iVisDesigner: Expressive Interactive Design of Information Visualizations

Donghao Ren¹,², Tobias Höllerer¹ and Xiaoru Yuan²

1. Four Eyes Lab, University of California, Santa Barbara
2. Key Laboratory of Machine Perception (Ministry of Education), Peking University
Motivation

- Design Information Visualizations without the need for Textual Programming or Templates.
- Incorporate Data into Vector Graphics Design Paradigm.
… Live Demo …
Related Work

- **Data Flow Systems**
  - Iris Explorer, AVS, VisTrails.

- **Programming Frameworks and Declarative Languages**
  - Prefuse, ProtoVis, D3.js, Vega.

- **Interactive Design Tools**
  - Tableau.
  - SketchStory.
  - Flexible Linked Axes.
  - Lyra.
System Design: Framework

Data Representation

Enumeration

Objects (Graphics, Mappings, Transformations)

Canvas

Transformed / Aggregated Data Variables / Layout Attributes

Dragging / Brushing: Update with Inverse Mapping

Graphical Elements

Dragging / Brushing: Update with Inverse Mapping
System Design: Data Representation

- **JSON-style Dataset:**
  - Object := { key: Value, key: Value, … }
  - Value := Primitive | Object | Reference | Array of Objects

- **Fixed Schema**

- **Examples:**
  - Tabular datasets.
  - Nested Lists.
  - Graphs: edges as references to nodes.
System Design: Visualization Representation

- **Object-oriented Framework.**

- **Object Classes:**
  - **Graphical Objects:** Map data items to graphical elements.
    - Circles, Lines, Arcs, Polylines, etc.
  - **Guide Objects:** Provide Information for Graphical Objects.
    - Axes, Scatters, Maps, Linear Mapping, etc.
  - **Generator Objects:** Attach derived data back to the dataset.
    - Statistics, Range, Expression, Brushing, ForceLayout, etc.
  - **Components:** Nest objects inside (for example, glyphs).
System Design: Graphical Objects

- **Object Classes:**
  - **Graphical Objects:** Map data items to graphical elements.
    - Circles, Lines, Arcs, Polylines, etc.

**Diagram:**

```
Dataset -> Enumeration -> Data Items -> Graphical Object -> Canvas
```

- **Enumeration:**
  - Root: `cars set (406)`
  - Data Items: `name` and `mpg`

- **Graphical Object:**
  - Select all “Cars” -> “Circles” Object
  - “Lines” Object
System Design: Guide Objects

- Object Classes:
  - **Guide Objects**: Provide Information for Graphical Objects.
  - Axes, Scatters, Maps, Linear Mapping, etc.
System Design: Generator Objects

- Object Classes:
  - **Generator Objects**: Attach derived data back to the dataset.
  - Statistics, Range, Expression, Brushing, ForceLayout, etc.

Example: Cars → PCA → Coordinates for Each Car
System Design: Components

- **Object Classes:**
  - **Components:** Nest Objects Inside.
System Design: End-User Interactions

- Designing (Provisioning) for End-User Interactions:
  - Dragging and Brushing
Scatterplot Anatomy:

- **Axis**: Define 1D positions.
- **Scatter**: Define 2D positions using two axes.
- **Circles**: Draw a circle for each element on the data selection, using the location provided by the Scatter object.
Design Examples: Graph Data

- Data: Character Co-occurrence Graph in *Les Misérables*.
Design Examples: Scatterplot with Glyphs

- SPPC with Star Glyph for Cars Dataset.

Component Object: Design Glyphs (Sub-Visualizations).
Design Examples: Map with Timelines

- Beijing Air Pollution Dataset (PM2.5).
- Component: Timeline Glyph for Each Station.
Design Examples: Microblog Data

- Weibo User Data: Friends, Followers, Tweets.
- Design: Map + Scatterplots, with links.
Design Examples: Dynamic Datasets

- Packet Flow Streaming Dataset
Beijing Air Dataset:
36 Stations
87 Measurements on Each Station
- time, SO2, PM2.5, ...
Design Examples: End-User Interaction / Level of Detail
Performance and User Feedback

• Rendering Performance
  • Handle ~2000 data items at an interactive rate in a web browser.
  • Further optimizations are possible.

• Feedback from User Survey
  • High scores on:
    • Expressive, flexible and useful, and good for designing visualizations for different types of data.
  • Lower scores on:
    • Easy to use, easy to understand.
Discussion & Conclusion

• **Goals:**
  - Interactively Design Information Visualizations.
  - Provision for End-User Interactions.

• **Approach:**
  - Data-driven Vector Graphics Editing Paradigm.
  - Represent designs with Graphical, Guide, Generator and Component objects.
Discussion & Conclusion

• **Pros:**
  • Expressive: Able to construct a variety of different designs.
  • Extensible: Templates / New Objects.
  • Web-based: Easily embed designs into websites or web applications.

• **Open Source:**
  • [https://github.com/donghaoren/iVisDesigner](https://github.com/donghaoren/iVisDesigner)

• **Still Improving:**
  • Send your feedbacks!
  • Refer to [https://donghaoren.org/ivisdesigner/](https://donghaoren.org/ivisdesigner/) for future updates.
Future Work

• More object types, and support for:
  • Scale Indicators.
  • Coordinate Spaces (Cartesian, Polar, etc).
  • Recursive Layouts (TreeMaps, etc).

• Usability Improvements:
  • Hints & Error Reporting.
  • Additional Higher-level Templates.
  • Automatic Design Recommendations.

• Ongoing Work:
  • Integration into an immersive Situation Room (UCSB Allosphere).
Many thanks to:
My Advisors: Prof. Tobias Höllerer at UCSB and Prof. Xiaoru Yuan at Peking University.
Fellow Students: Byungkyu (Jay) Kang, Zuchao Wang, Siming Chen, Qingya Shu, Limei Che, and Min Lu.
Anonymous Reviews.
Funding: NSFC No. 61170204, MURI Grant No. W911NF-09-1-0553, Office of Naval Research N00014-14-1-0133.

Questions?