Informations Visualisierung
SoSe 19

Aaron Winziers & Benedikt Lüken-Winkels

July 11, 2019

Contents

1 Introduction

1.1 Visualisation-Basics

Definition: Visualisation the use of computers or techniques for comprehending data or to extract knowledge from the results of simulations, computations, or measurements (not manually by humans)

Definition: Information Visualization the communication of abstract data through the use of interactive visual interfaces.

- Combine different kinds of information in one graphic (geographical, temporal, historical, numeric, etc.)
- Sharing and visualising abstract data, without physical representation
- Visualisation is not:
  - **Scientific Visualization** Visualization of data with a concrete physical representation (non-abstract data)
  - **Computer Graphics** Technical and mathematical aspects of visualization
  - **Graphic Design** Aesthetic graphical representation
- **Example** Treemap
  - representation of a hierarchy of a filesystem
  - no border used for a square (compression)
  - light effect shows curvature, indicating where the squares/areas end
  - $\Rightarrow$ only 4 pixels needed instead of 9
  - Several drawbacks (alternative: tree view)

Ways to display Abstract Data

- Text and Tables
- Hierarchies and Graphs
- Composed data (Multivariate data): Example Napoleon (Slide 1)
- Time series: multivariate data with time as a dimension

Visualization process

- graphical user interface
- interaction to create and manipulate the visualisation (**Visual steering**)

2 Infographics

2.1 Diagrams

Simple Diagrams

- Line Charts
- Bar Charts
- Pie Charts

Pie charts

- applicable to part-whole relation
- Several issues
  - difficult to compare values within a chart
  - difficult to compare differences between pie charts

Other Diagrams

- **Timelines** align temporal information along an axis
- **Sparklines** Reduced to show trend and the change of values over time - a sparkline is a small intense, simple, word-sized graphic with typographic resolution.

2.2 Metaphors and Symbols

Make constructs/concepts more accessible/imaginable

2.3 Symbols/Pictograms

highly simplified representation of objects and activities. Very suitable for depicting metaphors

Isotype using pictograms to convey statistical information. Quantity is better represented by the number of pictograms than by the size of a pictogram.

2.4 Infographics

Definition Infographics Information graphics or infographics are graphic visual representations of information, data or knowledge. These graphics present complex information quickly and clearly, such as in signs, maps, journalism, technicalwriting, and education.

- Eyecatcher to get people interested in the presented data
- Contain few text
- Self-explanatory
- Should tell a **story** ⇒ express an opinion

Elements of an Infographic

- Story
- Graphics
  - Illustrative
  - Simplified
- Text
  - Keywords and short texts
- Diagrams
  - Connected to graphics.

Infographics vs Information Visualization

- Infographics
  - Manually created
  - Specially designed for a particular data set
  - Self-explanatory
- Information Visualisation
  - Automatically computed
  - Suitable for a variety of data sets
  - Not necessarily self explanatory

3 Visual Perception

75% of information is perceived visually

3.1 Visual Memory

- The brain fills empty gaps
- Distraction by environment (contrast/structure)
- $\Rightarrow$ visual perception is selective (change blindness)

3.2 Visual Information Processing

3 Phases of processing

1. Simple patterns and colors are recognized
2. Action system: reflexes
3. Visual working memory/visual query

Human Eye

Usage of the properties of visual perception (Anticipation, pattern recognition)

- Eye Tracking (works by measuring the reflection form the eye's curvature)

Peripheral Acuity Center of vision:

- In focus
- Color and brightness
- Blurry
- Only brightness

3.3 Color Perception

3-Color-Theory

- Each color consists of rgb

Opponent-Color-Theory

- After image effect: color-receptors are getting exhausted, so white cannot be 'produced'
- three chemical processes with two opponent colors each
- Color is perceived by the difference between the opponent colors

⇒ Color and brightness are relative

Design Recommendations

- Emphasize with color
- Differences with brightness
- Coding of categories: max 6 to 12 different colors
- Color scales should vary in color and brighntess
- Color perception depends on culture
- Motion to grab attention/indicate a relation
- Strong colors/contrast can cause interta (ghost images)

3.4 Preattentive vision

- Detect patterns before an eye movement
- Motion is preattentive
- ⇒ Use preattentive patterns to encode information (spot an outlier)

3.5 Pattern Recognition

- Edge detection - Differences in brightness, color, texture or motion
- Simple patterns (detect small distortions)
- Complex patterns
- Object recognition (compare observation with learned patterns to recognise an object)

3.6 Motion recognition

Different elements perform similar motions

- Recognize patterns to identify object
- Recognize change after each frame
- Movements seem related, when they are in synch
- ⇒ Indicate a relation with a synchronous animation
- Motion can induce causality

3.7 Gestalt Psychology

- **Proximity** - Elements which are placed close to each other are perceived as a group.
- **Similarity** - Similar elements (form, color) are perceived as a group.
- **Connectedness** - Connected elements are perceived as one object
- **Continuity** - For humans it is easier to group continuous elements than elements with abrupt changes of direction.

3.8 Three-Dimensional Perception

Reconstruction of depth information

- Stereoscopic vision (in particular at close range)
- More depth cues: depth of field, perspective, shadow, scale, contrast, motion parallax (how near and far objects will move across the retina of an eye as we move along in the world)
- Prior knowledge

Main Issues

- Navigation is difficult
- Occlusion

4 Visualizations of Hierarchies

Hierarchy = Tree

4.1 Node-Link

Types

- Phylogenetic Tree
- Radial Tree
- Cone Trees

Advantages

- Intuitive
- Hierarchy immediately recognizable
- Very flexible layout

Disadvantages

- Edges require space
- Difficult to add labels
- Degenerated trees are difficult to represent

4.2 Indented Outline Plots

Examples/Types

- Windows explorer
- XML File

Advantages

- Very readable
- Easy to add labels
- Familiar; used daily by many people (file explorer)
- Degenerated trees can be represented
- Hierarchy is well recognizable

Disadvantages

- Inner nodes require space
- Somewhat inflexible layout

4.3 Icicle Plots

Examples/Types

- InfoVis Toolkit
- Sunburst
- Hierarchical Edge Bundles

Advantages

- easy to add labels
- hierarchy is well recognisable
- flexible layout
- uses screen space efficiently

Disadvantages

- somewhat less intuitive
- available width for children restricted by the width of of their parents.

4.4 Treemap

Examples/Types

- Treemap
- Information Pyramids
- CodeCity

Advantages

- area of leaf nodes can be used
- can fill arbitrary shapes e.g. Voronoi treemaps)
- inner nodes require less space
- edges require (almost) no space

Disadvantages

- less intuitive
- hierarchical structure difficult to recognise
- difficult to add labels

4.5 Empirical Study of Efficacy

Recommended

- Node-Link Diagrams
- Icicle Plots
- (Indented Outline)

Questionable

- Treemap
- radial layouts

Conclusion

- Empirical evaluation is just beginning
- More research is needed to make well-founded design recommendations
- There is also a lack of domain-specific results.

5 Visualization of Graphs

Graph Drawing - The art of drawing a diagram of a graph to facilitate understanding of relations between objects

Application

- Map-drawing: indicate multiple data sets in one map (London Underground)
- Ego(-centric) network: graph with personal connections

Visual Encoding

- Thickness, color of edges
- Color of nodes

part

Aesthetic Criteria Readability does not induce aesthetic

- Minimize edge crossings
- Minimize drawing area
- Minimize edge length
- Minimize number of bends
- Maximize symmetry
- Uncover clusters
- Maximize continuity amongst paths

5.1 Layouting algorithms

Radial Layout

- fair node weight, every node's representation is equal
- lots of edge crossings
- applicable, if there is no further info about the data

Force-Directed Layout

- force edges to a certain length
- reorder nodes
- try to find equilibrium, where the forces cancel out each other

Hierarchical Layout

- for cyclic structures: flip the edges that close the cycle while drawing the graph
- depth first search provides a topological ordering of the nodes
- sort nodes on the lower layer until the bottom is reached, then go back to start
- to have a clean layout, put in dummy nodes as a spacer

Orthogonal Layout

- edges follow grid (orthogonal paths)
- shape metrics
  - describe the path the edges take by turns
  - evaluate the paths

Edge Bundling

- structured radial layout
- bundle edges with the same direction

5.2 Matrix visualization of Graphs

Adjacency Matrix

- indicate an edge in a matrix
- uncovering clusters is hard

Layouting

Compound graphs

5.3 Visualization of dynamic graphs

Dynamic graph: sequence of graph states

5.4 Approaches to dynamic graphs

Animation Animation of the sequence of graphs

- **Local goal** - Optimal graph layout
- **Preserving the mental map**

Time Line - Visualization of the sequence of graphs as a series of static images along a time line. Examples:

- TimeSpiderTrees, cirular layout, each ring is one graph
- TimeRadarTrees, cicular layout, outer circles are a representation of the inner. The inner circle shows incoming edges, the outer shows outgoing

6 Multivariate data and time series

Multivariate Data

- Several variables/dimensions per object/observation
- Types of variables - numeric, categorial
- Easy to represent in a table

Descriptive Statistics

- Mean
- Median
- Quartile
- Mode
- Standard Deviation
- Standard Error

6.1 Graph types

Boxplots box showing 50 percent of data, outer borders not standardized

Fan Chart wide part shows the mean (similar to the box plot)

Histogram Frequency distribution shown as bar chart (value range split into intervals)

Extended table - With color coding, bars and icons

Sparklines in tables

Scatterplot

Scatterplot matrix - creating multiple 2-dimensional scatterplots in a matrix

Parallel Coordinates

Star Plots - radial variant of parallel coordinates

7 Software Visualization: Code

Software visualization - Visualization of artifacts related to software and its development process

- Structure
  - Software architecture
  - Dependencies between software artifacts
  - Data structures
- Behavior
  - Execution of an algorithm
  - Runtime behavior
  - Program state
- Evolution
  - Development history of a software system
  - (Sequences of) source code changes
  - Team buildung and development

Pretty Printing

- Line breaks to discern statements
- Indentation to make the structure more explicit

Syntax Highlighting

8 Interaction

Shneiderman's Taxonomy of Information Visualization Tasks

- **Overview**: see overall patterns, trends
- **Zoom**: see a smaller subset of the data
- **Filter**: see a subset based on values, etc.
- **Details on demand**: see values of objects when interactively selected
- **Relate**: see relationships, compare values
- **History**: keep track of actions and insights
- **Extract**: mark and capture data

Shneiderman‘s Information-Seeking Mantra - Overview first, zoom & filter, then details-on-demand

Categories of Interaction Techniques

- **Select** - Mark something as interesting
- **Explore** - Show me something else
- **Encode** - Show me a different visual representation
- **Reconfigure** - Change the spatial arrangement
- **Abstract/Elaborate** - Show me more or less detail
- **Filter** - Show me something conditionally
- **Connect** - Show me related items

Standard vs. Semantic Zoom

- **Geometric Zooming (Standard)** - View depends on the physical properties of the presented object
- **Semantic Zooming** - A different visual representation is chosen depending on what meaning of the presented object should be preserved.

9 Software Visualization: Architecture

Software Architecture - Architecture is the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution

9.1 Common Architectures

Pipes and Filters Input stream providing data, putting it into a pipe of filters

Layered Systems Layers provide functionality of upper layers (radial or stacked). Radial: small core, Pyramid: neutral representation

Blackboard-driven Different processes share info on one blackboard

UML

9.2 Reverse Engineering

Reverse engineering is the process of analyzing a subject system to create representations of the system at a higher level of abstraction. → used for automatically creating architecture visualizations

9.3 Enriched Node-Link Diagrams

Visuialize/Encode software metrics. Aggregation of information to simplify.

Software Metrics A software metric is a measure of some property of a piece of software or its specifications.

- software metrics provide additional information
  - automatic computation
  - usually: multivariate data
- may increase understanding, help to find problems

Class Blueprint Categorize methods by name and access attributes into:

- **Initialization -** methods with substring "init" or "initialize", constructors
- **Interface -** public or protected methods only invoked by init layer within the same class
- **Implementation -** private methods invoked by other methods in the same class
- **Accessor -** methods to get and set the values of attributes (getter/setter)
- **Attributes -** all attributes of the class

Depenecies Viewer Visualize package graph and dependencies between packages and methods

Dependency Structure Matrix DSM Detect cycles and indirect cycles with highlighting

Software Cities and Maps 2D plane represents system. Hierarchy shown with trees/dimesions. 3rd dimension can be used to show other metrics, like evolution/age/dependencies

Summary Ad-hoc diagrams hard to understand without explanation. With reverse engineering automatic creation for specific techniques are possible

10 Lecture

10.1 Dynamic Program Visualization

Dynamic Data Acquisition invasive method, monitoring the behavior of a program before/after each instruction. Might alter the program execution.

- Instrumentation
  - before/after each instruction
  - at certain program points
    * before/after loops
    * before/after method calls
    * defined by user (→ interesting events)
  - data structures
    * Whenever data is changed (daemon, observer)
- Parallel thread, which reads memory
- Capture messages (for distributed programs)
- Virtual Machine/Interpreter
- Special Purpose Hardware (→ embedded systems)

What data is to be captured?

- Program position (PC, called method, line number in source code)
  - Problem for compiled programs: mapping machine instructions to line numbers in the source code
- Values of program variables
- Heap contents of the program
- For messages:
  - Point in time (enables temporal ordering of messages, which have been captured at different computers, Problem: local vs. global time)

Architectures for Algorithm Animation

- Ad Hoc Visualization and Libraries
  - Don‘t use a tool at all. Implement everything from scratch.
  - Use libraries with graphical abstractions, control-elements, etc.
- Special data types
  - Program the algorithm with datatypes which have built-in visualizations
- Post-Mortem Visualisierung
  - Record an event log or animation script during the execution of the algorithm.
  - Animation after the execution of the algorithm.
- Interesting Events
  - Annotate interesting program points
  - Send events to concurrently executed animation (view)
- Declarative
  - Separation of annotation and algorithm
  - Demon monitors state changes and visualizes the state
- Semantics-Directed
  - automatic visualization by visual interpreter or debugger for the programming language

10.2 Visual Debugging

Slices are parts/slices of the huge dependency graph in a program

Static Slice How **can** a variable changed by other code points. Slice is a small part

Dynamic Slice How **is** a variable changed by other code points. Slice is a small part

Execution Slices Sequence of program points.

Dice Difference of two Slices.

X-Slice (Heuristic) Compare a run with failing and compiling input. Only the failing program points are highlighted. Color coding coverage data by failure propability and evidence for failure.

Test Blueprint Highlight non-executed program points in the Class Blueprint.

10.3 Software Evolution

aka Software Development Process ⇒ Software changes in its lifetime.

Software Archive version control/collection of the history of a program of any kind.

Color-coding

- Line Representation: indentation/different metrices
- Code Age: when was a file/line changed
- Pixel Representation
- Version-specific Code: highlight eg platform specific code
- Depth of nested blocks
- CVS Scan: different versions for a file with LOC as bar height.

Evolution Matrix Classes are represented as boxes. Box height and width encode a certain metric. ⇒ No insight on program structure

Call Graph Which function calls which function (low level info). Encode program structure. **Edge splatting** (the more often an edge is drawn the more intense it color gets) shows call clusters.

10.3.1 Visual Data Mining in Software Architecture

Data Mining Process Starting with a version control program (git)

1. Analysis
2. Extraction
3. Data Mining
4. Visual Data Mining

Coupling

- Evolutionary Coupling artifact are related, when they are changed together.
- Logical Coupling artifacts are related, when they are programmatically calling each other.