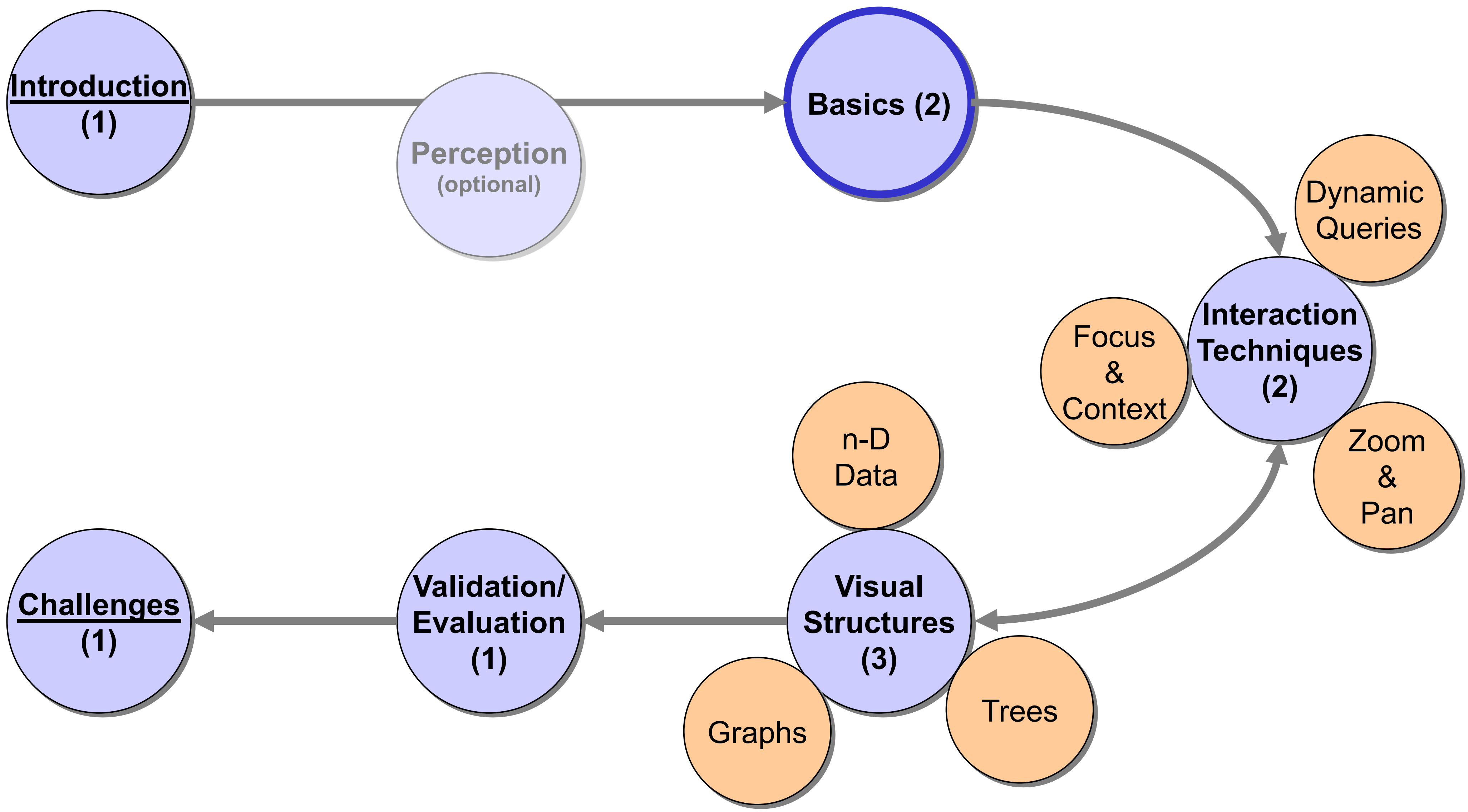
Information Visualization

2. Basics

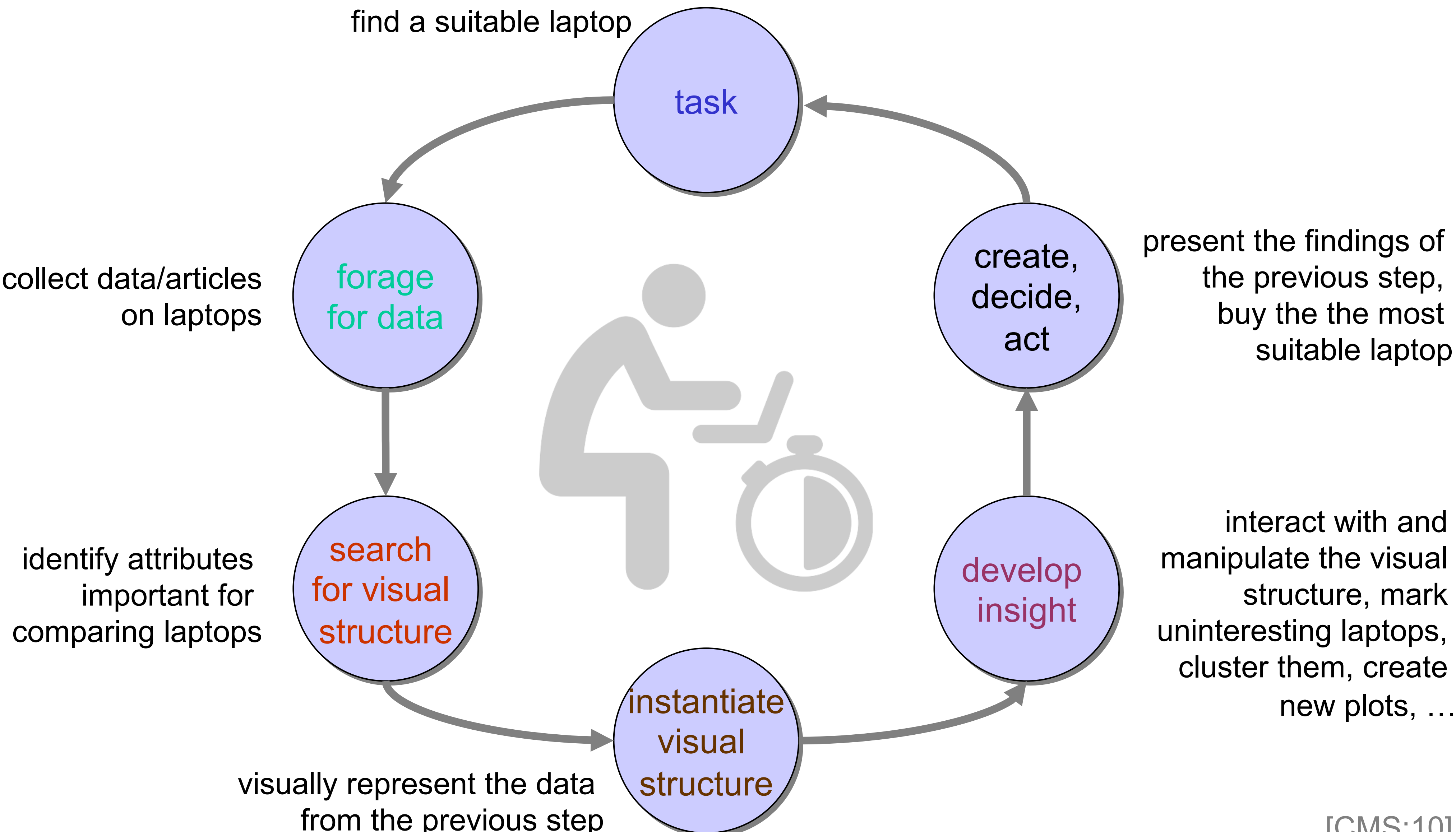
TNM111: 10 Lectures



2.1 Knowledge Crystallization

- Def.: Knowledge Crystallization Task (KCT) [CMS:10]
  - A person collects information for any task and tries to bring a sense or an order into it by constructing a suitable form of representation (*schema*) [Russel et al, The Cost Structure of Sensemaking, CHI '93, 1993]
  - The aim is to perform an action, to make a decision, to send a message, etc. on the basis of this schema
  - In InfoVis, the schema is a suitable visual structure
  - For describing these processes in context of InfoVis, we can use the following diagram …

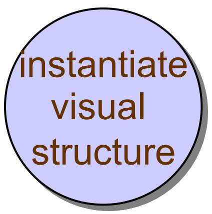
2.1 Knowledge Crystallization

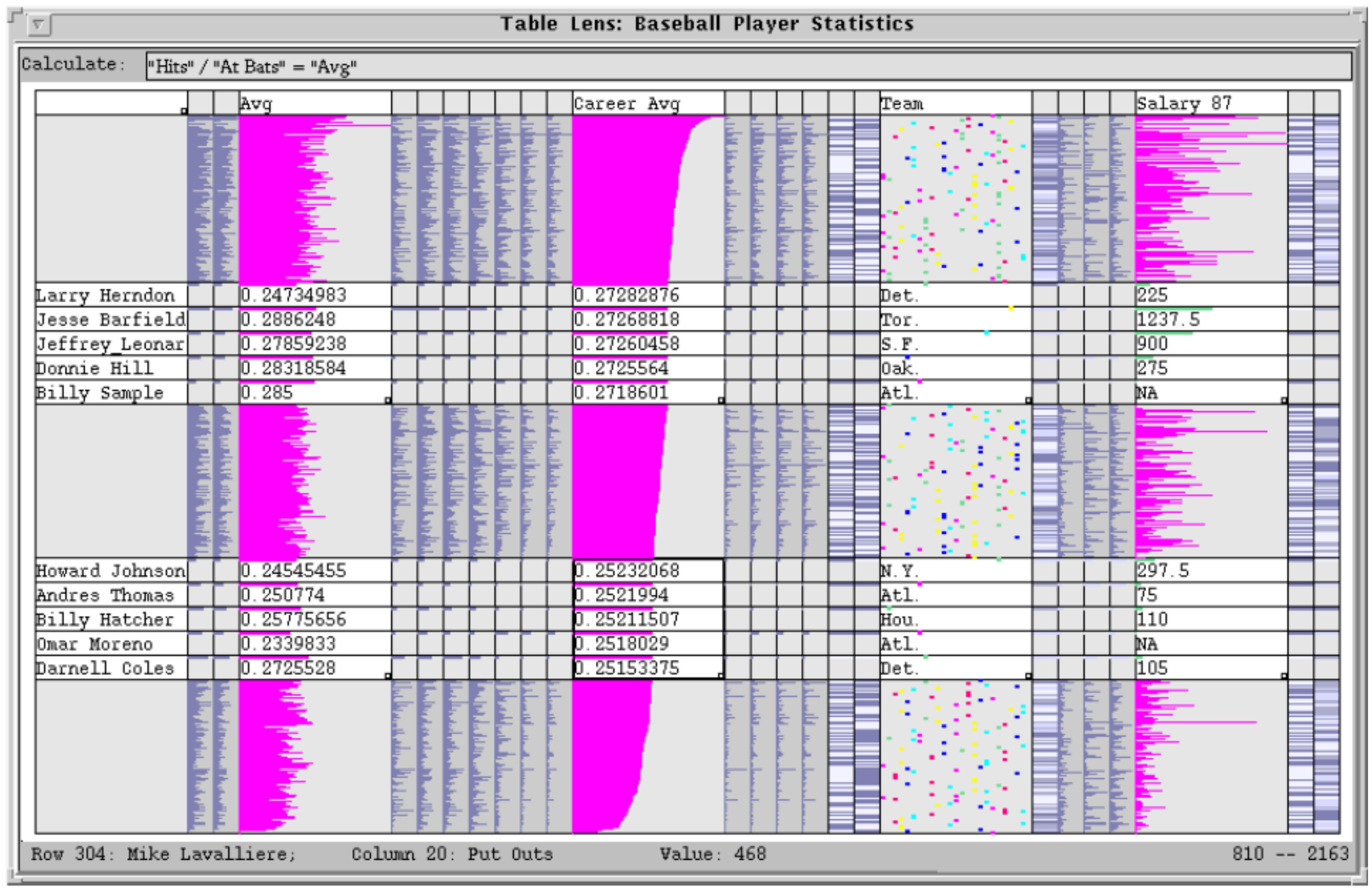


[CMS:10]

2.1 Knowledge Crystallization

- InfoVis meets the most parts of the KCT model well
- <u>Example:</u> Table Lens Tool (Inxight Software, bought by SAP)

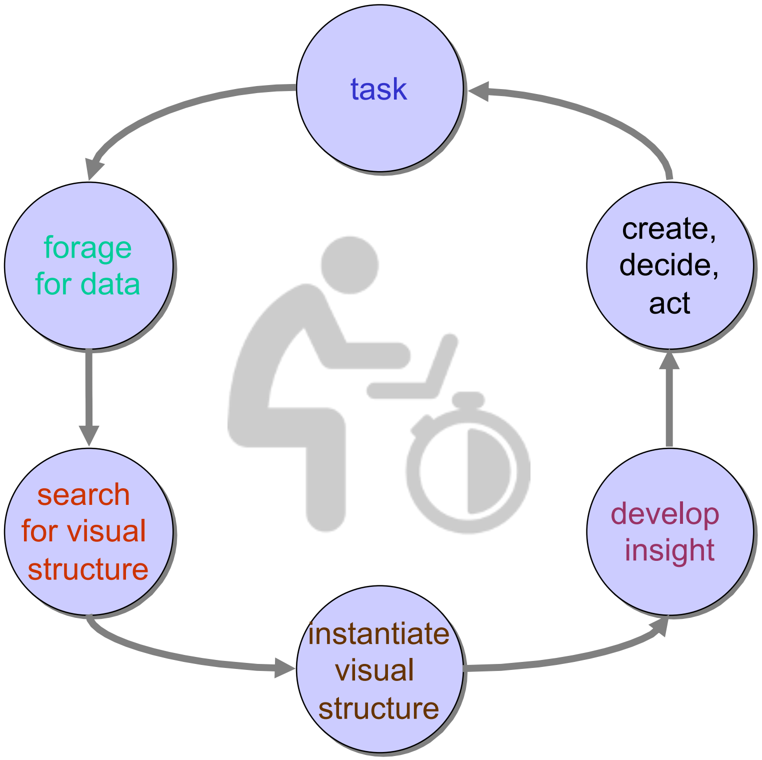




Manipulate cases or variables in a spread sheet, in which data are visually represented as bars, dots, etc.

2.1.1 Cost Structure

- Each KC-task has specific **costs**

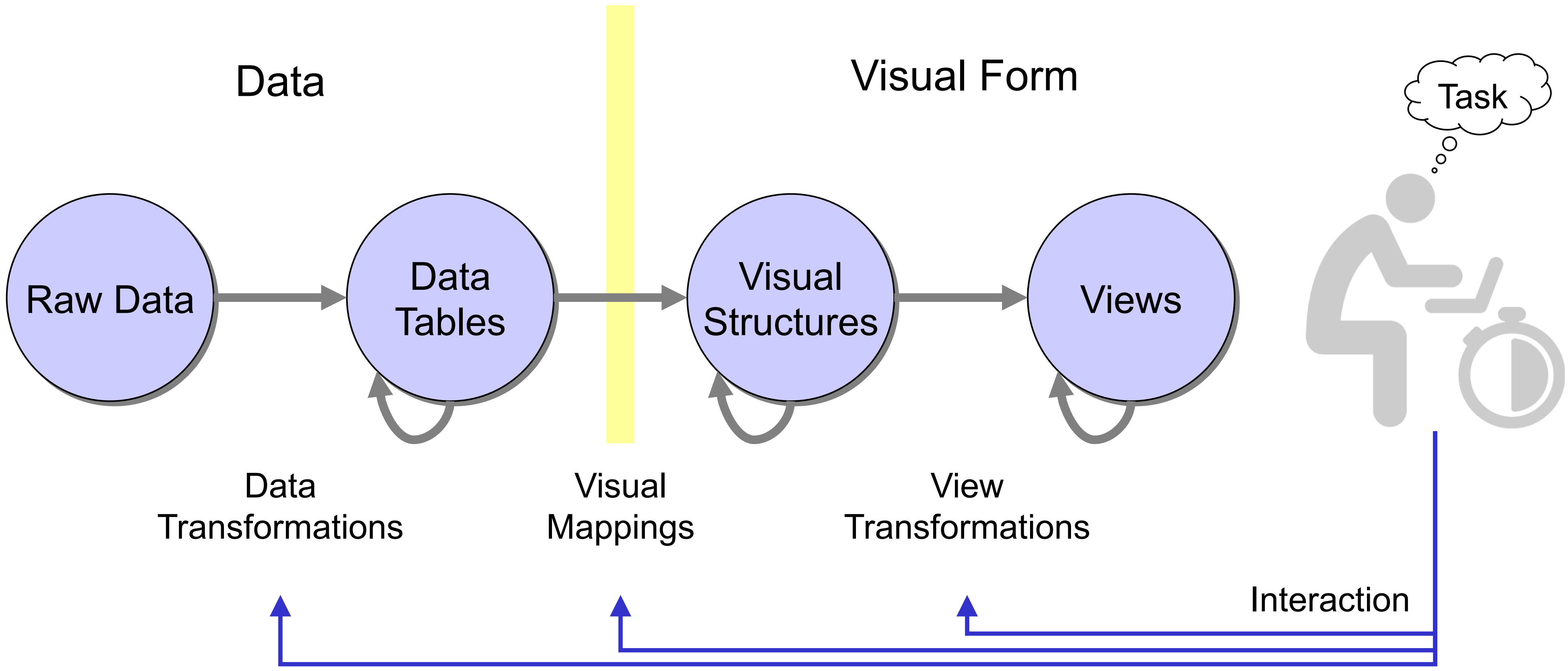


- InfoVis improves the cost structure of the work to get information
  (= all things to support knowledge transfer)

- Such as cost structure function could be a measurement for the usability of a visualization
  - Unfortunately, not existing because hard to specify in practice

[CMS:14]

2.2 InfoVis Reference Model



- Raw Data: specific idiosyncratic formats
- Data Tables: relations (cases by variables) + metadata
- Visual Structures: spatial substrates + visual elements + graphical properties
- Views: graphical parameters (scaling, zooming, clipping, …)

[CMS:17ff]

2.2.1 Data Transformation

- Raw Data
  - Spreadsheets
  - News (documents)
  - Program output
  - Data of mobile phone providers
  - …
- Aim: transformation into a more structured form
- Reason: easier to map to a visual form
- Example:

  BMW, 1, 120d, 36983, 190, … ; Volvo, XC60, T6, 43832, 140, … ; …

2.2.1 Data Transformation

- Data Transformation
  - Process of converting raw data into data tables
  - Used to build AND improve data tables

- Data Tables
  - Cases/Items/Objects
  - Variables (Attributes)
    - Nominal
    - Ordinal
    - Quantitative
  - Values
  - Metadata (e.g., V-Type)

a possible data table

Model	V-Type	120d	T6
Manufac.	N	BMW	Volvo
Series	N	1	XC60
ID	O	36983	43832
HP	Q	190	350

2.2.1 Data Transformation

- Problems during transformation (resp. of raw data)
  - Errors
  - Missing data
  - Variable types
  - Variable formats
  - Table structures

Model	120dd	T6
Manufac.	BMW	
Series	1	XC60
ID	AKBH	43832
HP	190,6	350

	120d	T6
Manufac.	BMW	Volvo
Series	1	XC60
ID	36983	43832
HP	190	350

Now, these are metadata! The table structure itself counts to metadata

2.2.1 Data Transformation

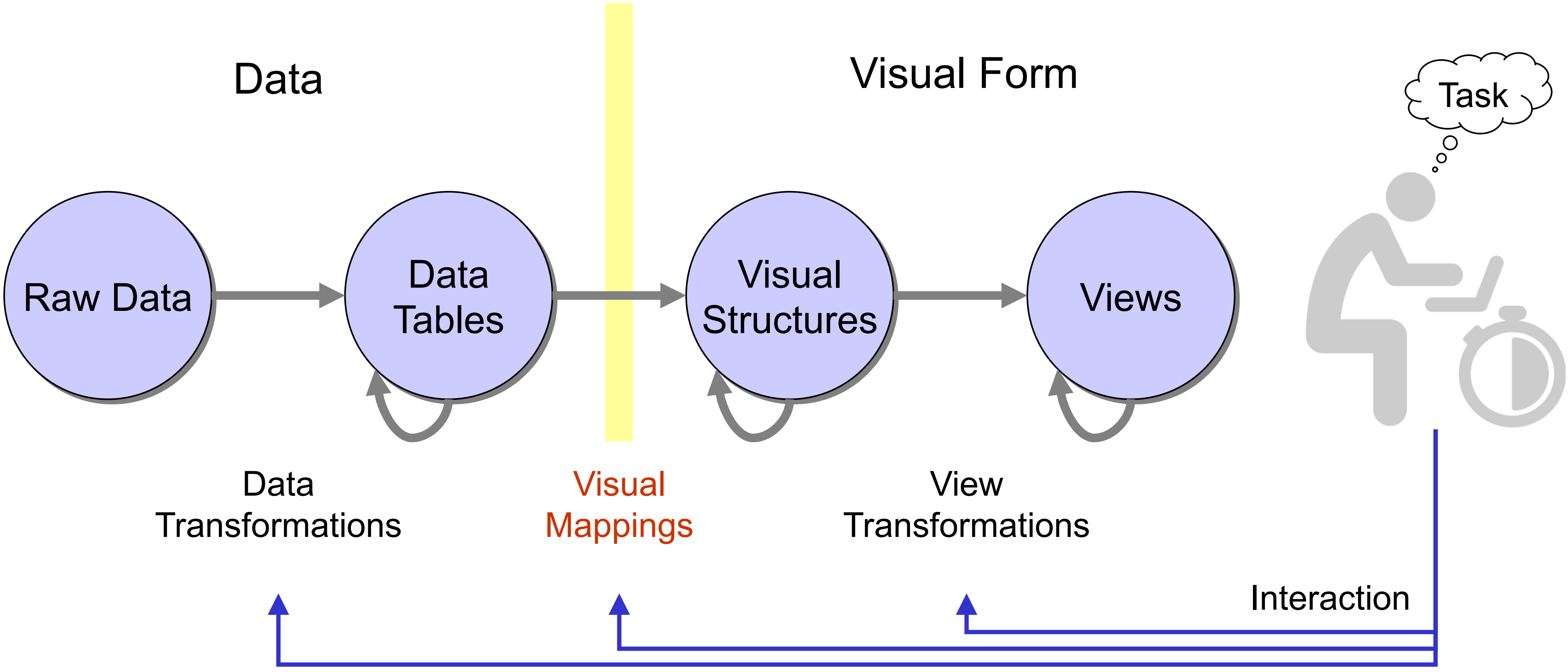
- Dimensionality (of data)
  - Depends of the number of attributes (variables)
  - 1D → univariate data
  - 2D → bivariate data
  - 3D → trivariate data
  - ≥4D → multivariate data (*hypervariate* data)

- In InfoVis, we try to map all input data to 2D (or sometimes 3D) (visual mapping) in order to create a **visual encoding** of the data items that results in a final **visual representation/structure**

[Spe:34ff]

2.2.2 Visual Mapping

- Reminder



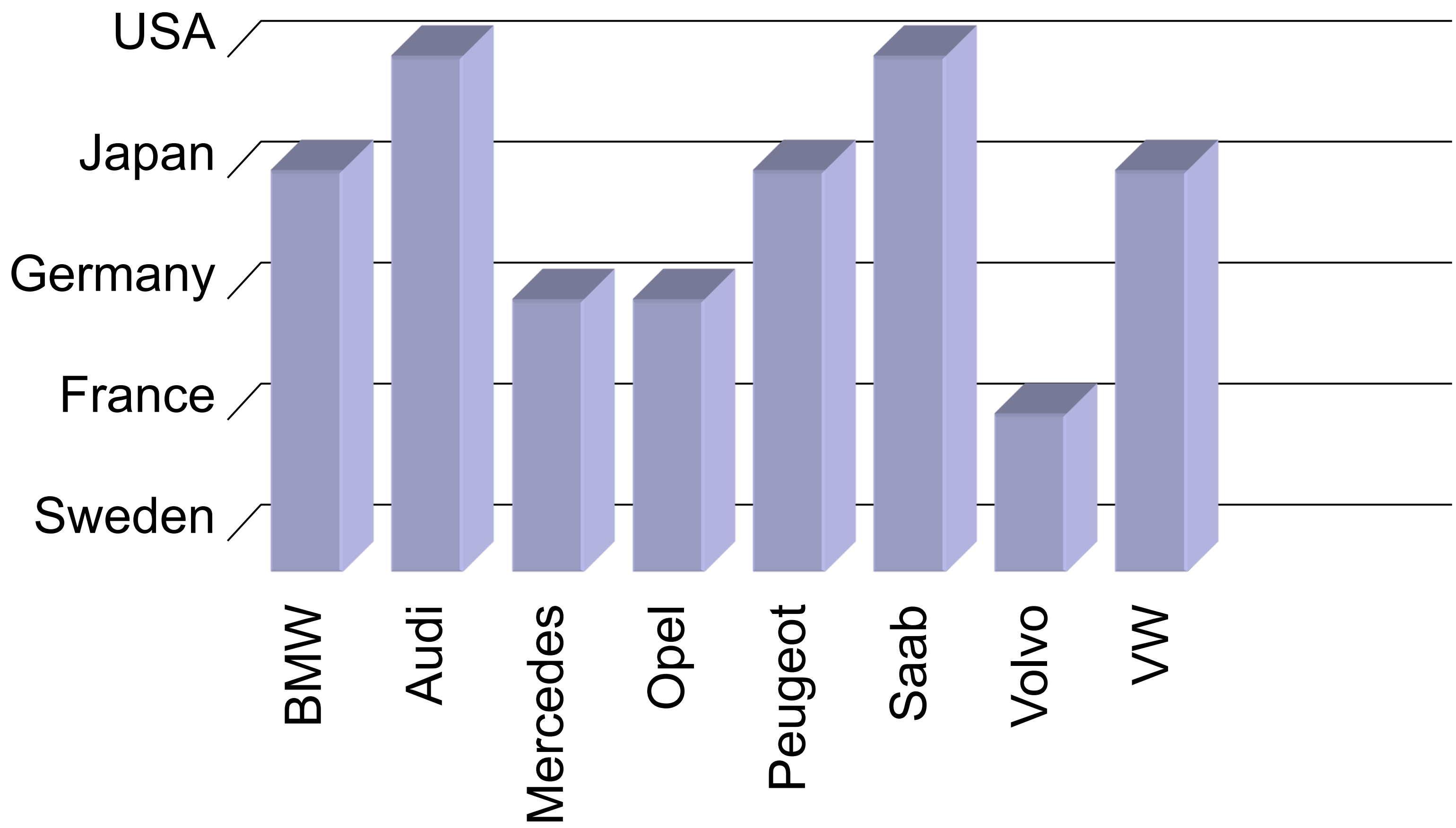
2.2.2 Visual Mapping

- Visual structures extend a spatial medium with visual elements and graphical properties in order to code information
- The mapping to a visual structure should reflect the data table
- To find a good mapping is often very difficult and the design space is huge
- Expressiveness of the mapping
  - A mapping is *expressive*, if all and only the data in the data table are also represented in the visual structure
- Effectiveness of the mapping
  - A mapping is more *effective* than another, if it is faster to interpret, can convey more distinctions, or leads to fewer errors

[CMS:23ff]

2.2.2 Visual Mapping

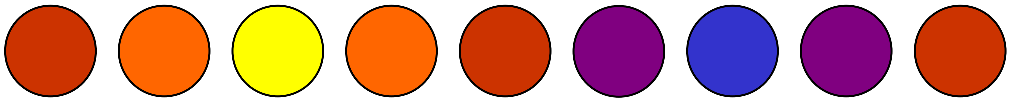
- Expressive Mapping?



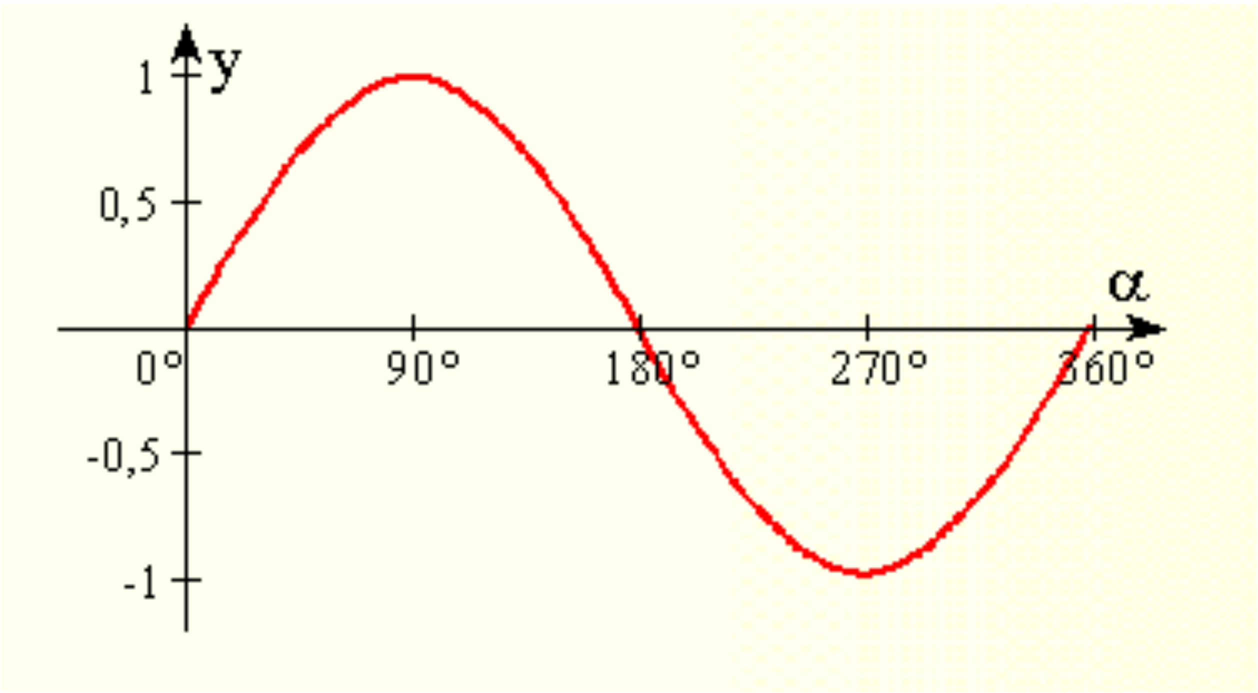
2.2.2 Visual Mapping

- What mapping is more effective for the sine wave?

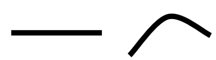
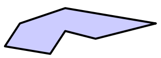
a)



b)



2.2.2 Visual Mapping

- Visual perception is *very important* for visual mapping and design of visual structures [→ TNM093, suppl. material]
- Visual Structures
  - Spatial substrate (medium) is extended by
    - visual elements (also called: marks)
      - Points 
      - Lines 
      - Areas 
      - Volumes 
    - graphical properties (also called: visual channels/features)
      - connection, enclosure, color, animation, saturation, …

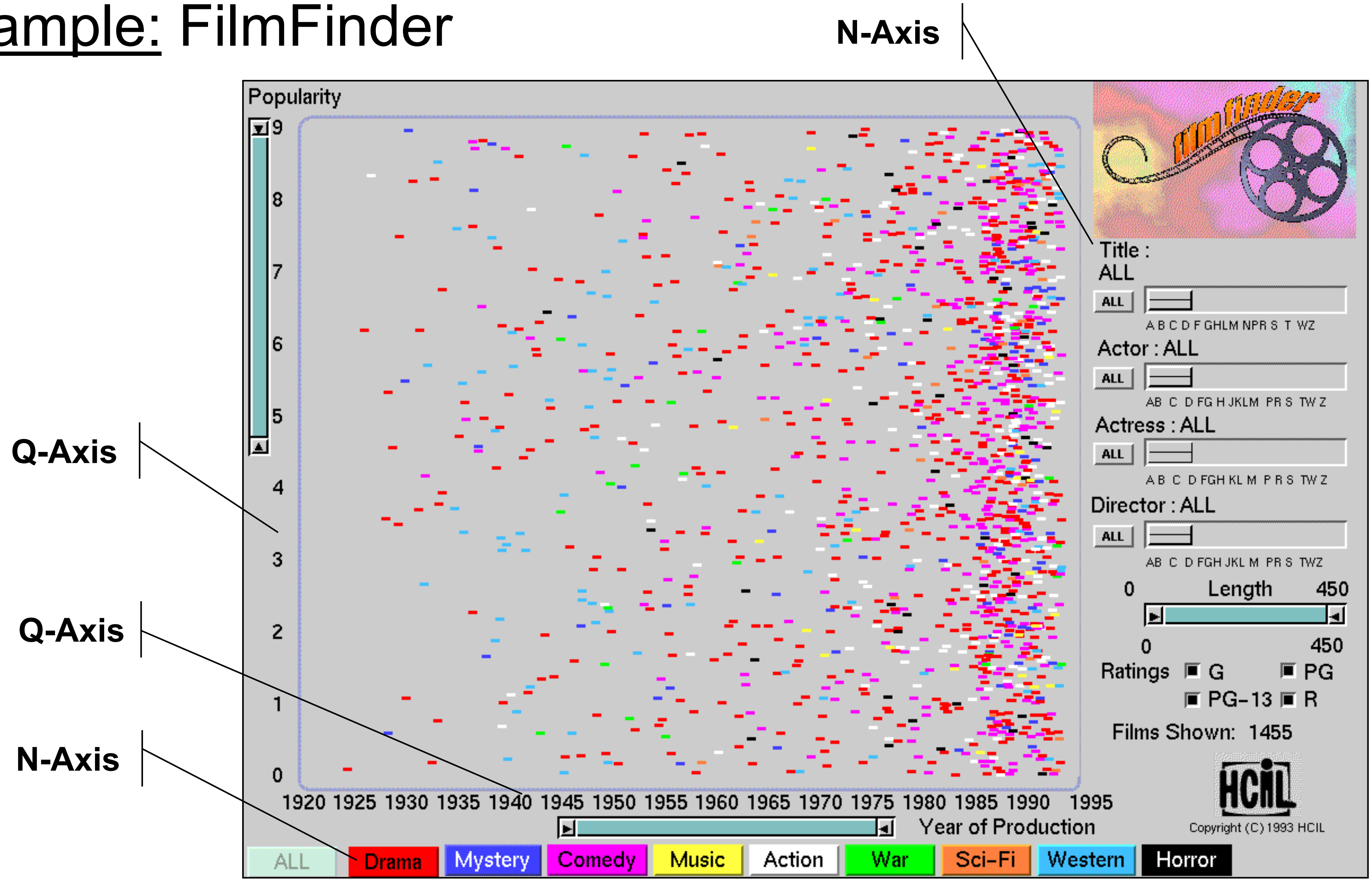
2.2.2 Visual Mapping

- Spatial substrate, i.e., the empty space, can be considered as a kind of container if it has a metric structure
- Description of this structure by different axes
  - **U**nstructured Axes (i.e., no axis)
  - **N**ominal Axes
    - Region can be divided into subregions
  - **O**rdinal Axes
    - Order of this subregion has a meaning
  - **Q**uantitative Axes
    - Region has a metric

[CMS:26ff]

2.2.2 Visual Mapping

- Example: FilmFinder



http://www.cs.umd.edu/hcil/spotfire/

2.2.2 Visual Mapping

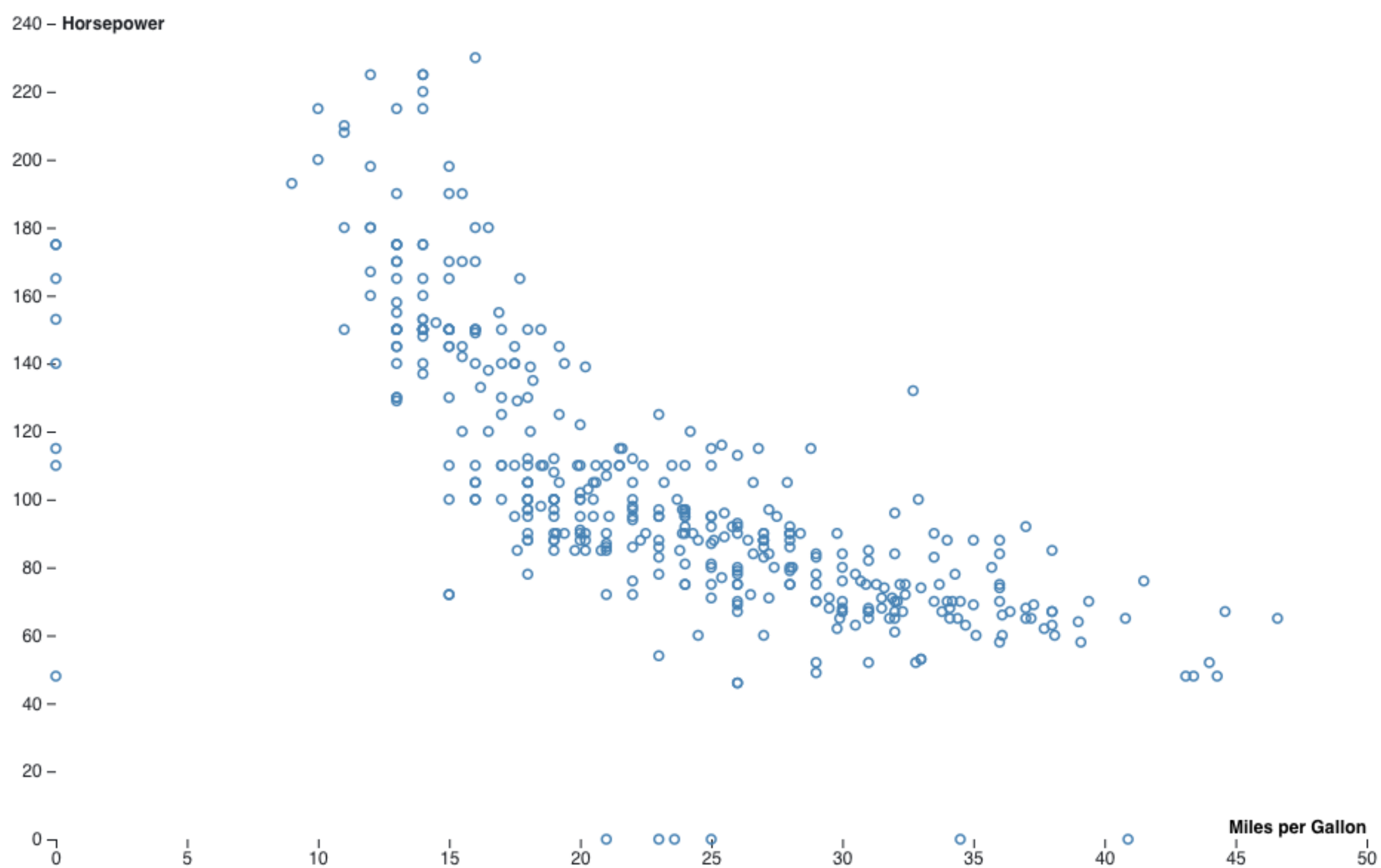
- Techniques to extend the variety of information within the spatial substrate
  - Composition
    - Orthogonal placement of axes
  - Alignment
    - Repetition of an axis at a different position in space
  - Folding
    - Continuation of an axis in an orthogonal dimension
  - Recursion
    - Repeated subdivision of space
  - Overloading
    - Reuse of the same space for the same data table

- <u>Composition:</u> Example

Scatter Plots

Orthogonal placement of axes!

Good until 3 dimensions. For more than 3 dimensions, it can be difficult to perceive.



https://observablehq.com/@d3/brushable-scatterplot

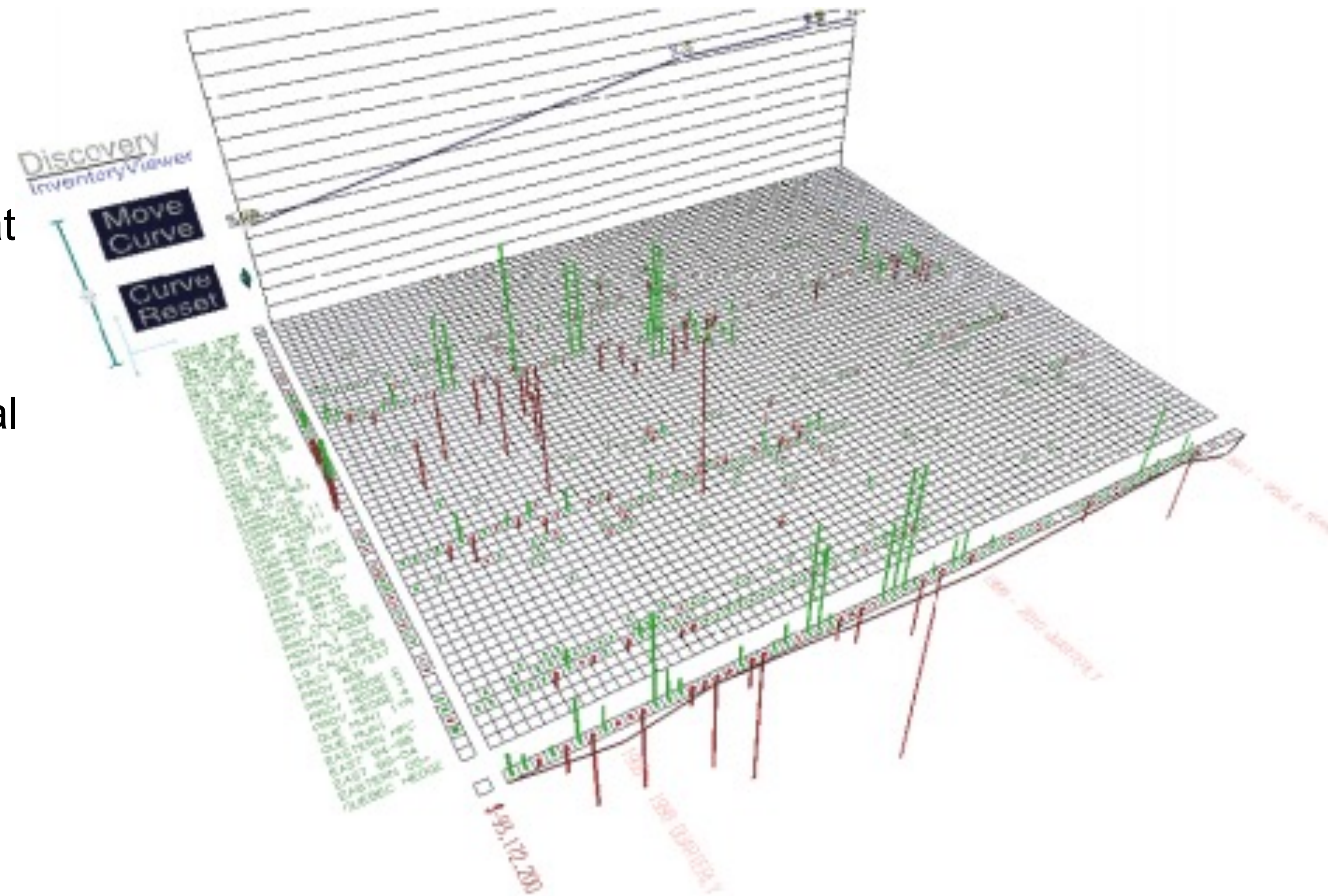
2.2.2 Visual Mapping

- Alignment: Example

Visible Decisions

Repetition of an axis at a different position in space!

Alignment of two visual structures at a common x-axis (time) resp. z-axis (performance).



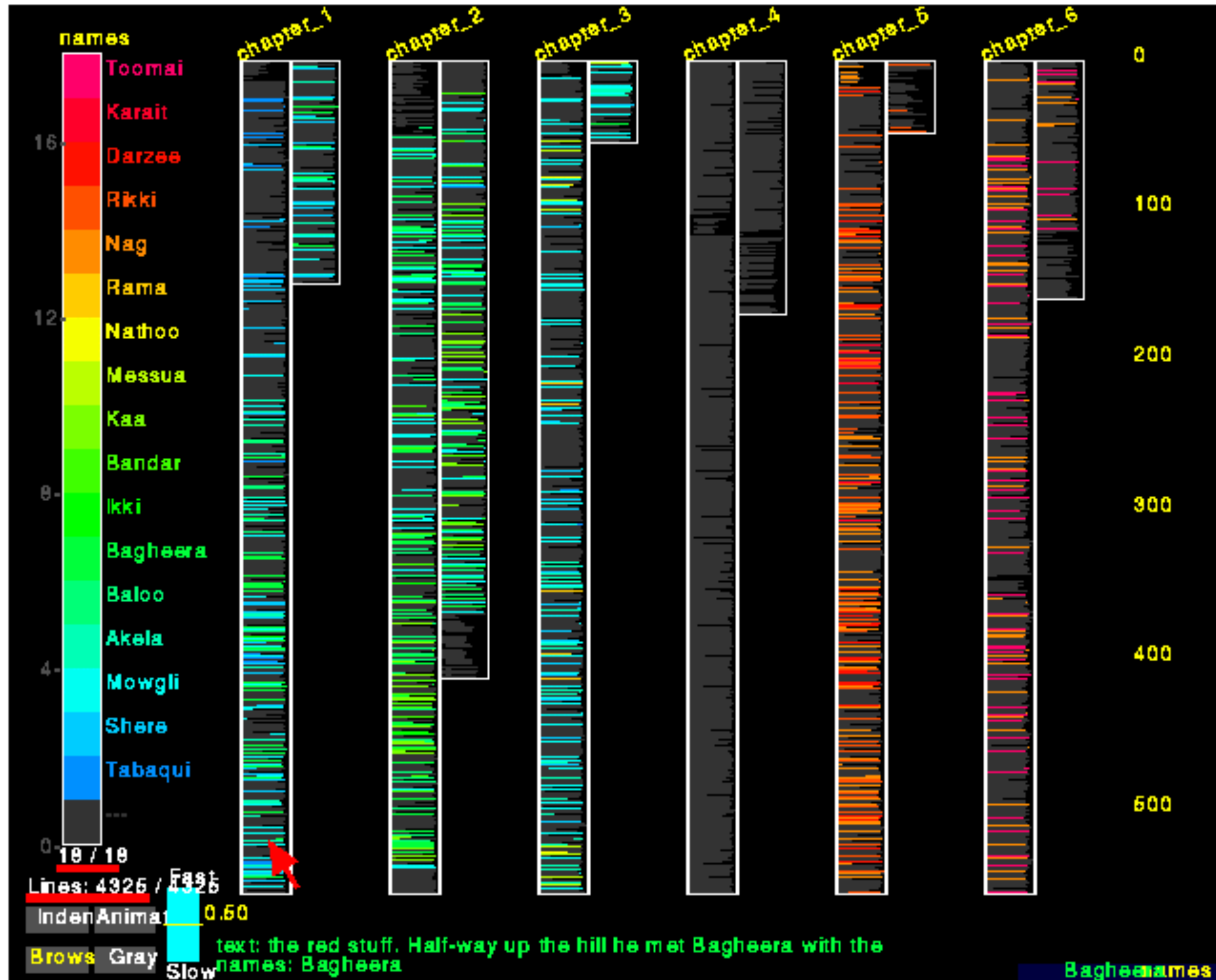
2.2.2 Visual Mapping

- Folding: Example

SeeSoft

Continuation of an axis in an orthogonal dimension!

Axes represent software modules and lines with one pixel height represent single code lines. If axes (O-axes) do not fit in the window then they are wrapped.



[Eick et al. SeeSoft -A Tool for Visualizing Line Oriented Software Statistics. 1992]

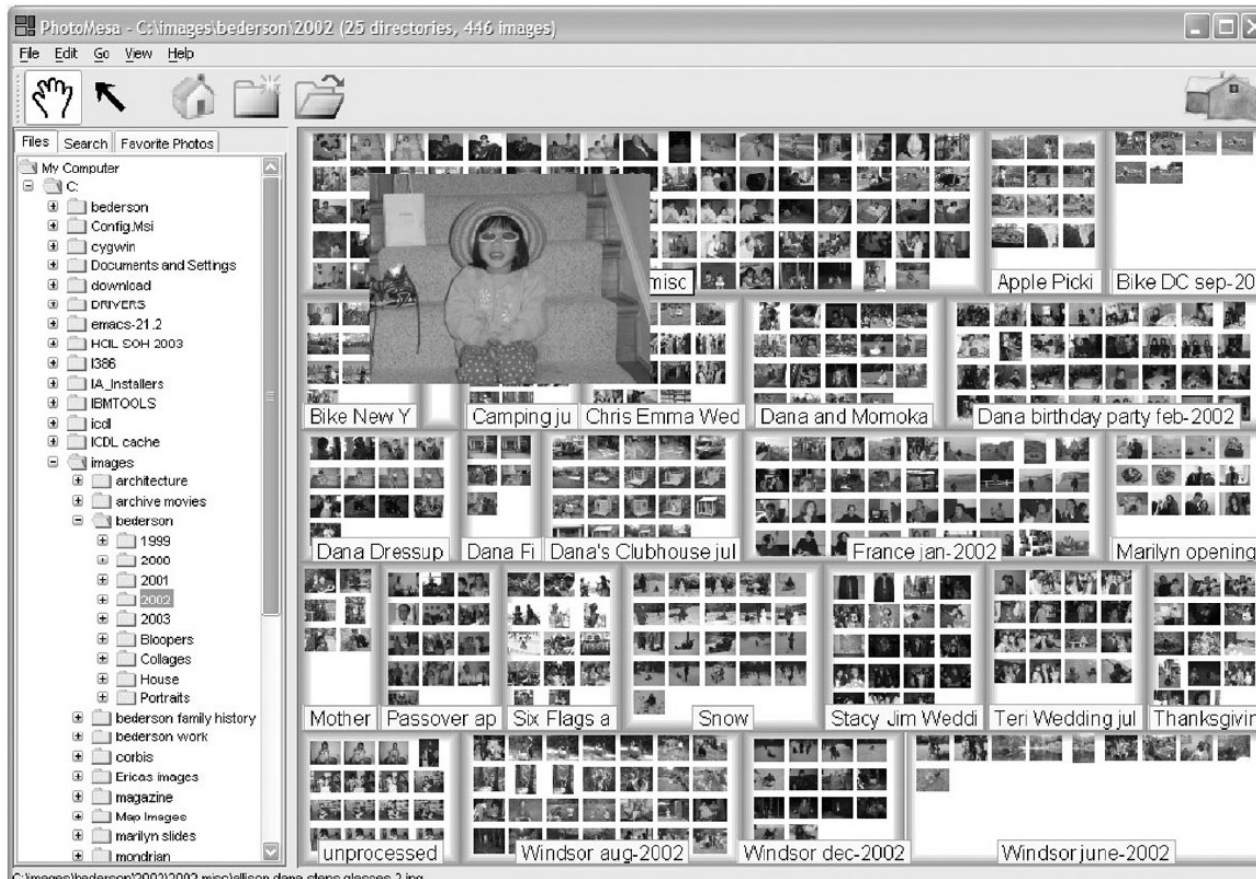
2.2.2 Visual Mapping

- Recursion: Example

Jazz

Repeated subdivision of space!

Recursive space of directories and files. Directories are represented by rectangles and files by small areas that include their content. Navigation occurs by zooming.



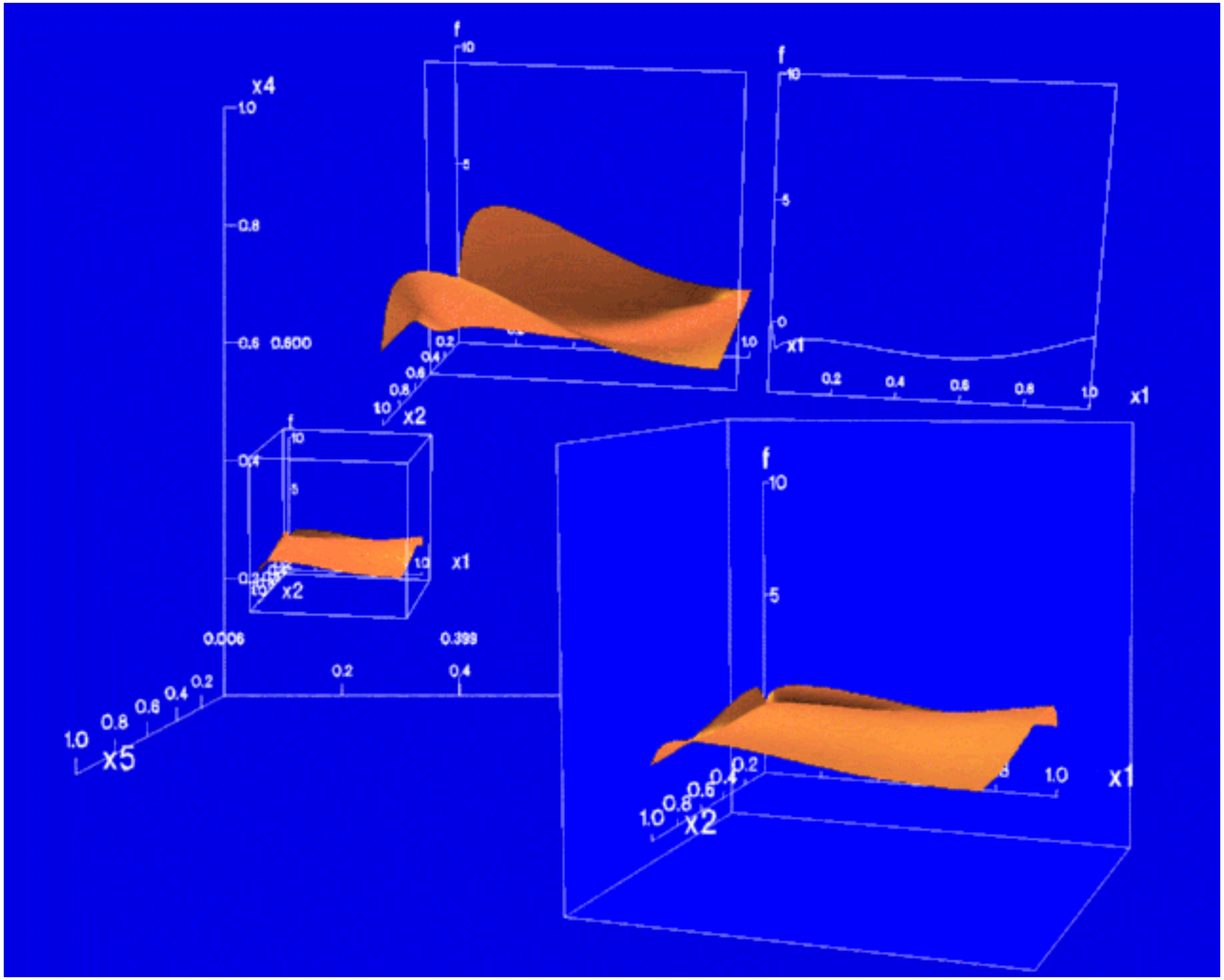
https://doi.org/10.1145/502348.502359

- <u>Overloading:</u> Example

Worlds within Worlds

Reuse of the same space for the same data table!

Meaning of a coordinate system is defined by its placement within another coordinate system.



http://graphics.cs.columbia.edu/projects/AutoVisual/AutoVisual.html

2.2.2 Visual Mapping

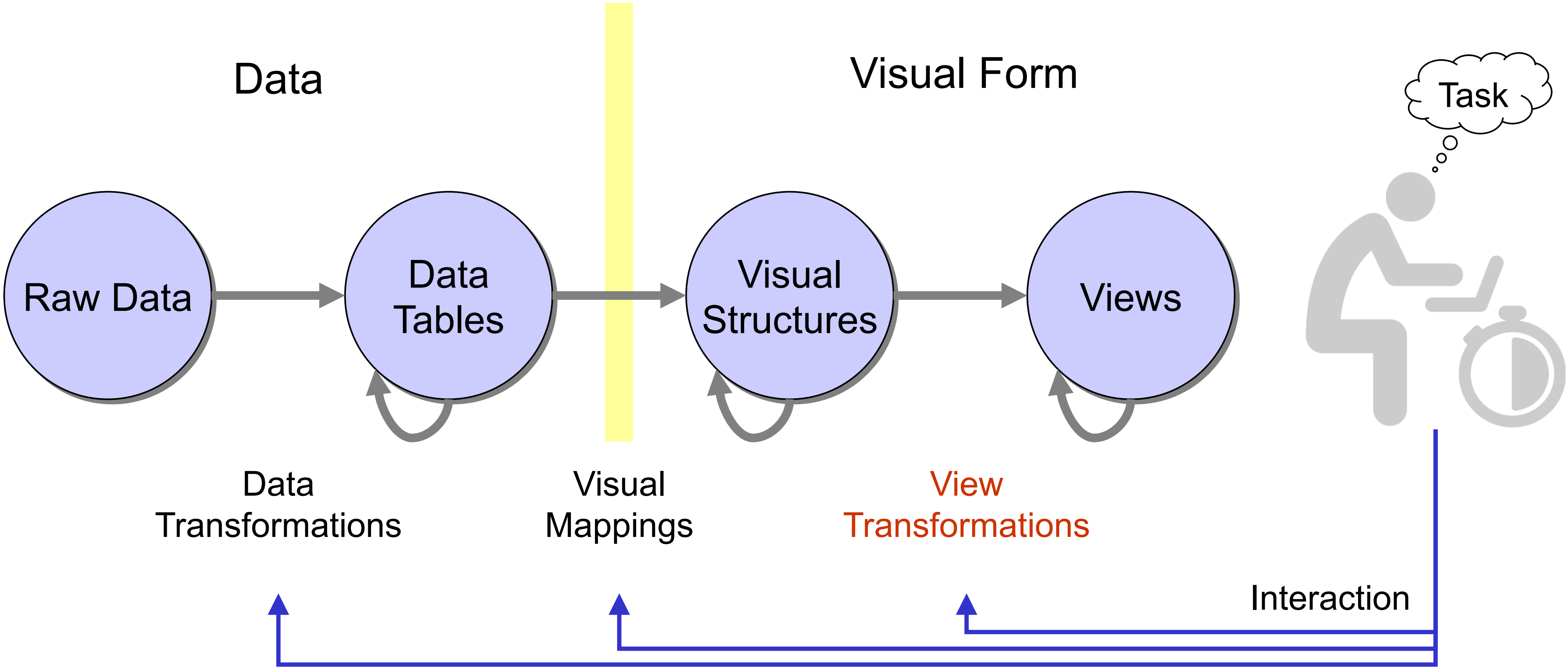
- We will discuss various visual mappings for different data types and dimensions in Chapters 4-6
- Remember that a visual representation of the input data should not go along with avoidable additional cognitive load for the user
- So far, we only saw more or less „static“ visual representations
- Regarding our definitions in Chapter 1, we cannot speak about „visualizations“ in narrower sense

2.2.2 Visual Mapping

- Why? Remember our definition
  - „The use of computer-supported, interactive, visual representations of abstract data to amplify cognition“ [CMS:7]
- What is missing?
  - **Interaction!**
- This way to formulate our definition has several important reasons. So far, our purely static visual structures have disadvantages …

2.2.3 View Transformations

- Reminder



2.2.3 View Transformations

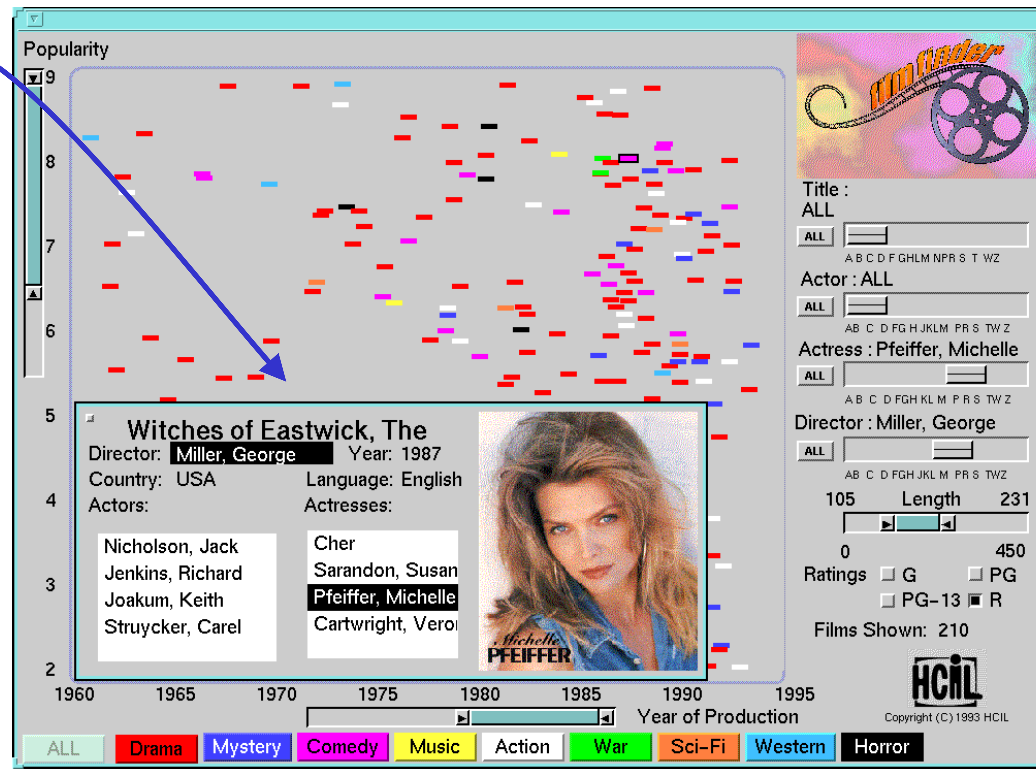
- The visual structure alone leads to problems
  - Scalability
  - Static representations can only show few data aspects
  - Where is the interesting part of the visual structure?
  - How can I find this interesting part?

    → Focusing

    → Overview must be preserved
- View transformations (and **interaction** in general) support users to modify and extend visual structures, i.e., they make static representations to „real“ visualizations

2.2.3 View Transformations

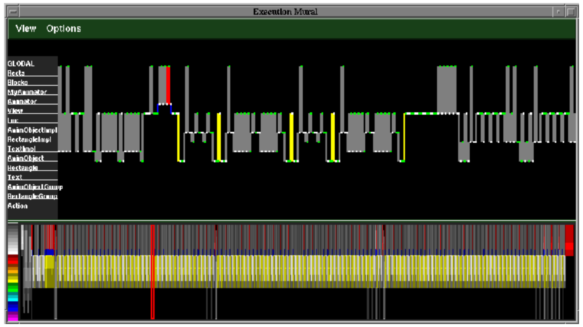
- View transformations will be broadly discussed in Chapter 3. Here, we will only list some examples
- Location Probes
  - Pop-up windows
  - Brushing
  - Magic Lenses
  - …

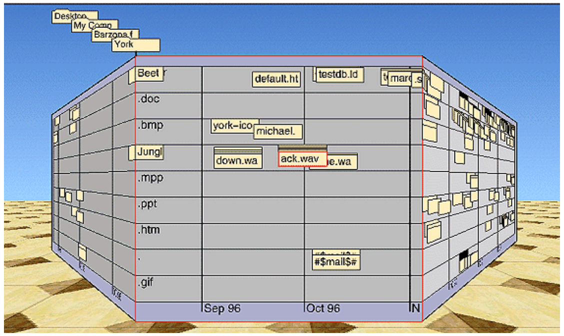


http://www.cs.umd.edu/hcil/spotfire/

2.2.3 View Transformations

- Viewpoint Controls
  - Use affine transformation to zoom, cut or move the visual angle
  - Overview + Detail
    - Information Mural, …

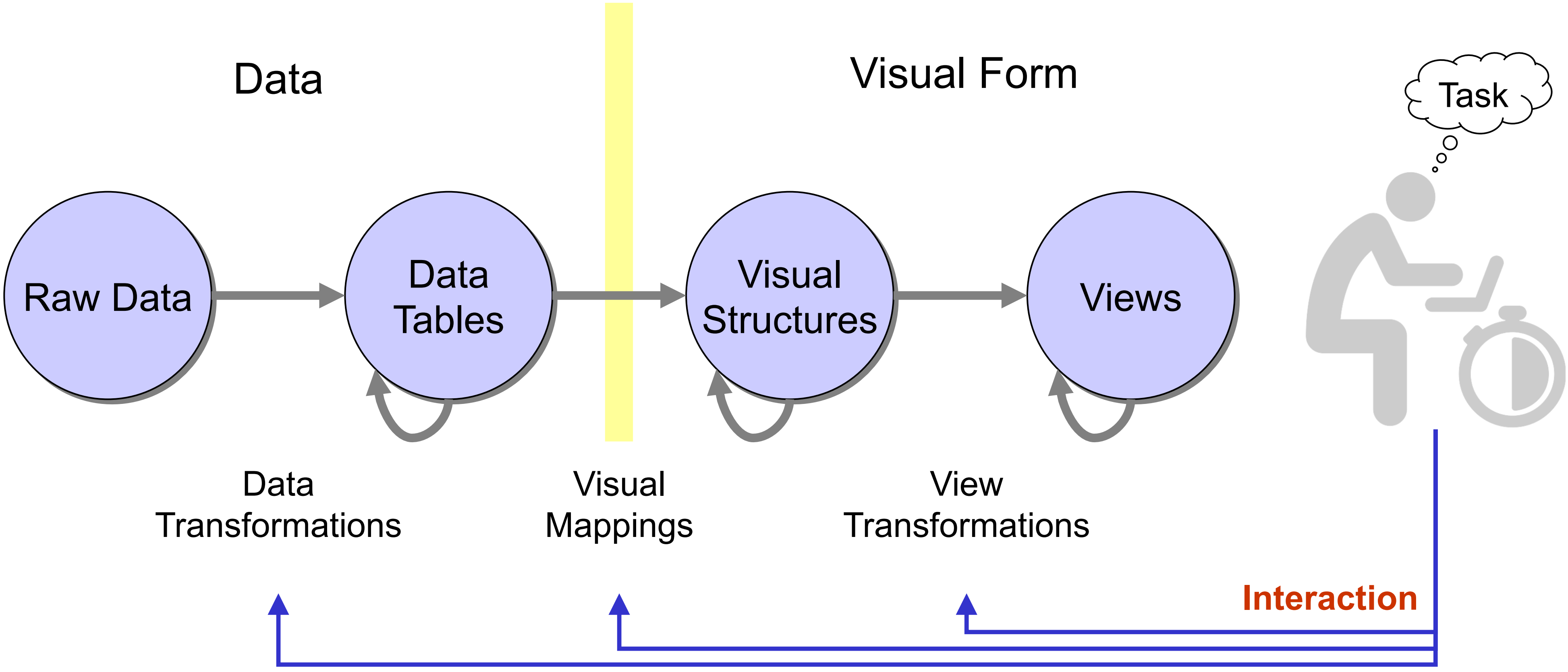




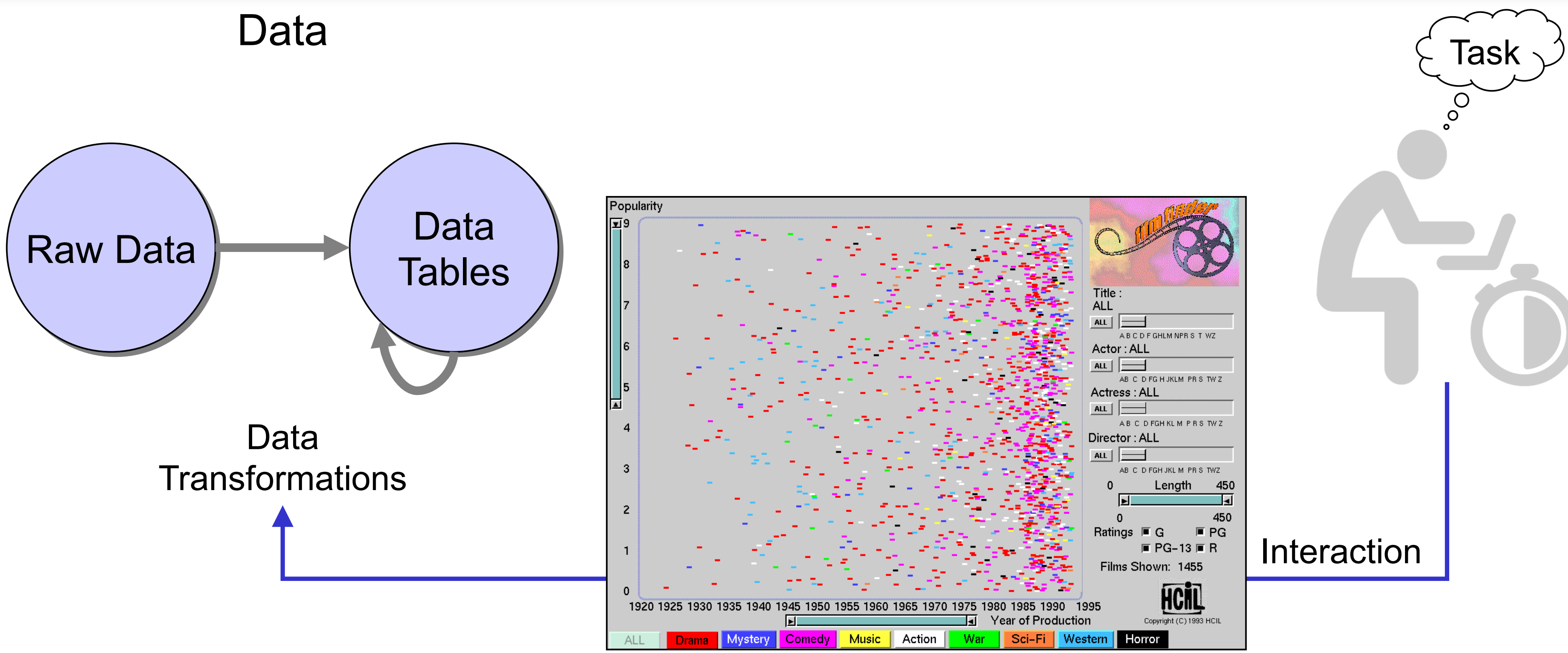
- Distorsions
  - Modify visual structures
  - Focus + Context
    - Perspective Wall, Bifocal Lenses, Hyperbolic Trees, …

2.2.4 Interaction

- Reminder

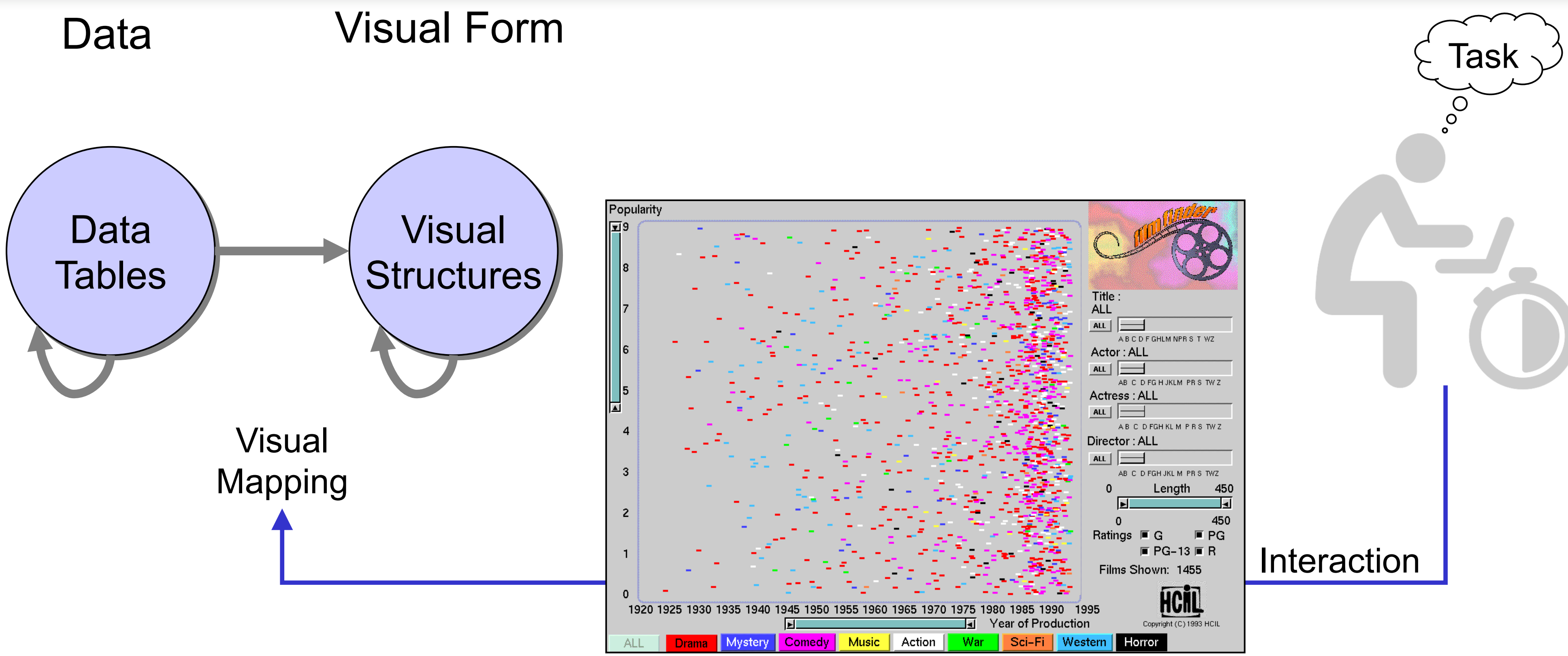


2.2.4 Interaction



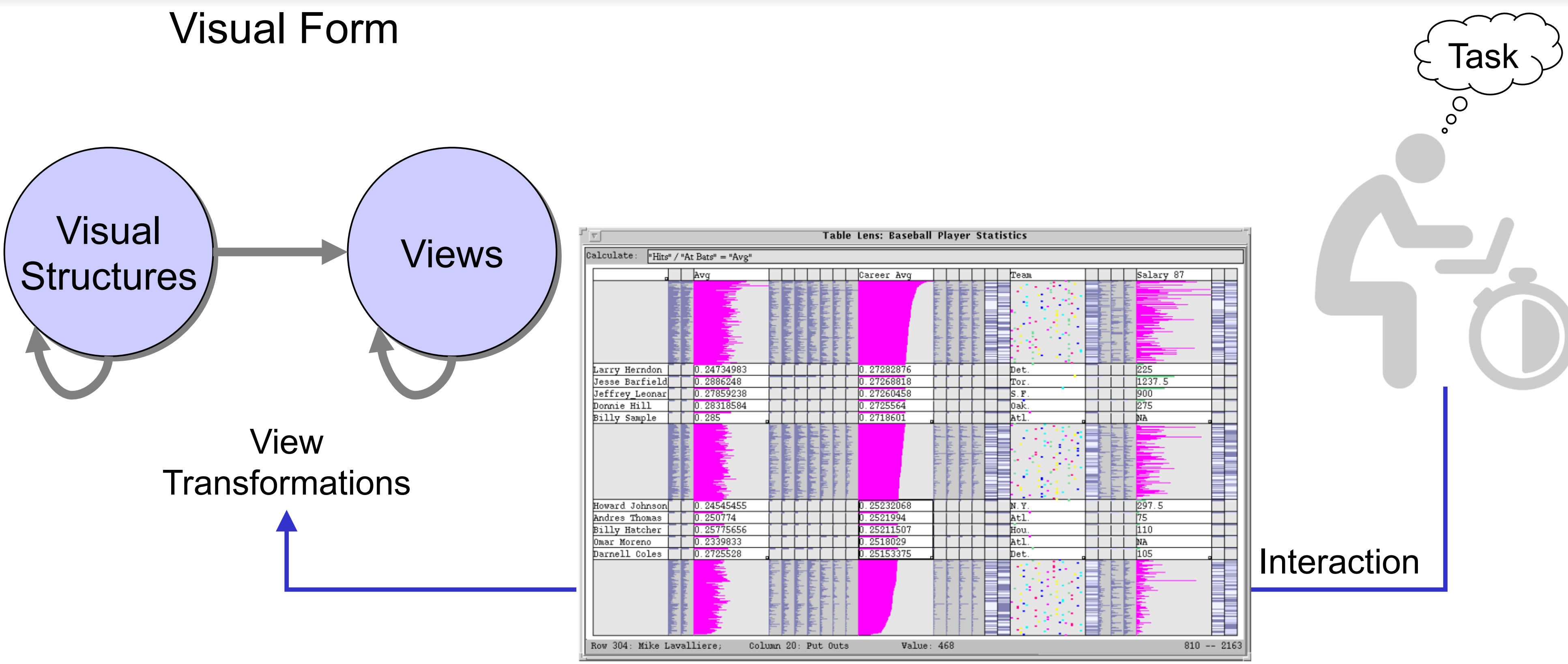
- <u>Example:</u> FilmFinder
- Sliders filter different cases. Only selected cases are shown in the plot

2.2.4 Interaction



- <u>Example:</u> FilmFinder
- Here, we have an integrated solution in which the user can change the labels from *Popularity* to *Rating* by mouse-click on the y-axis

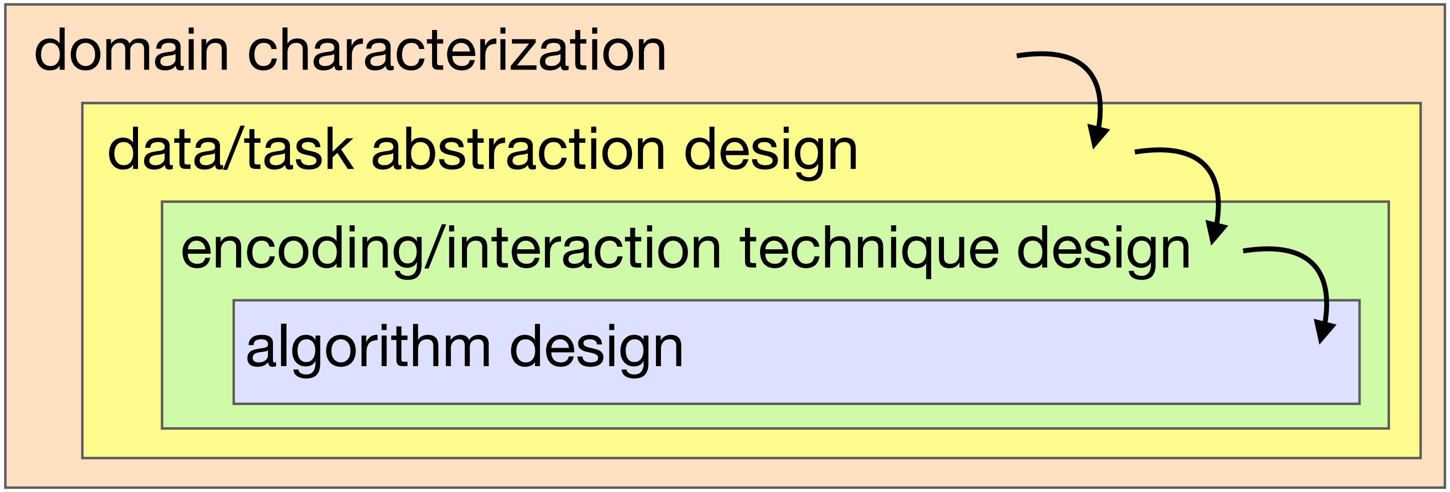
2.2.4 Interaction



- Example: Table Lens
- Location Probes, Viewpoint Controls and distortion techniques are integrated in the GUI. In the screenshot, two areas (bundles of rows) are focused

2.3 Munzner's Nested Model

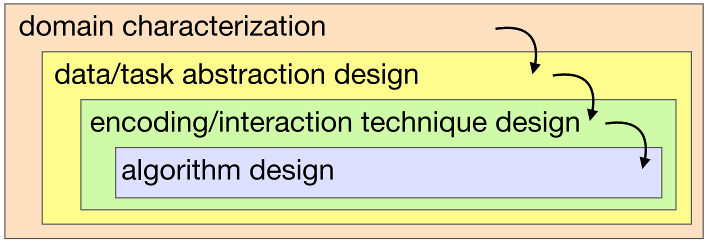
- The InfoVis Reference Model has the drawback that it does **not** directly consider domain or user properties
- Munzner proposed in her textbook a nested model with four cascading levels [MUN:66ff]



- Each inner level receives input from its outer level. Thus, early mistakes propagate further to inner levels

2.3 Munzner's Nested Model

- The Reference Model covers the two inner blocks, and the concrete algorithm is used to produce a visual representation (e.g., a bar chart) or to implement a specific interaction technique (e.g., brushing)



- The two outer blocks are discussed now in this section
- Note on Munzner's model
  - Each level can be analyzed separately
  - But in practice, design decisions are **iterative processes**!

2.3.1 Domain Situation

- The *domain situation* covers
  - The group of target users
  - Their domain of interest which comes with its own vocabulary and workflow
  - Their concrete analytical questions
  - Their data sets
- Challenges for visualization designers
  - Must understand the needs of the users and their requirements
    - Users usually cannot directly articulate their needs and tasks that they want to solve with a visualization
    - Asking users typically does not provide the complete picture

[MUN:69f]

2.3.1 Domain Situation

- Challenges for visualization designers (cont.)
  - In consequence, deriving system requirements for a more sophisticated vis tool is hard
  - All those issues will result in a time-consuming, iterative process that may even span over a longer period, but this is well invested time
- Findings and results from this process of understanding the domain situation leads to the next level in the model

→ **Data and Task Abstraction**

(this is to abstract from the specific domain questions/data into a domain-independent vocabulary → helps us to realize common demands and tasks to be solved across domains and/or different vocabularies)

[MUN:69f]

2.3.2 Data Abstraction

- In Sect. 2.2, we discussed data tables and the data transformation step
- Input data sets may also describe more complex data structures, for instance
  - Tables as previously discussed
  - Networks and trees
  - Geo-tagged data (points on maps, …)
  - Temporal data
- Usually, all of them can be depicted by tables
- The attributes/values in a table have specific types

2.3.2 Data Abstraction

- We know the following data types
  - *Nominal* (also: categorical) data
  - *Ordered* data can be subdivided into
    - *Ordinal* data and *quantitative* data
- These data types can occur in all dimensions
  - uni-, bi-, tri-, or multivariate
- Typically, all these data types are mixed in the data tables
- Our data abstraction opens up a huge design space on how to visually encode the data

2.3.3 Task Abstraction

- User tasks are equally important for vis designers as the kind of input data. Why?
  - A vis approach may work well for one task but bad for another one, for exactly the same data set
  - Another reason is to decide if we need to derive new data from the original data (e.g., by computation of the average of a table column). This decision is driven by the concrete user tasks

- We need a vocabulary-independent task description that is general enough in order to reason about similarities/ differences between them

[MUN:43ff]

2.3.3 Task Abstraction

- Example of user task descriptions
  - Epidemiologist
    - "contrast the prognosis of patients who were intubated more than one month after exposure to patients hospitalized within the first week"
  - Biologist
    - "see if the results for the tissue samples treated with LL-37 match up with the ones without this peptide"
- What is meant here from a more abstract level?
  - Simply *comparison*!
  - "contrast" and "match up" → compare values between two groups (easier to understand and to remember)

[MUN:43ff]

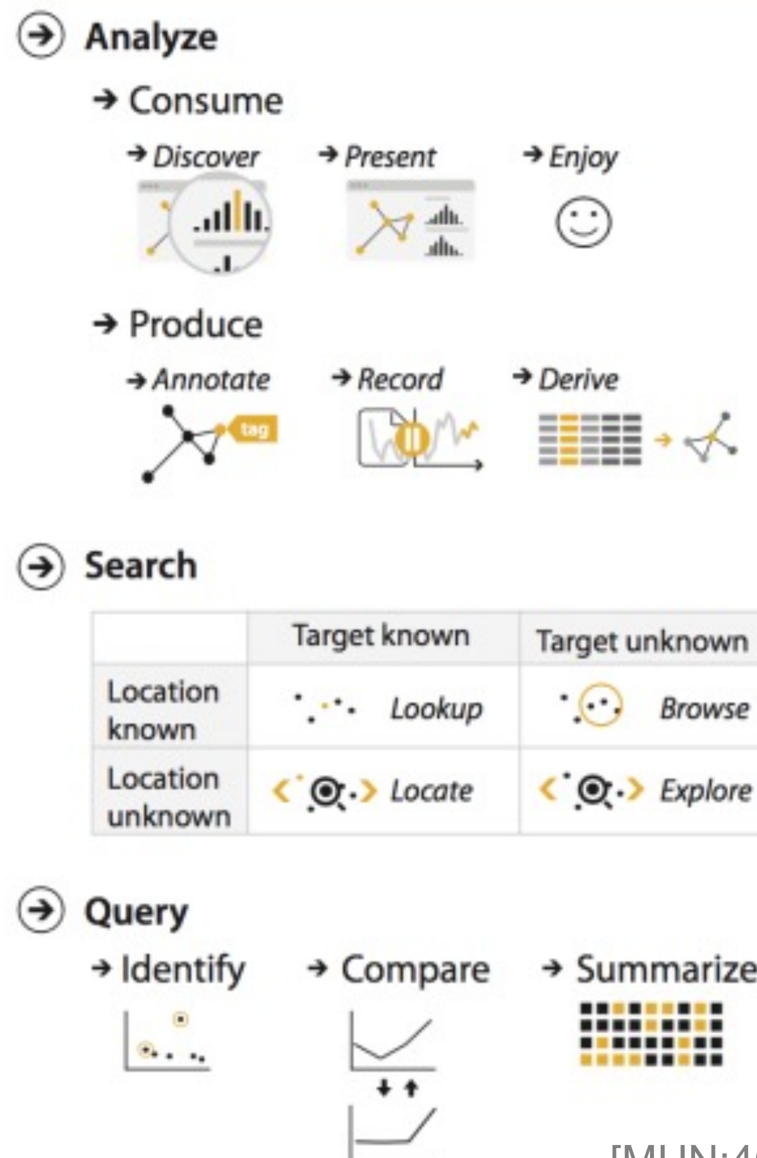
2.3.3 Task Abstraction

- Visualizations can be more general or more specific
  - Specific tools
    - Usually narrow
    - Designer put some effort into the design choices especially for a domain or specific group of data/tasks
    - Thus, the users have less decisions to make when visually analyzing their data as the tool has been specifically designed for them
  - General tools
    - Broad and generally usable for many domains, data, and tasks
    - Flexible, and the users have many possibilities to make choices
    - This flexibility is nice, but also can be overwhelming and may lead to less effective choices

[MUN:43ff]

2.3.3 Task Abstraction

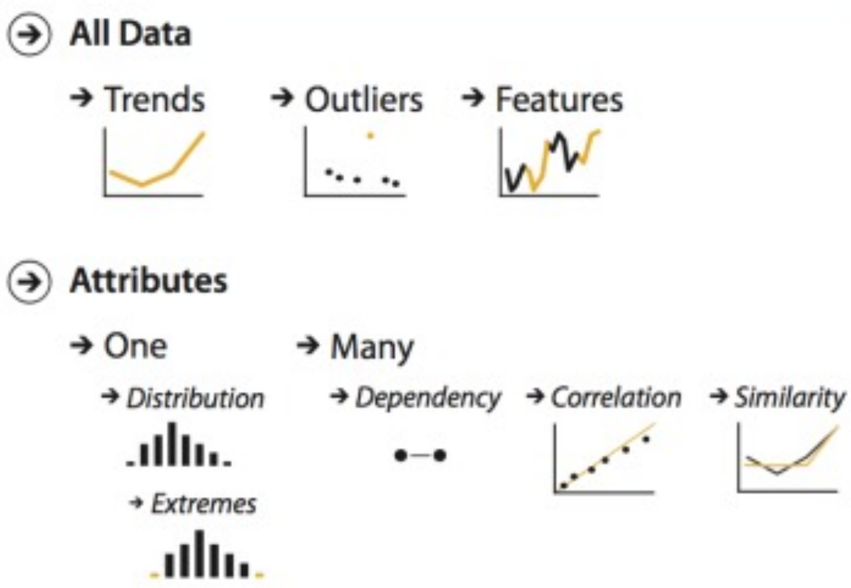
- **Actions** define user goals
  - Analyze
    - Consume existing data sets
      - Discover, Present, Enjoy
    - Produce new data
      - Annotate, Record, Derive
  - Search
    - Lookup, Browse, Locate, Explore
  - Query
    - Identify, Compare, Summarize



[MUN:46]

2.3.3 Task Abstraction

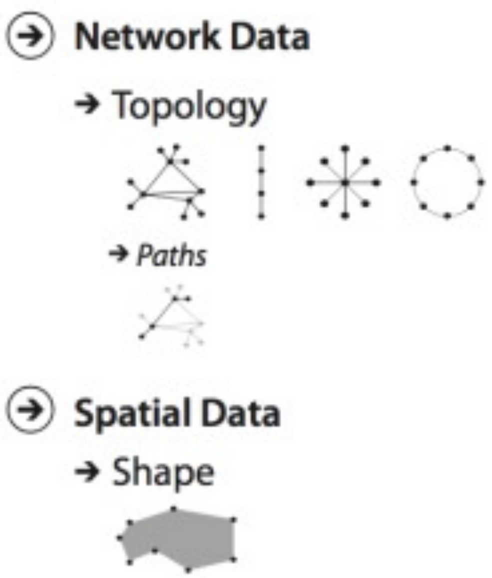
- **Targets** that are somehow of interest to a user
  - Targets for all kinds of data
    - Trends (= higher-level patterns)
    - Outliers (= anomalies)
    - Features (= low-level patterns)
  - Targets are attributes
    - Find one specific value
      - Distribution, Extremes
    - Consider values of many attributes
      - Dependency, Correlation, Similarity



[MUN:46]

2.3.3 Task Abstraction

- **Targets** that are somehow of interest to a user (cont.)
  - Network data
    - Topology, Paths
  - Spatial data such as maps
    - Shape

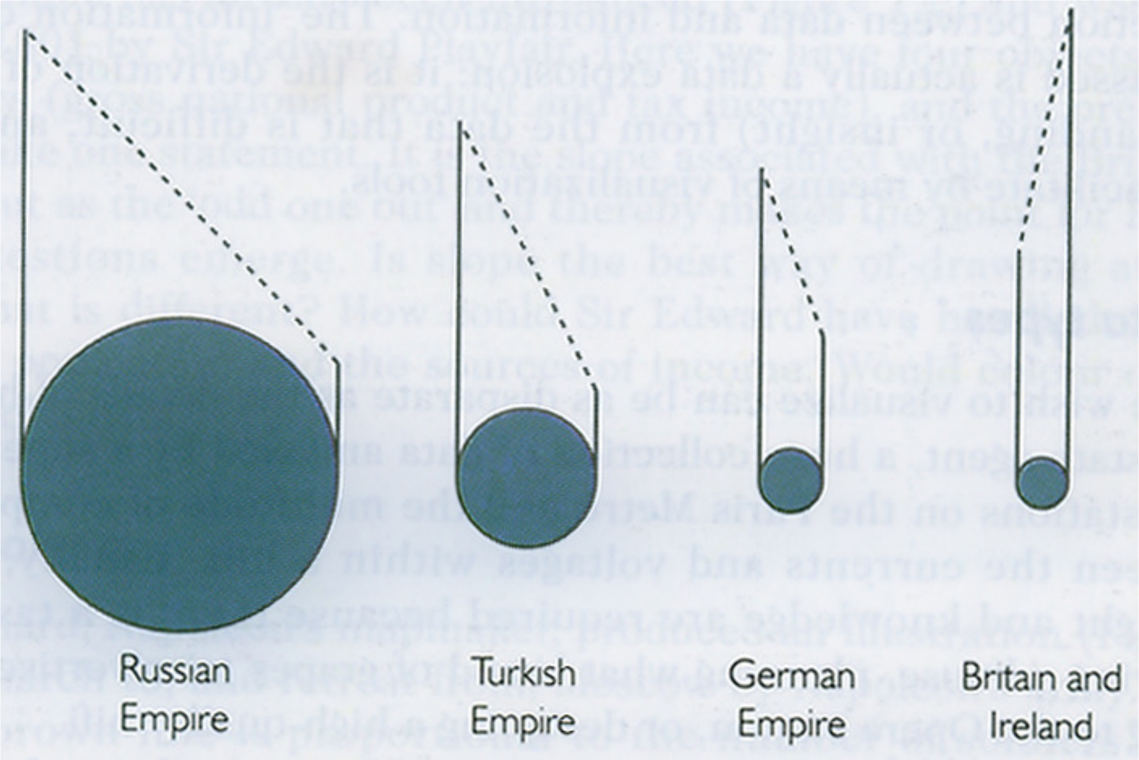


- Now, the challenge is **how** to visually represent data in context of the performed data and task abstraction by the vis designer

[MUN:46]

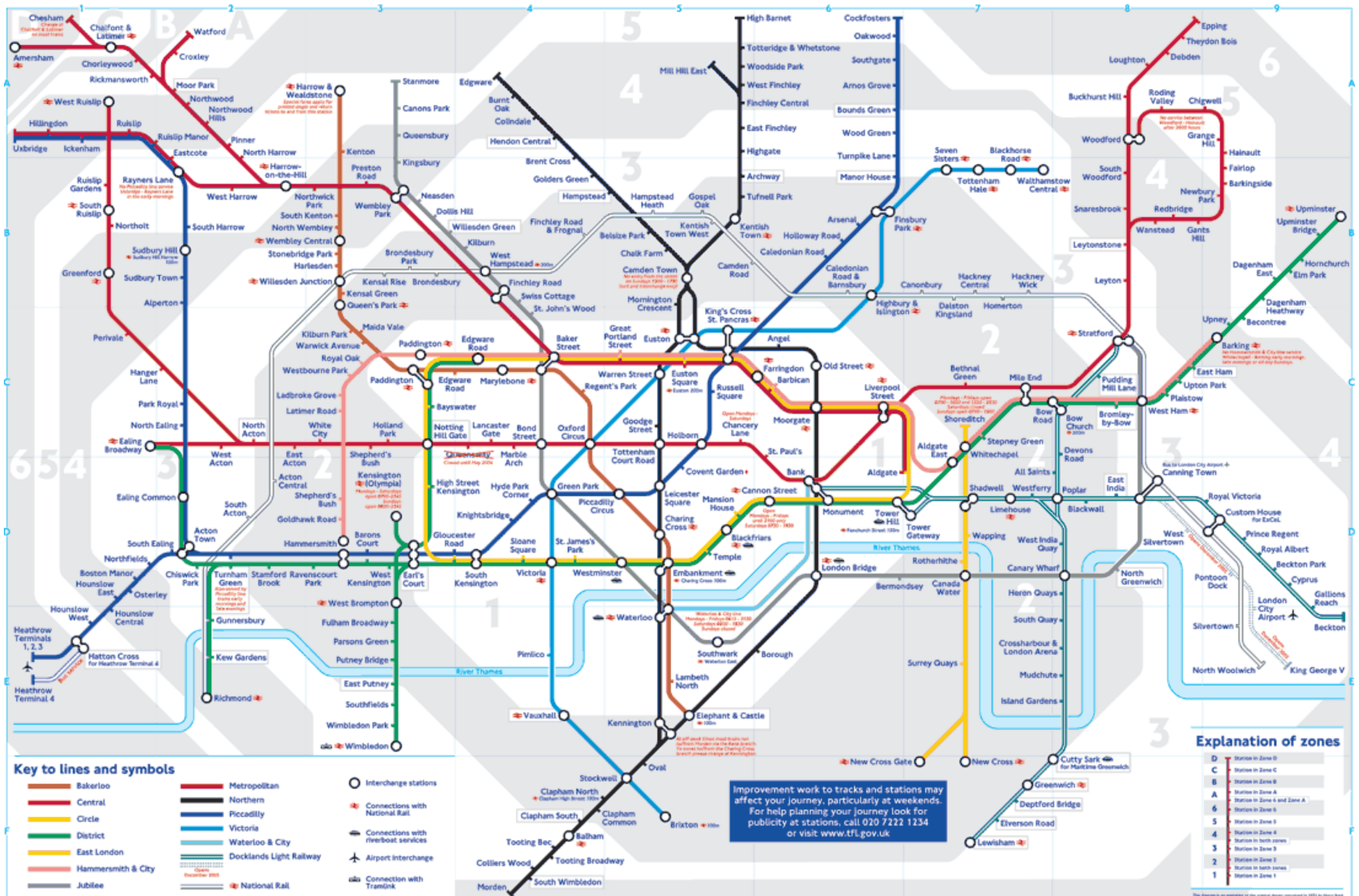
2.7 Visual Representation

- This challenge will accompany us in this course and forms a creative design step (i.e., no straightforward act of identification)
- In Chapter 1, we saw already several examples, e.g., the *slope* of Playfair's Circles represent *nominal data* (positive/negative)



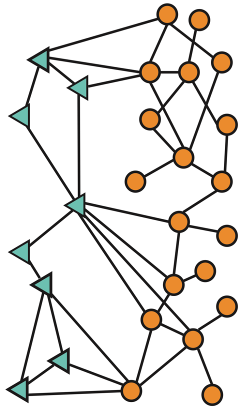
2.7 Visual Representation

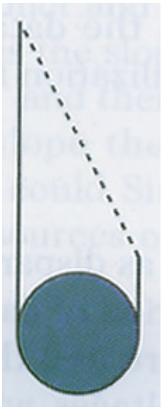
- *Color* is used in Beck‘s Map for representing different railway lines (*nominal data*). Finding a *path* might be a *target* here!



2.7.1 Encoding Types

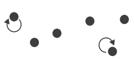
- Visual encoding
  - We map attribute values (or nD data objects) onto visual features (or more complex visual structures)
  - These features and/or structures are arranged on the screen
- Encoding types
  - Size, Length, Angle, Curvature, … 
    - Compare country size, GNP, or degree of slope at Playfair‘s Circles
  - Shape + ● ■ ▲
    - Nominal data only
    - Compare metro lines in Beck’s map or different node types in a network

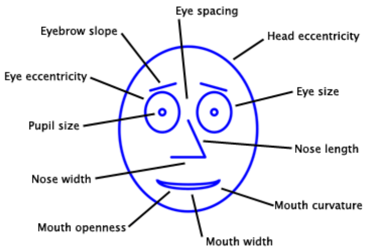


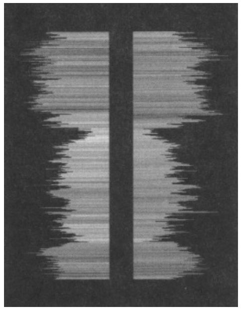


2.7.1 Encoding Types

- Encoding types (cont.)
  - Color 

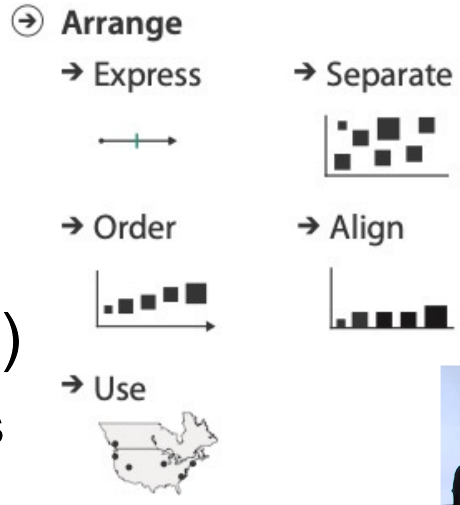
    - Hue (nominal only), saturation, luminance
    - Our sine wave 
  - Motion 
    - Common fate 
  - Icons (for nD data objects)
    - Chernoff Faces
  - Complex patterns (for nD data objects)
    - Symmetry comparison



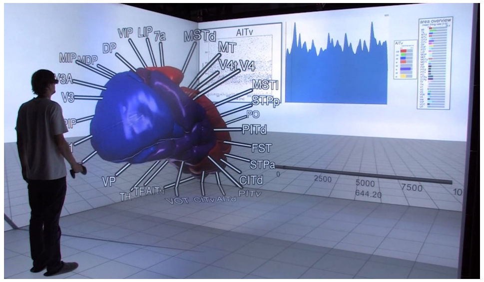


2.7.1 Encoding Types

- Encoding types (cont.)
  - Sound
    - Sound can be data to be visualized
    - Sound can be used for *sonification* of non-auditive data
      - status tone, Windows-sound (open windows, new email, …), etc.
  - Verbalization

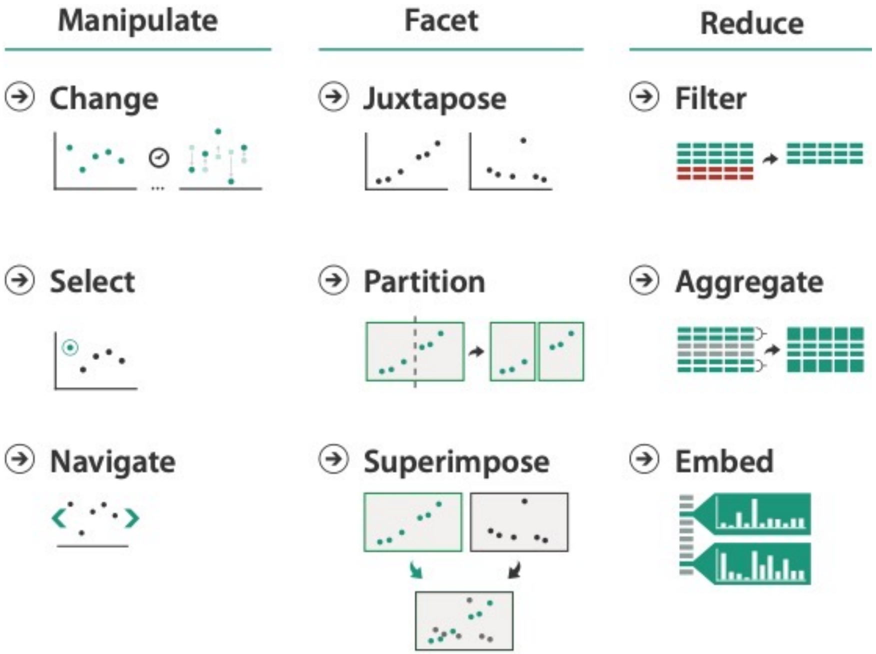


- Arrangement in space
  - Express (value/s)
  - Separate, Order, Align (regions)
    - cf. Sect. 2.2.2 on visual mappings
    - virtual reality, immersive analytics, …
  - Use (of a given spatial layout)



2.7.2 Further Aspects

- Further aspects support the choice to
  - Manipulate the view
    - Change, Select, Navigate
  - Facet data between views
    - Juxtapose, Partition, Superimpose
  - Reduce the data shown in the display
    - Filter (DQs), Aggregate, Embed (F&C)



[MUN:58]