constraint in the choice of visualization type and generalizes the equivariance constraint to preserving a binary operator from one domain to another. The algebraic model of visualization [?], proposed by Kindlmann and Scheidegger, restricts equivariance to invertible transformations.



date	prcp
2021-01-16	22.40
2021-02-01	33.50
2021-05-28	27.40
2021-05-30	23.90
2021-07-03	27.90

date	prcp
2021-01-16	2.24
2021-02-01	3.35
2021-05-28	2.74
2021-05-30	2.39
2021-07-03	2.79

Fig. 4: 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.

2.2.1 *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 [?], 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.

For more information on category theory, see Barr and Wells [?], Fong and Spivak [?], Riehl [?] and Bradley et. al. [?].

3 Artist

In this section, we use category theory to formally express the implicit assumptions that visualization library components make in transforming data into graphical representations. We propose that the visualization transformation can be modeled as a functor, which we call the Artist $\mathcal{A}$[1]. The artist Aconverts data, which we model as the catgory $\mathcal{E}$ to graphics, which we model as the category $\mathcal{H}$.

$$A : \mathcal{E} \to \mathcal{H} \tag{5}$$

We define the categories $\mathcal{E}, \mathcal{H}$ as sheaves of sections [?], [?], of fiberbundles, as introduced in **??**. We assume that $\mathcal{E}$ is defined in terms of a fixed base and fiber space, for example a class of images or tables with the same schema. The graphic category $\mathcal{H}$ consists of the set of graphics that an artist $A$is allowed to generate from a given $\mathcal{E}$, which is discussed further in **??**.

[1] We call this transformation the artist because the `Artist` object in Matplotlib [?]

As introduced in **??**, sets of open sets form the basis of topological spaces. We propose that we can express how the elements in our data are connected to each other by modeling the data continuity as a topological space K. We express subsets of the continuity, which can be mapped into subsets of the data, as open sets U_i in K. We introduce the category $\mathcal{K}$ to encapsulate the subsets of the continuity and the inclusion maps that glue them together.

Definition 3.1. The category of open sets $\mathcal{K}$ consists of

- *objects* all open sets $U_i \in \{U\}$ in the topological space $(K, \mathscr{T})$), including the empty set $\varnothing$ and the maximum set K.
- *morphisms* inclusion maps from a subspace to a larger space $\iota : U_i \hookrightarrow U_j$, which is equivalent to $U_i \subseteq U_j$ and

The inclusion maps $iota$ can be used to piece together the subsets U_i, particularly individual points $k \in K$. This is how we provide knowledge of continuity to the artist A in a way where the artist does not need all the data at once. This is what allows the artist to generate graphics where the subset of data on view is dynamically updated, such as pan and zoom navigation [?] and sliding windows on streaming data [?], [?].

Besides expressing continuities, fiber bundles also provide us a way to richly express the structure of the fields in the data. Specifically, we do this by adopting Spivak's formulation of the fiber as a (column name, data domain) simple schema [?], [?]. Spivak formally maps column names and field types to the set of values associated with the field type, for example $\mathbb{R}$ for a `float` column named *temperature*. We define F as the cartesian cross product of the fibers F_c for each field, where c is the field name.

Definition 3.2. The category $\mathcal{F}$ consists of a single object F and morphisms between the object and itself $M : F \to F$.

The object F can be an object of any category $\mathcal{C}$, allowing F to encode most data fields. For example, a lists of strings is an instance of an object in **Set**, networks are an instance of **Graph**, and images are vector spaces which are a specific type of topological space **Top**. We note that a one object category is a monoid [?], [?]. ? why is this important The morphisms M are functions from the data to itself, for example the binary operations, group actions, and measurement scales discussed in **??**. These functions could also be monoid actions [?], which provide a way of applying partial order relations to data [?], such as to build multi-ranked indicators [?]

We encode individual datasets in the fiberbundle as sections, and we follow the convention that $\Gamma(U_i, E)$ denotes the set of sections $\{\tau^j : U_i \to F_{U_i}\}_{j \in I}$ of the fiber bundle E over an open set $_i$. As mentioned in **??**, the fiber space is embedded in the total space $F \hookrightarrow E$. The sections $\Gamma(K, E)$ are called the global sections of E [?], [?]. Each section is how we represent a unique dataset, where the set of sections Γ

is the set of all datasets that have the same data fields in the F and continuity as encoded in$K$For example, Spivak describes tables that have the same schema and discrete index as sections of the same fiber bundle [?].

Definition 3.3. The category $\mathcal{E}$ consists of

- *object* the contravariant functor $\mathcal{O} : \mathcal{K}^{op} \to \Gamma(\mathcal{K}, E)$
- *morphisms* the endomorphic [?] bundle maps $\bullet : E \to E$, which are the fiber maps $M : \mathcal{F} \to \mathcal{F}$ when the sheaf is over a single point in the base space $k \in K$

The functor object of $\mathcal{E}$ is a presheaf because the definition of a presheaf is that it is a contravariant functor from a category $\mathcal{K}$ to a set [?], [?], [?]. As illustrated in the diagram,

$$\begin{tikzcd}
U_1 & \Gamma(U_1, E) \\
U_2 \arrow[from=1-1, d, "\iota"'] \arrow[r, phantom, ""] & \Gamma(U_2, E) \arrow[u, "\iota^*"']
\end{tikzcd} \quad \xrightarrow{\mathcal{O}} \tag{6}$$

the functor $\mathcal{O}$ is contravariant because the inclusion map ι between objects in $\mathcal{K}$ goes the in the opposite direction from the map ι between open set objects in $\mathcal{K}$ [?]. The open set U_1 is a subset of U_2, therefore $\iota : U_1 \to U_2$. On the other hand, a set of functions defined on a space U_2 must be defined on the subspace $\iota : \Gamma(U_2, K) \to \Gamma(U_1, K)$ but the reverse does not need to be true. For example, a continuous function defined over the interval $[.25, .5]$ need not be defined over $[0, 1]$ but must exist over $[.3, .45]$. We use the notation $\mathcal{K}^{op}$ to signify that the functor $\mathcal{O}$ is contravariant and $\mathcal{O}$ to denote that it is a sheaf. A sheaf is a presheaf where sections are defined over unions of open sets [?], [?]

The graphic category $\mathcal{H}$ consists of the functor $\mathcal{O} : \mathcal{S}^{op} \to \Gamma(\mathcal{S}, H)$. The category $\mathcal{S}$ is the category of open sets over the graphic base space S and $\Gamma(\mathcal{S}, H)$ is the set of sections ρ over $\mathcal{S}$. The artist is a morphism of sheafs $A : \mathcal{E} \to \mathcal{H}$, which means it is by definition a natural transform [?]. This means that the following diagram must commute

$$\begin{tikzcd}
\mathcal{E} \arrow[r, "A"] \arrow[d, "M"'] & \mathcal{H} \arrow[d, "A(M)"] \\
\mathcal{E} \arrow[r, "A"] & \mathcal{H}
\end{tikzcd} \tag{7}$$

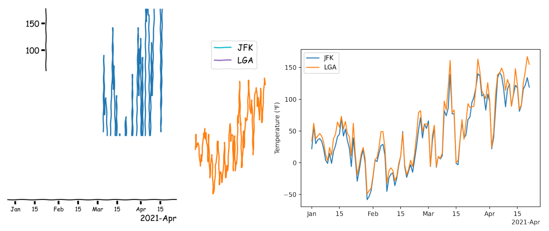
such that the morphisms on the graphic category $\mathcal{H}$ are homomorphic to the morphisms M on the fibers $\mathcal{F}$ [?], [?]. In **??**, we show a method of constructing the artist such that this constraint can be satisfied, thereby ensuring equivariance. We then use this formulation to guide the development of visualization library components in **??**.

3.1 Union of Artists

Visualizations generally consist of more than one graphic, for example two line plots, a legend, axis ticks, spines, and labels. formalize superposition, than break down what that means for composition

These graphics can be generated by disjoint data inputs $(E_i : i \in I)$ into different artist functions A with the same output target space H. We define this composition via the disjoint union of the inputs

$$+ \coloneqq \bigsqcup_{i \in I} E_i \tag{8}$$



(a) + (b) $\oplus$

Fig. 5: In **??**, the outputs of various artist are independently rendered to the same graphic space H. The inputs to the artists in **??** are sections over the same fiber bundle base space K; therefore, they should share transformation functions to produce a consistent image. add a border/make the same size/etc so 1 doesn't run over into two

This approach does not guarantee a coherent image, as illustrated in **??**, because each data is explicitly considered as distinct from the others.

Many times though, graphics are generated from subsets of a larger data set, such as when using multiple views [?], [?] and brush-linked views [?], [?]. In these situations, we consider each of the data inputs as component bundles over a shared continuity K.

$$\pi : E_1 \oplus \ldots \oplus E_i \oplus \ldots \oplus E_n \to K \tag{9}$$

The sections τ on each component bundle E_i over a point $k \in K$ are the elements of the data associated with that point. We use this property to establish which of these elements are the same, because they come from the same fiber, and therefore should share transformation functions to visual elements. This bookkeeping yields graphics with consistent visual encoding of data elements, as shown in **??**.

4 Constraints: Construction and Formal Properties of Artists

Definition 4.1. Following from the artist $A$is a special case of natural transformations

$$A : \Gamma(E, K) \nrightarrow \Gamma(H, S) \tag{10}$$

between the profunctor objects $\Gamma(E), \Gamma(H)$ that preserves continuity and equivariance.

Natural transformations are functions that map functions between categories (functors) to other functors in a structure preserving way [?], [?], [?].

Definition 4.2. We define the natural transformation Artist A as the tuple $(\xi, \nu, Q, \mathcal{E}, \mathcal{V}, \mathcal{H})$ where

1) ξ is a continuity preserving functor
2) ν and Q are equivariance preserving functors
3) $\mathcal{E}$, $\mathcal{V}$, and $\mathcal{H}$ are profunctor categories of sheafs
4)

$$\begin{tikzcd}
E \arrow[r, "\nu"] \arrow[dr, "\pi"'] & V \arrow[d, "\pi"] & \xi^*V \arrow[l, "\xi^*"'] \arrow[r, "Q"] \arrow[d, "\xi^*\pi"] & H \arrow[dl, "\pi"] \\
& K & S \arrow[l, "\xi"] &
\end{tikzcd} \tag{11}$$

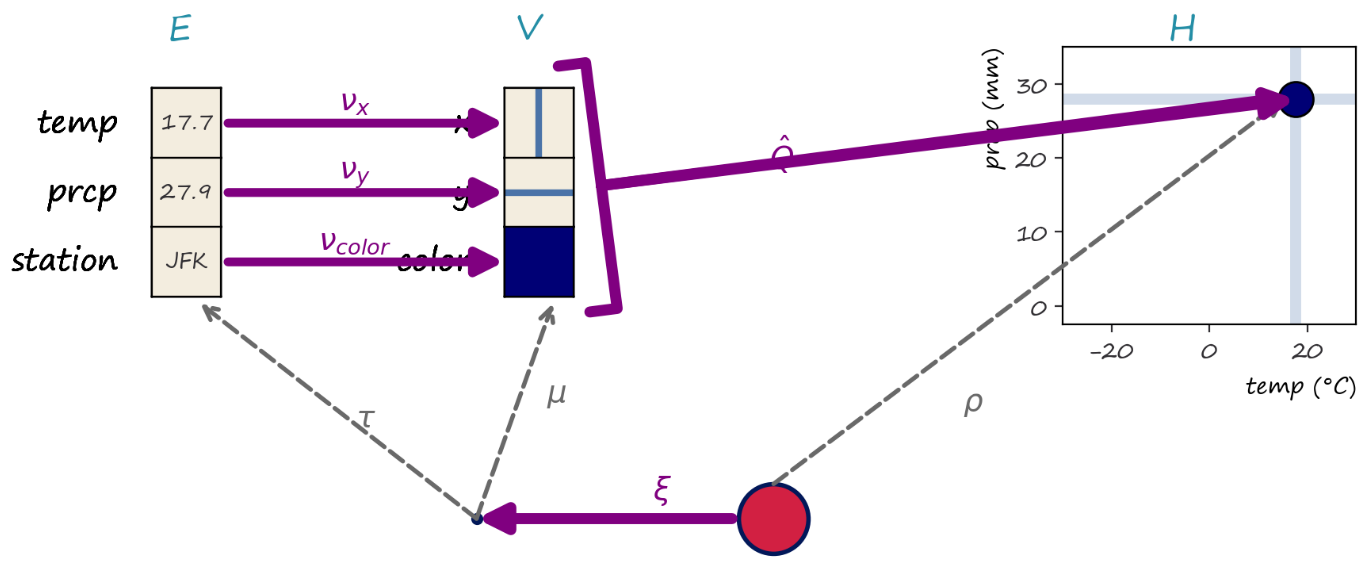


Fig. 6

4.1 Data

We model data as sections of a fiber bundle (E, π, K, F). We encode the continuity of the data as the base space K. does this go in intro? The fiber space F is the space of all possible values of the data **??**. We model the data as the section τbecause, as described in **??**, the section is the map from the indexing space $K$to the space of possible data values F. Spivak's notation, as discussed in **??**, also allows for associating elements in a section with the field it comes from. This allows for field based selection of values, while inclusion allows for continuity (index) based selections.

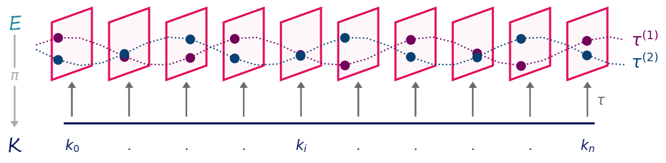


Fig. 7: replace with more concrete

One example of encoding data as a section of a fiber bundle is illustrated in **??**. In this example, the data is a not totally decided yet table of weather station data. Here we are interested in the time series of temperature values in the data, so we encode the continuity as the 1D interval K. In this multivariate data set, the fields we want to visualize are `time`, `temperature`, and `station`. The fiber space F is the cartesian cross product of the fibers of each field

$$F = F_{\texttt{time}} \times F_{\texttt{temperature}} \times F_{\texttt{station}}$$

where each field fiber is the set of values that are valid for the field:

$$\begin{aligned} F_{\texttt{time}} &= \mathbb{R} \\ F_{\texttt{temperature}} &= \mathbb{R} \\ F_{\texttt{station}} &= \{s_0, s_1, \cdots, s_i, \cdots, s_n\} \end{aligned}$$

The section τ is the abstraction of the data being visualized. The section at a point $k \in K$ in the base space returns a value from each field in the fiber.

$$\tau(k) = ((\texttt{time}, t_k); (\texttt{precipitation}, p_k); (\texttt{station name}, n_k))$$

4.2 Graphic

The object of the graphic category $\mathcal{H}$ is the fiberbundle H. The bundle H has the same structure as the data bundle E

$$\begin{tikzcd} D \arrow[r, hook] & H \arrow[d, "\pi"'] \\ & S \arrow[u, bend right, "\rho"'] \end{tikzcd} \tag{12}$$

with a fiber space D embedded in the total space H and a section map $\rho : S \to H$. The attributes of the graphic bundle (H, π, D, S) encode attributes of the graphic and display space

base space S continuity of display space (e.g. screen, 3D print)

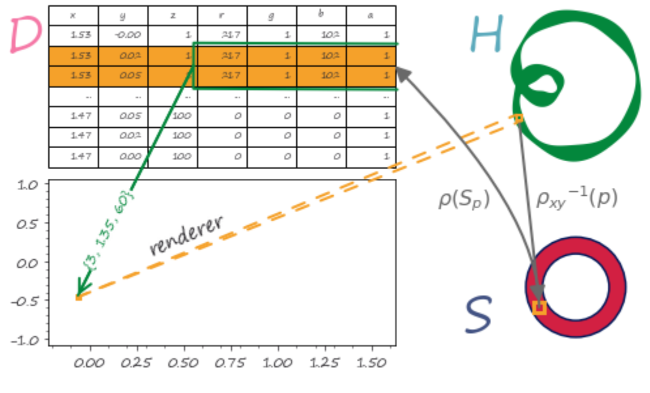
fiber space D attributes of the display space (e.g a pixel = (x,y,r,g,b,a))

section ρ graphic generating function

.

We represent the graphic output as a fiber bundle because it is an abstraction that is generalizable to various output mediums (screens, 3D prints) and types of graphics. In this work, H assumes the the display is an idealized 2D screen and ρ is an abstraction of rendering. For example, ρ

can be a specification such as PDF [?], SVG [?] or an OpenGL scene graph [?], or a rendering engine such as Cairo [?] or AGG [?].



x	y	z	r	g	b	a
1.53	-0.00	1	217	1	102	1
1.53	0.02	1	217	1	102	1
1.53	0.05	1	217	1	102	1
..	..	..	..	..	..	..
1.47	0.05	100	0	0	0	1
1.47	0.02	100	0	0	0	1
1.47	0.00	100	0	0	0	1

Fig. 8

Example 4.1. As illustrated in **??**,

4.3 Visualization Library Components

4.3.1 Graphic to Data: ξ

$$\begin{tikzcd} E \arrow[d, "\pi"'] & H \arrow[d, "\pi"'] \\ K & S \arrow[l, "\xi"'] \end{tikzcd} \tag{13}$$

The functor ξ is a deformation retract, which means.... [?], [?]

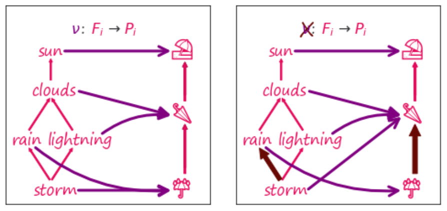
4.3.2 Visual Bundle V

$$\begin{tikzcd} P \arrow[r, hook] & V \arrow[d, "\pi"'] \\ & K \arrow[u, bend right, "\mu"'] \end{tikzcd} \tag{14}$$

4.3.3 Data to Visual Encodings: ν

Visual encoding functions ξ are functors because they are expected to preserve structure on the data and visual side, as discussed in **??**.

This constraint is expressed in our construction of ν


ν: $F_i \rightarrow P_i$
sun
clouds
rain
lightning
storm
$\cancel{\nu}$: $F_i \rightarrow P_i$
sun
clouds
rain
lightning
storm

(a) Artists with shared μ_i renderered correctly (b) Artists without shared μ_i

Fig. 9: Simulation results for the network.

$$\{\nu_0, \ldots, \nu_n\} : \{\tau_0, \ldots, \tau_n\} \mapsto \{\mu_0, \ldots, \mu_n\} \tag{15}$$

We enforce the equivariance constraint

$$\begin{tikzcd} E_i \arrow[r, "\nu_i"] \arrow[d, "m_r"'] & V_i \arrow[d, "m_v"] \\ E_i \arrow[r, "\nu_i"] & V_i \end{tikzcd} \tag{16}$$

One advantage of expressing the data to visual encoding ν_i as a functor from data fiber F_i to P_i is that it lets us formally express that encodings are bound to data and therefore should be consistent across views, as proposed by Hullamn and Qu [?].

4.3.4 Visual to Graphic: Q

Visual encodings to something like marks

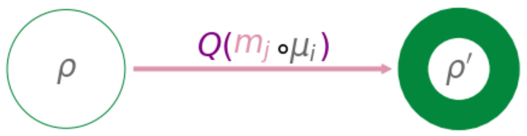


Fig. 10: rework this as a commutative box w/ the r in E row associated w/ this qhat(k)

5 Case Study

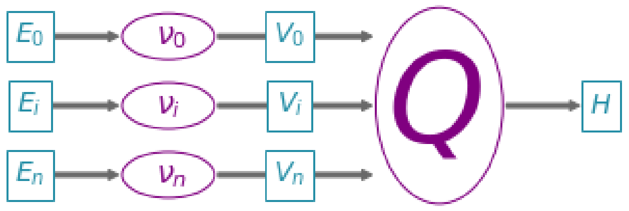


Fig. 11: add in xi!

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

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 $\hat{Q} \leftarrow Q$

$$\begin{tikzcd}
E \arrow[r, "\nu"] \arrow[dr, "\pi"'] & V \arrow[d, "\pi"'] & \xi^*V \arrow[l, "\xi^*"'] \arrow[r, "Q"] \arrow[d, "\xi^*\pi"'] & H \arrow[dl, "\pi"] \\
 & K \arrow[u, bend right, "\mu"'] & S \arrow[l, "\xi"'] \arrow[u, bend right, "\xi^*\mu"'] &
\end{tikzcd} \tag{17}$$

Acknowledgments

The authors would like to thank...

Hannah Aizenman Biography text here.

Thomas Caswell Biography text here.

Michael Grossberg Biography text here.