VisInfoVis: a Case Study

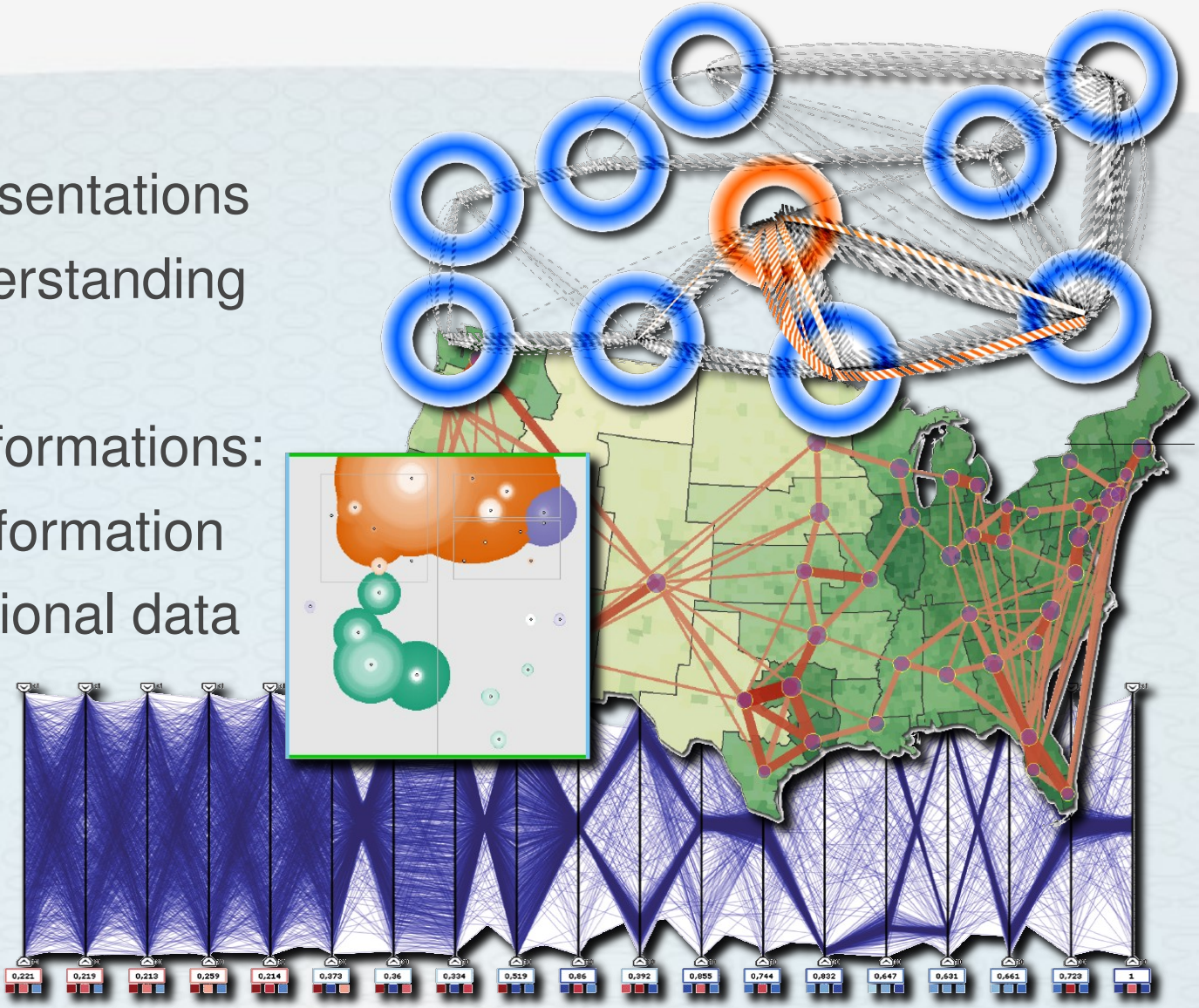
J. Tierny <jtierny@sci.utah.edu>

Overview

- VisInfoVis:
  - Relationship between the two
- Case Study: Topology of Level Sets
  - Introduction to Reeb Graphs
  - Examples of visualization applications
  - InfoVis problem formulation
  - Examples of solutions

What's Information Visualization?

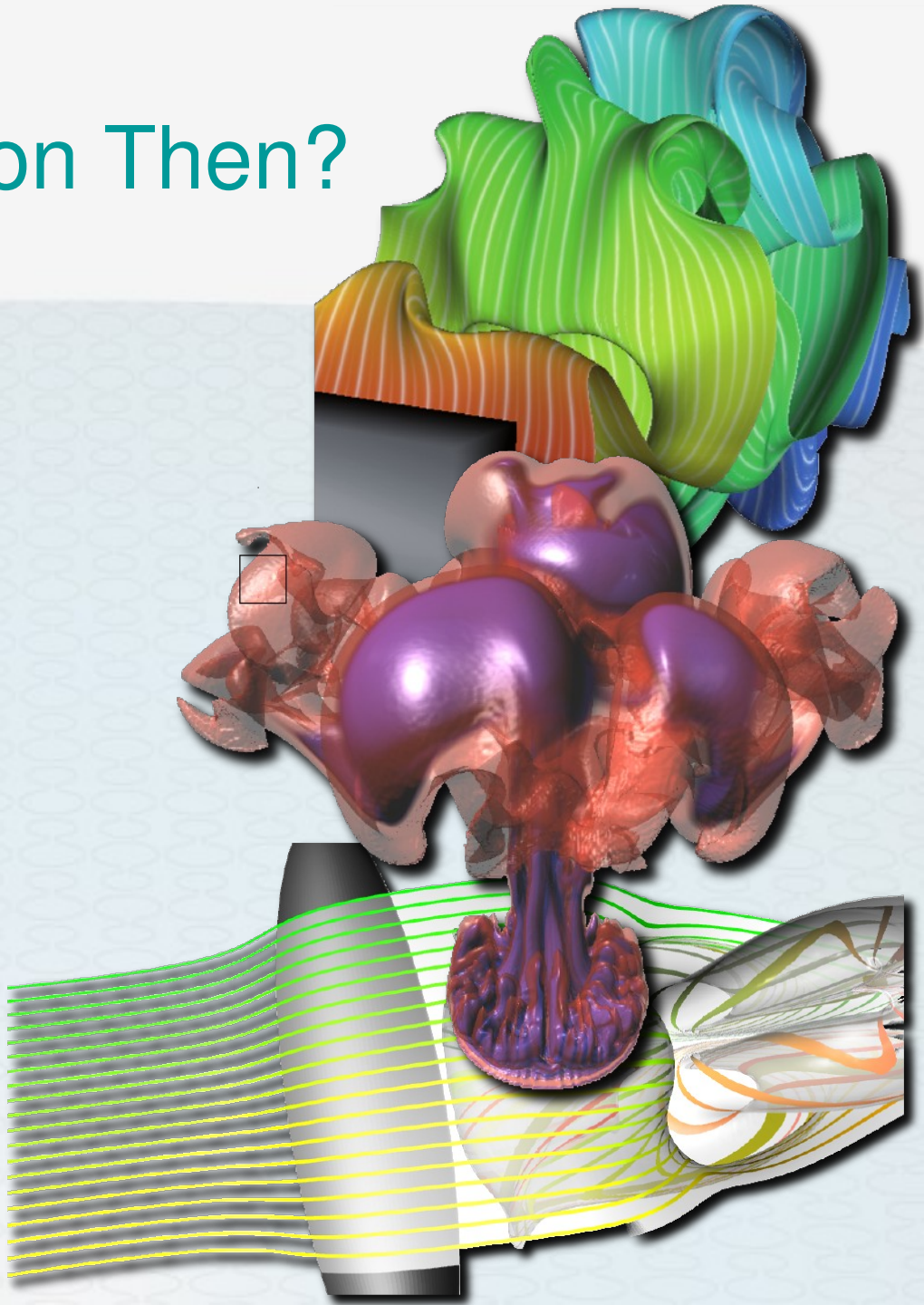
- Science of:
  - **Visual** representations
  - Intuitive understanding
- For **abstract** informations:
  - High-level information
  - High-dimensional data



VIS09

What's Visualization Then?

- Science of:
  - **Visual** representations
  - Intuitive understanding
- For *scientific* data:
  - Spatio-temporal data
  - Scientific simulations & measurements



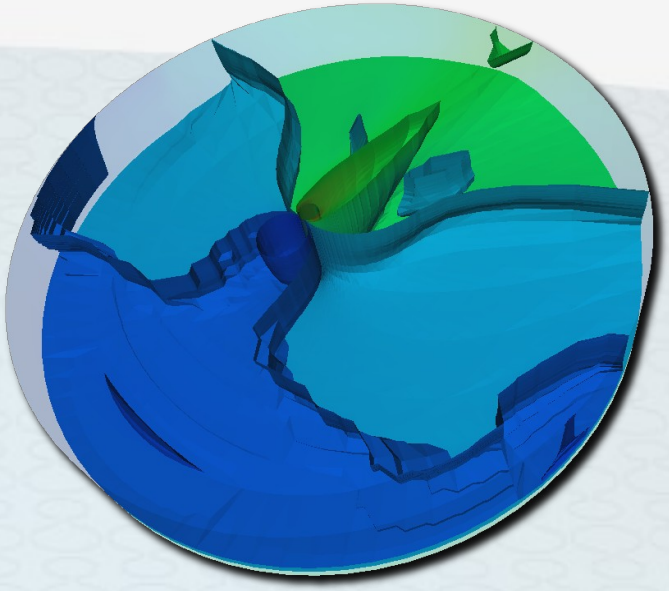
Are the Two any Related?

- A priori:
  - No
- InfoVis:
  - Abstract input
  - General techniques
- Vis:
  - Specific input (spatio-temporal)
  - Specialized techniques
- However, IEEE Vis and IEEE InfoVis: same event

Case Study: Topology of Level Sets

- Overview:
  - Need for abstract representations
  - Introduction to the Reeb graph
  - Examples of application
  - InfoVis problem formulation
  - Some solutions

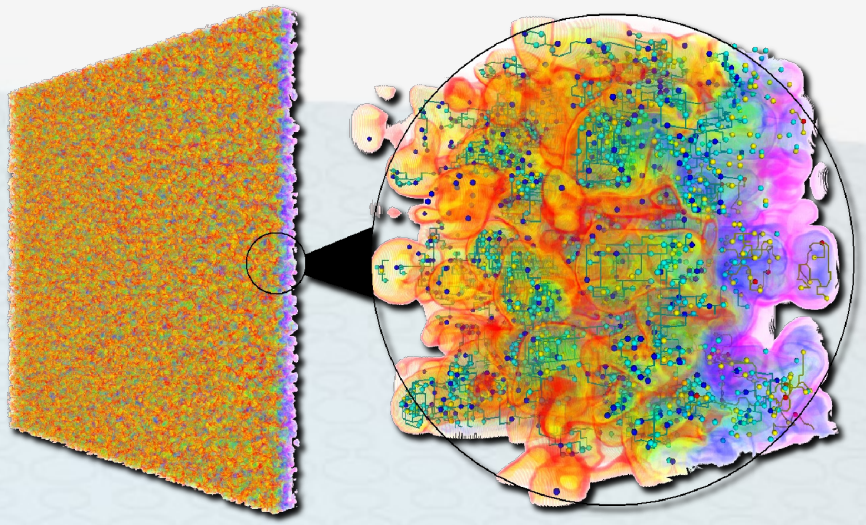
Level Sets:
Needs for Abstract Representations



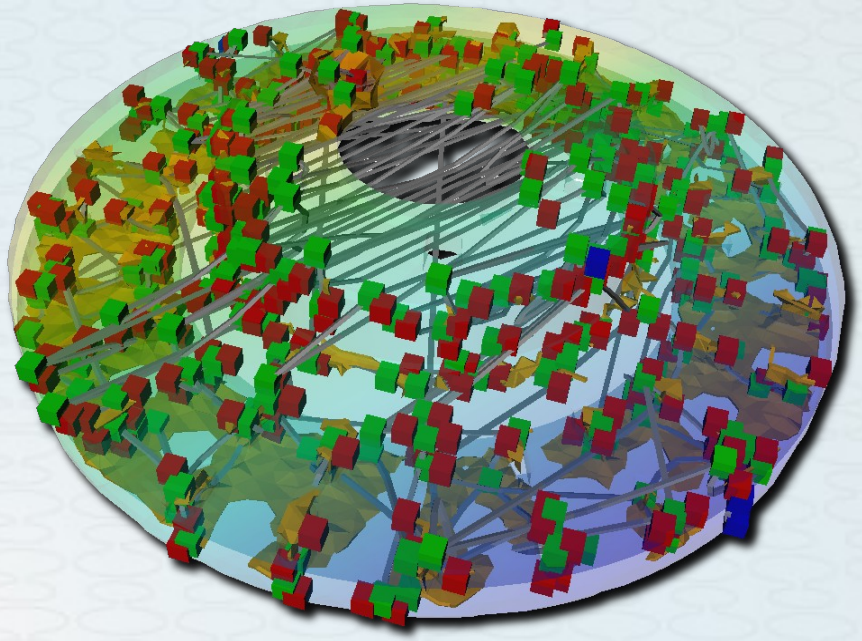
- Iso-surfaces are important!
  - Volume Rendering
  - Data Analysis
  - Geometry Processing
- **But..**
- Finding *interesting* isovalues can be difficult:
  - May require to restart the whole analysis pipeline
- Need for abstractions independent from a given isovalue

Scalar Field Topology Abstractions

- Morse-Smale Complex:
  - Segment the domain in regions of homogeneous gradient



- Reeb Graph:
  - Contract the connected components of level sets in a 1-dimension skeleton
  - Global abstraction of the **topology of level sets**



Reeb Graphs in a Nutshell

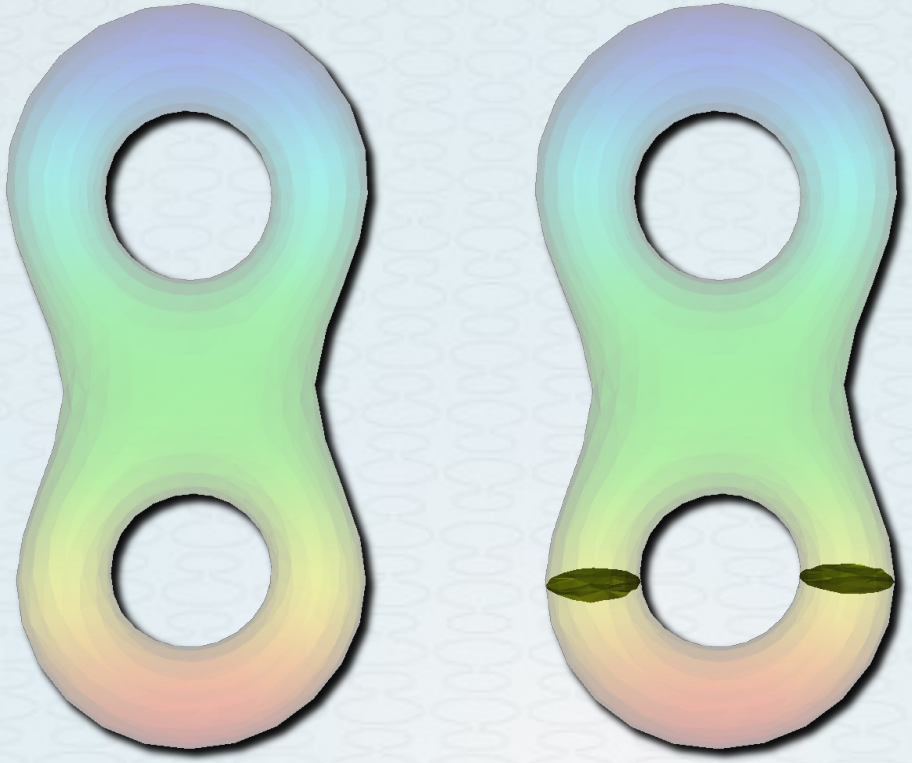
- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Reeb Graphs in a Nutshell

- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Reeb Graphs in a Nutshell

- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f



Reeb Graphs in a Nutshell

- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Reeb Graphs in a Nutshell

- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Reeb Graphs in a Nutshell

- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Reeb Graphs in a Nutshell

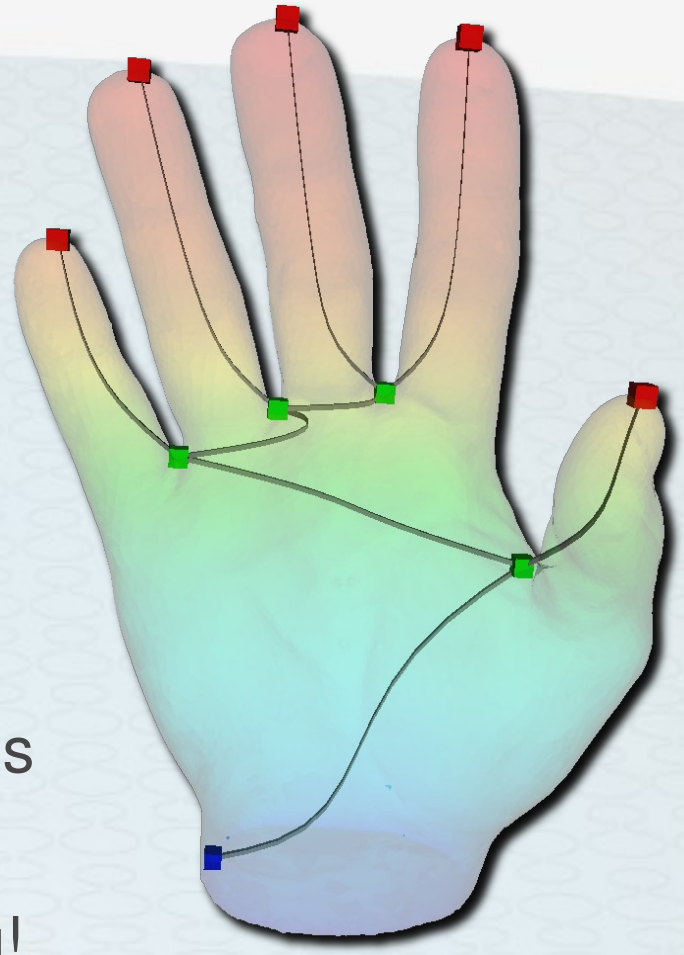
- Given a Morse function f on a manifold M:
  - The **Reeb Graph RG(f)** [Reeb Ac. Scie. 1946] is:
  - Continuous *contour retract* of M under f

Construction Algorithms

- In arbitrary dimensions:
  - Streaming algorithm [Pascucci SIGGRAPH07]
  - Output sensitive algorithm [Doraiswamy SOAC08]
  - Loop-free result: [Carr SODA99] (**optimal**)
- Specialized for fixed dimensions:
  - 2D: [ColeMcLaughlin SOCG03] (**optimal**)
  - 3D: [Tierny VIS09]

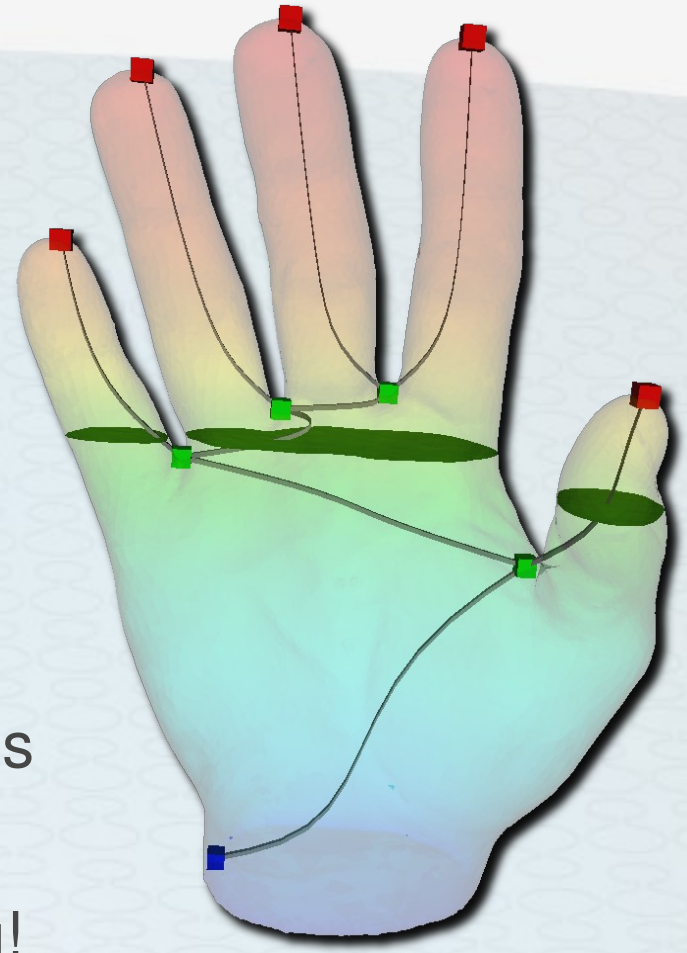
Fast IsoSurface Traversal

- Output data-structure:
  - Collection of arcs:
    - List of regular vertices
- Hum...
  - **Optimal** isosurface seed sets!
  - Arcs stored in interval trees
    - Segments stored in interval trees
- Indexing of level sets for fast querying!



Fast IsoSurface Traversal

- Output data-structure:
  - Collection of arcs:
    - List of regular vertices
- Hum...
  - **Optimal** isosurface seed sets!
  - Arcs stored in interval trees
    - Segments stored in interval trees
- Indexing of level sets for fast querying!



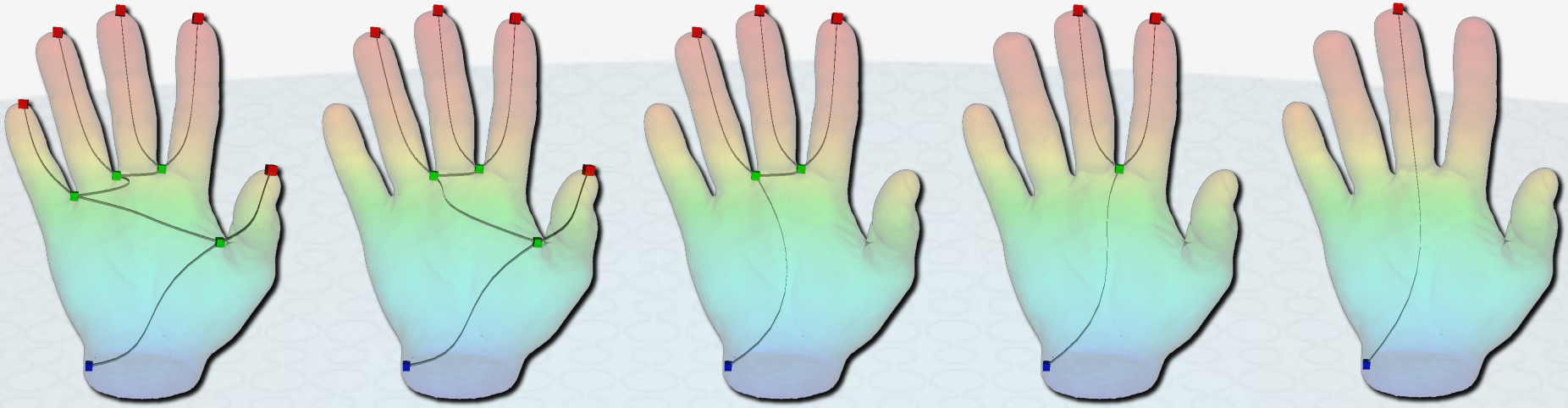
Hierarchical Topology Abstractions

- But we can do better than that...
  - Hierarchical representations of topology abstractions
- Notion of *Persistent Homology* [Edelsbrunner FOCS00]
  - General framework for the measurement of topological features
  - Originally: ranking of the classes of the Homology Groups
- Trivial development to topology abstractions

Hierarchical Topology Abstractions

- What does it mean wrt the Reeb graph?
  - We want to build a hierarchy (fine to coarse)
  - Representation with progressive complexity
  - From a computed RG(f)
    - Remove the arcs progressively (coarser version)
    - Ranking of the classes of the first Homology Group
    - Extended persistence: second Homology Group

Hierarchical Topology Abstractions

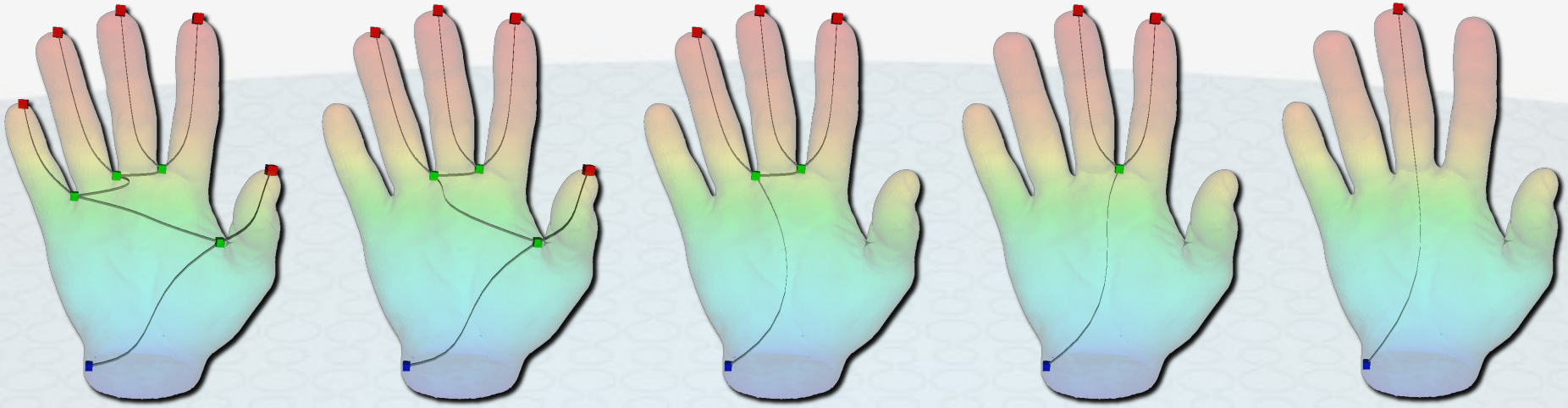


- Given a *persistence threshold*:
  - Remove progressively the arcs with smaller persistence
  - Maintain the topological structure
- One persistence threshold per hierarchy level
- Persistence threshold: originally, difference in f value

Hierarchical Topology Abstractions

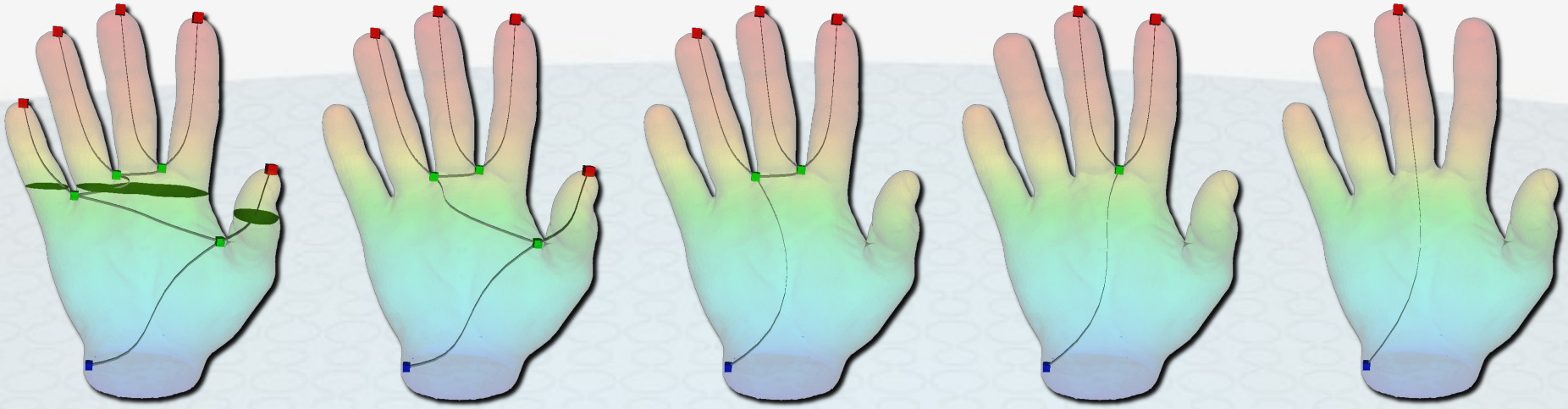
- **Flexible framework!**
  - Need to tune the importance ranking of features
- Persistence measurement:
  - Geometry-aware measures [Carr VIS04]:
    - Enclosed area/volume
    - Enclosed hyper-area/hyper-volume
    - etc.
- Specialized simplification: maintain features defined explicitly [Gyulassy VIS07]

What's Cool about That?



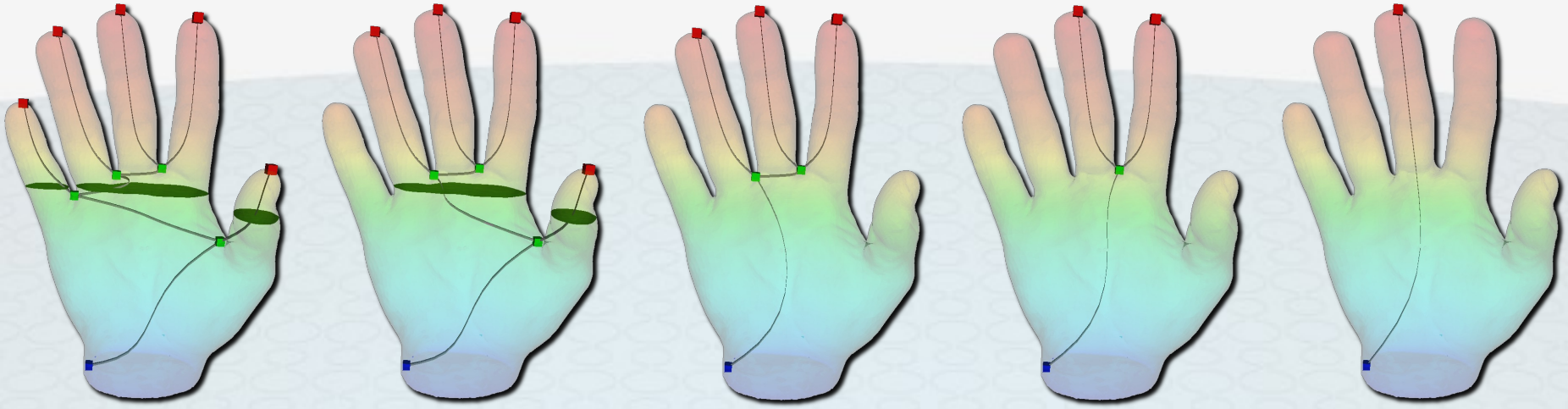
- Hierachical topology abstractions:
  - Simplified topology representations
  - "*Simplified*" access to the underlaying data
- Progressive access the iso-surfaces

What's Cool about That?



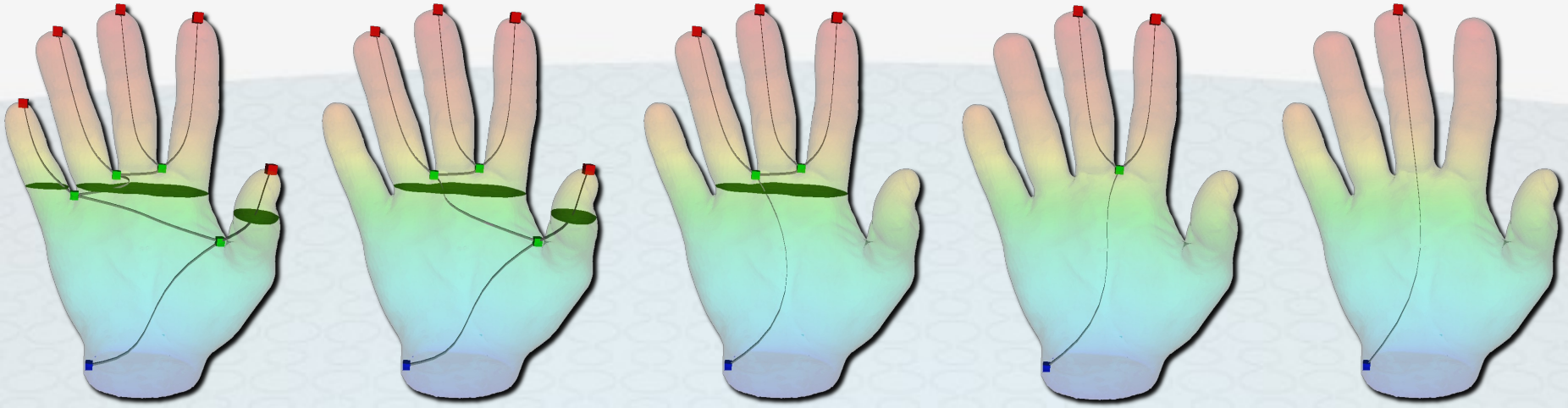
- Hierachical topology abstractions:
  - Simplified topology representations
  - "*Simplified*" access to the underlaying data
- Progressive access the iso-surfaces

What's Cool about That?



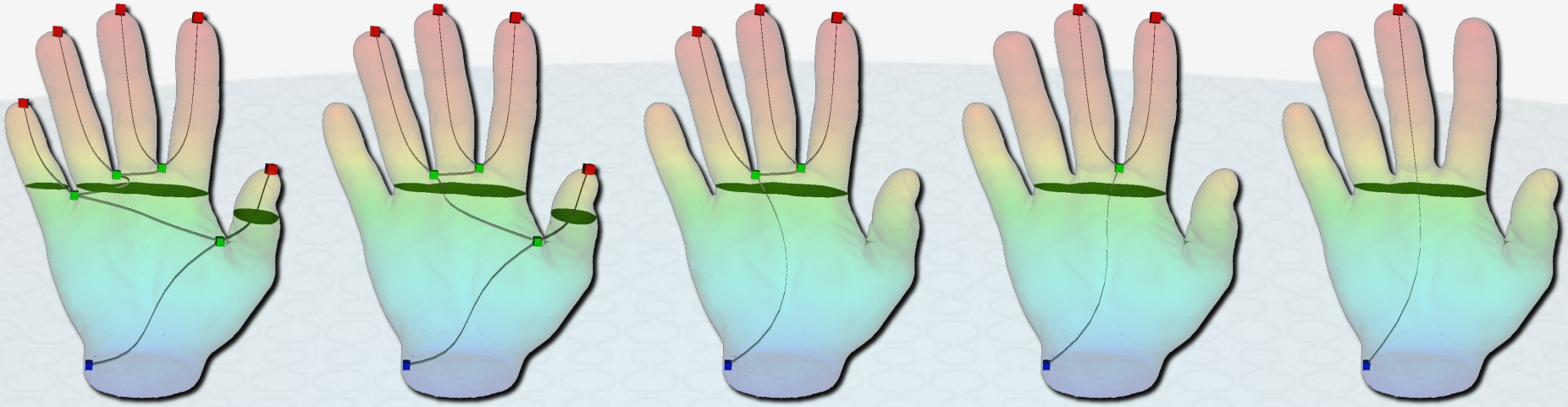
- Hierachical topology abstractions:
  - Simplified topology representations
  - "*Simplified*" access to the underlaying data
- Progressive access the iso-surfaces

What's Cool about That?



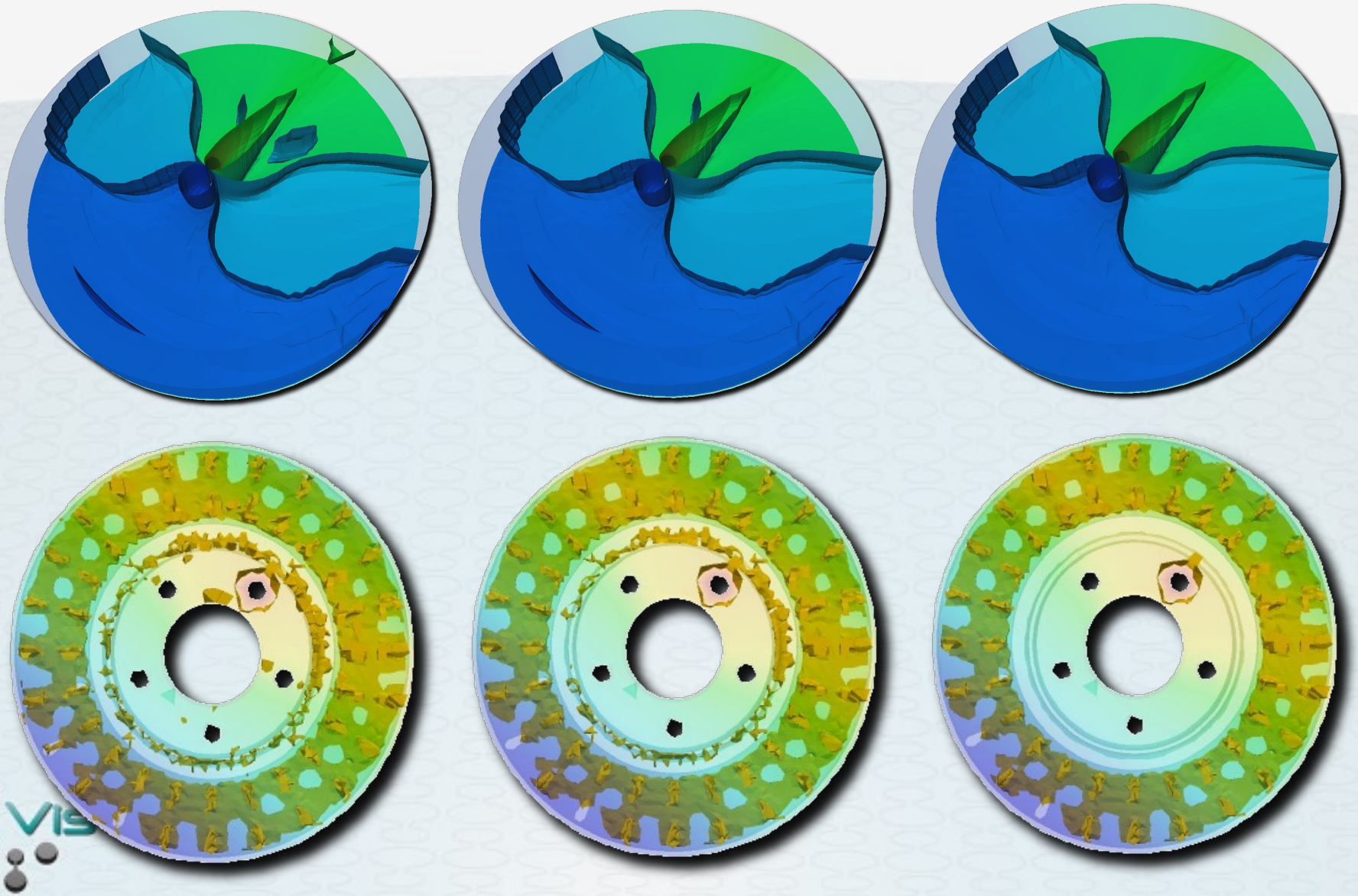
- Hierachical topology abstractions:
  - Simplified topology representations
  - "*Simplified*" access to the underlaying data
- Progressive access the iso-surfaces

What's Cool about That?



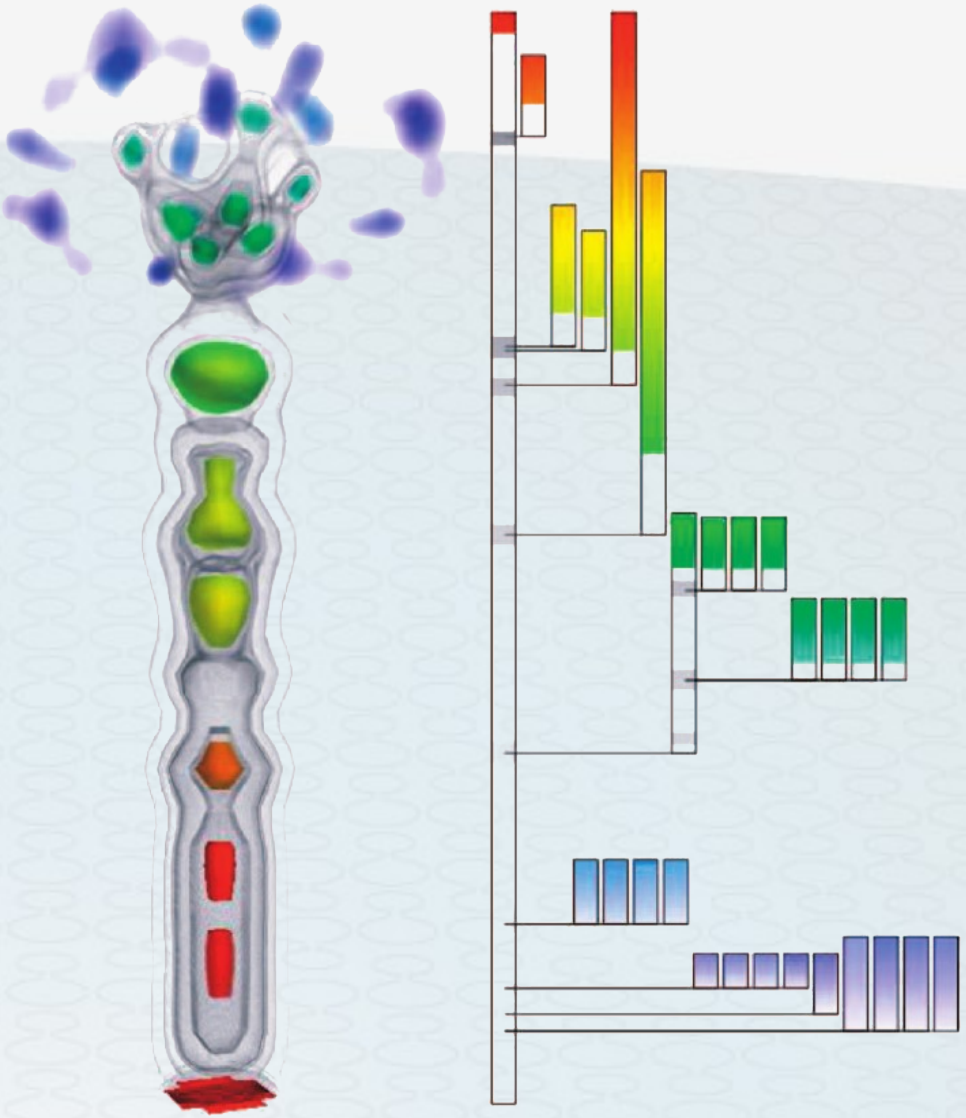
- Hierachical topology abstractions:
  - Simplified topology representations
  - "*Simplified*" access to the underlaying data
- Progressive access the iso-surfaces

Topologically Clean Isosurfaces



Other Examples

- Flexible Iso-surfaces
  - [Carr VIS04]
- Visual Metaphors
  - [Weber VIS07]
- Transfer Function Design:
  - [Bajaj VIS97]
  - [Weber TVCG07]

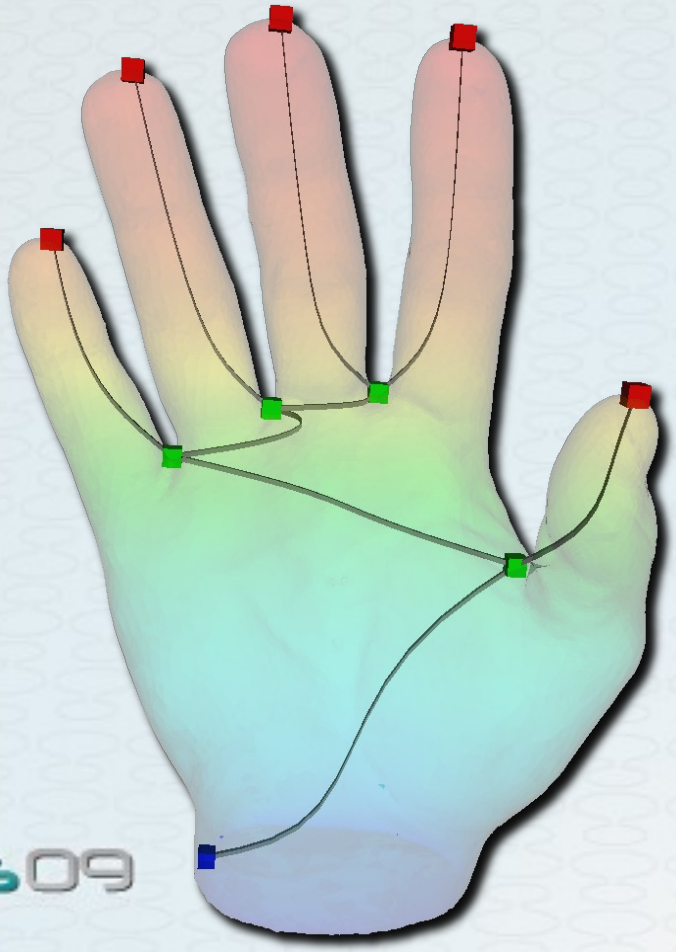


Now... What's the Point?

- The Reeb graph provides
  - An **abstract** representation of the topology of level sets
  - A fast access to level sets
  - A flexible access to level sets
- But
  - To convey its expressive power to the user,
  - We need to **display it** to the user (interaction)

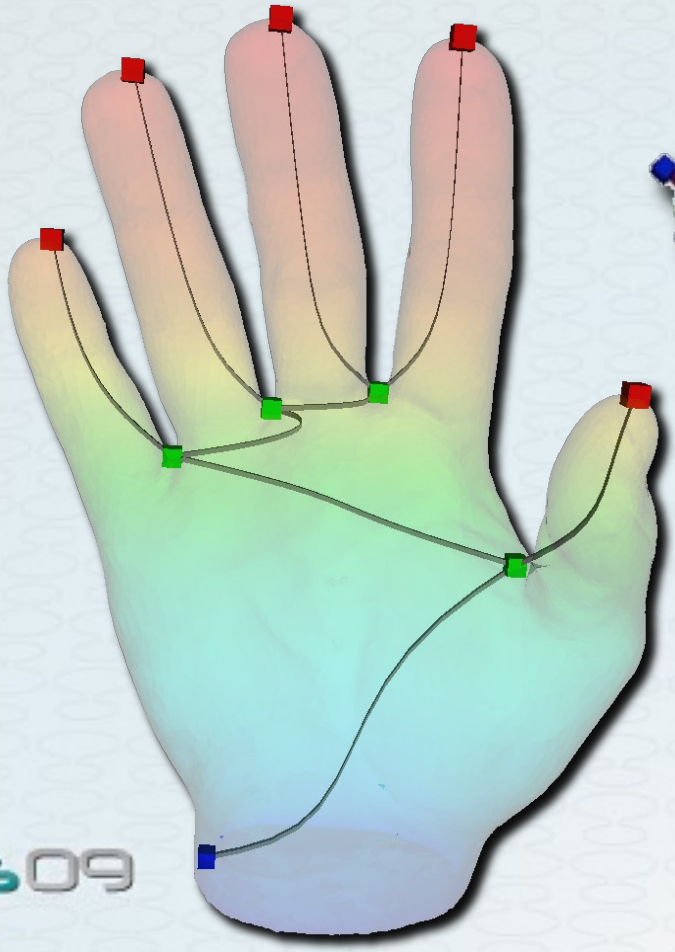
Reeb Graph Display

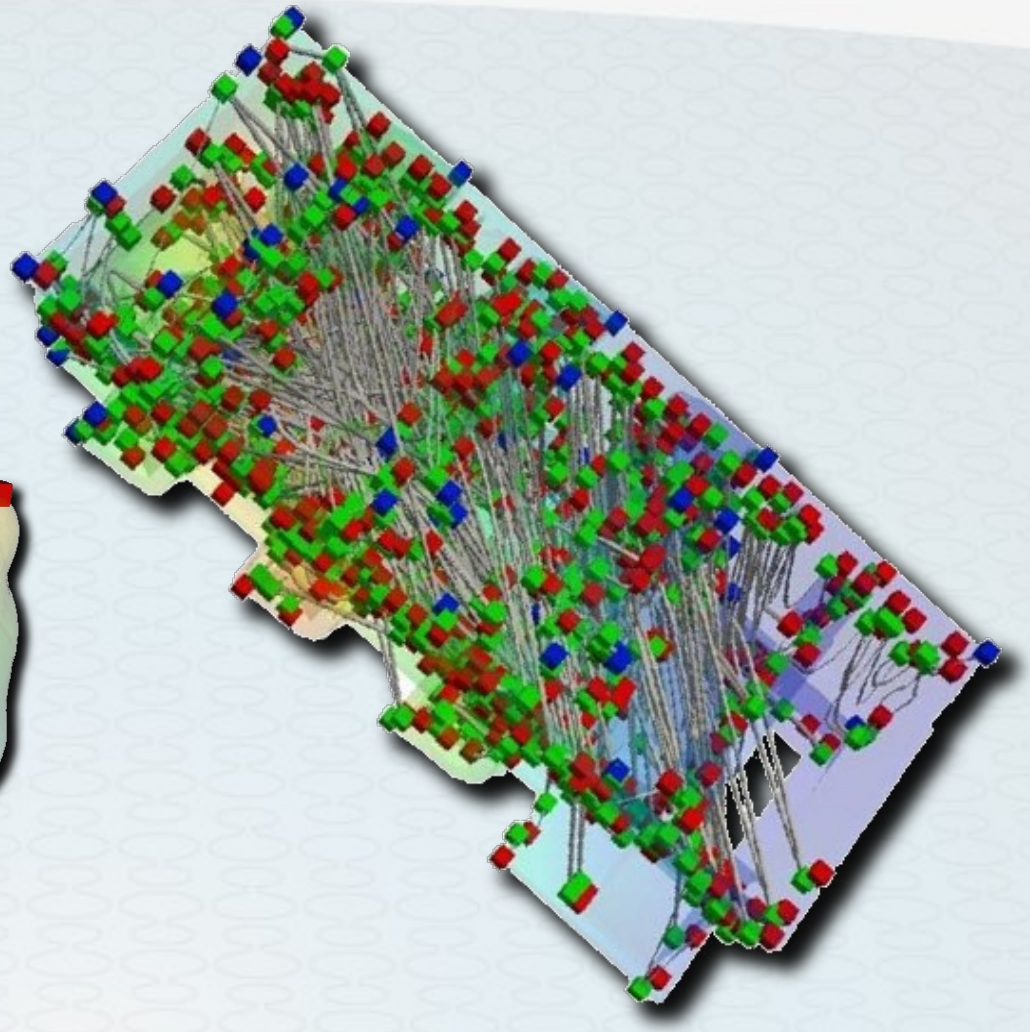
- Is it really a problem?



Reeb Graph Display

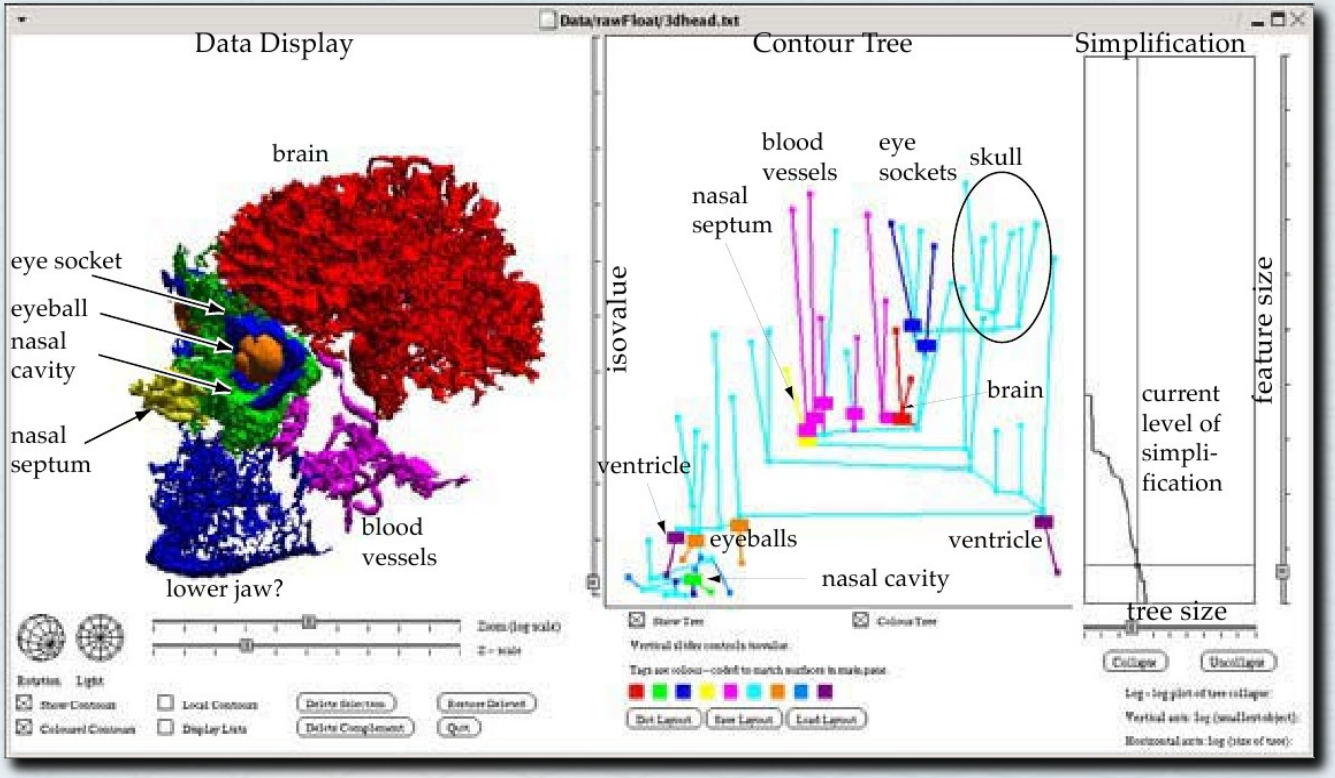
- Is it really a problem?





Reeb Graph Interaction

- Need for intuitive visual representation
- Supporting user interaction



Reeb Graph Based Interactive Visualization: an **InfoVis** Problem

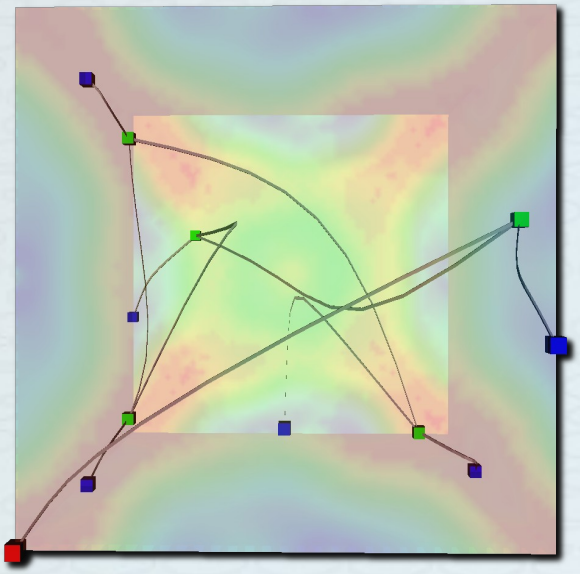
- Input data:
  - An abstract representation of spatio-temporal data
  - A planar graph
    - Can contain thousands of arcs!
- Problem:
  - Need to represent this data in an intelligible way
  - Support for interaction (sub-graph selection)

Solution Panorama

- Hierarchical Embeding of the Reeb graph
- Visualization Metaphors
- Planar layouts
- Abstract 3D layouts

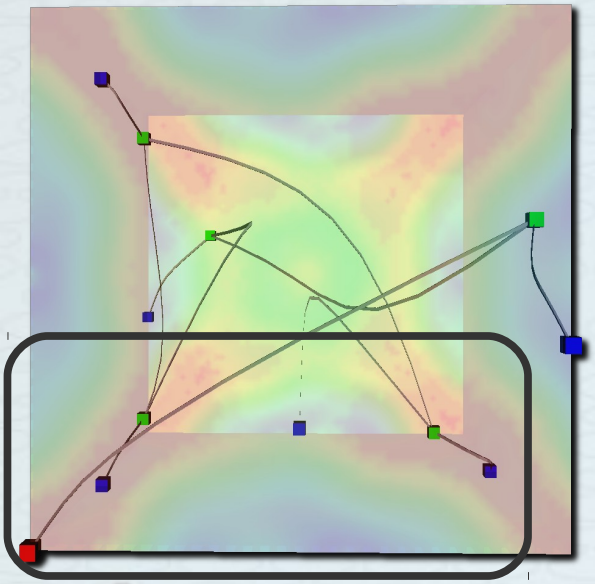
Hierachical Embedding

- View dependent embedding:
  - Select a level in the hierarchy of the RG
  - Limit the set of rendered arcs



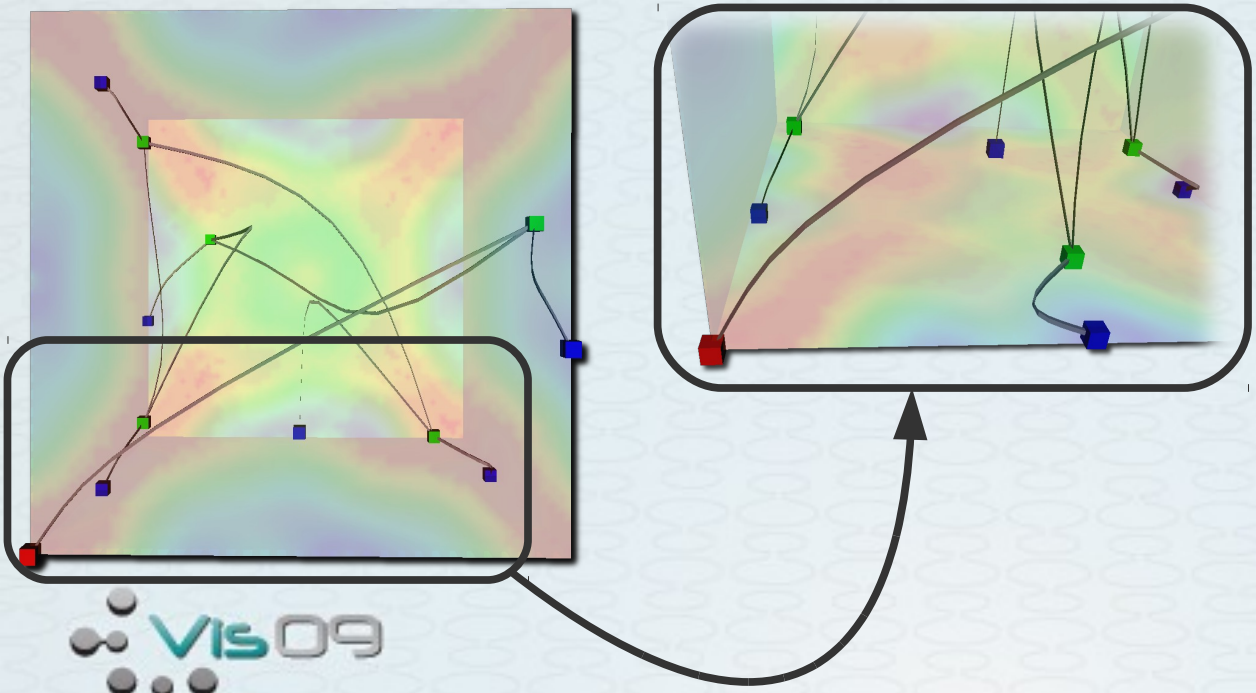
Hierachical Embedding

- View dependent embedding:
  - Select a level in the hierarchy of the RG
  - Limit the set of rendered arcs



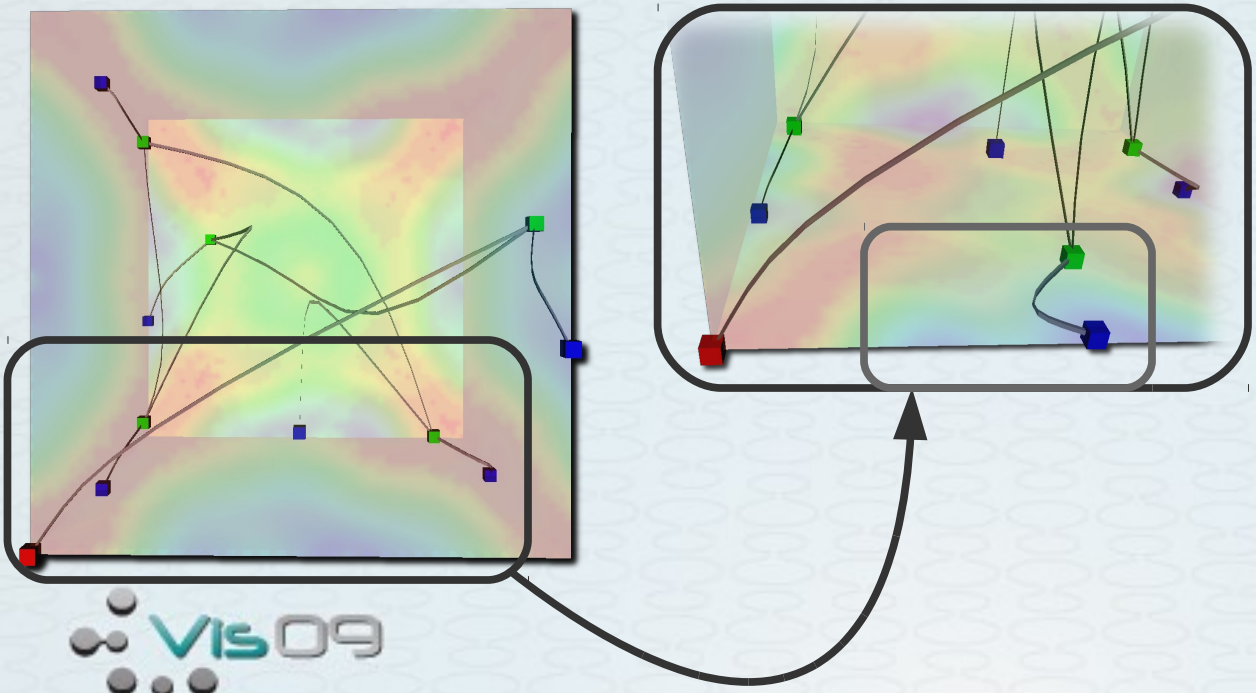
Hierachical Embedding

- View dependent embedding:
  - Select a level in the hierarchy of the RG
  - Limit the set of rendered arcs



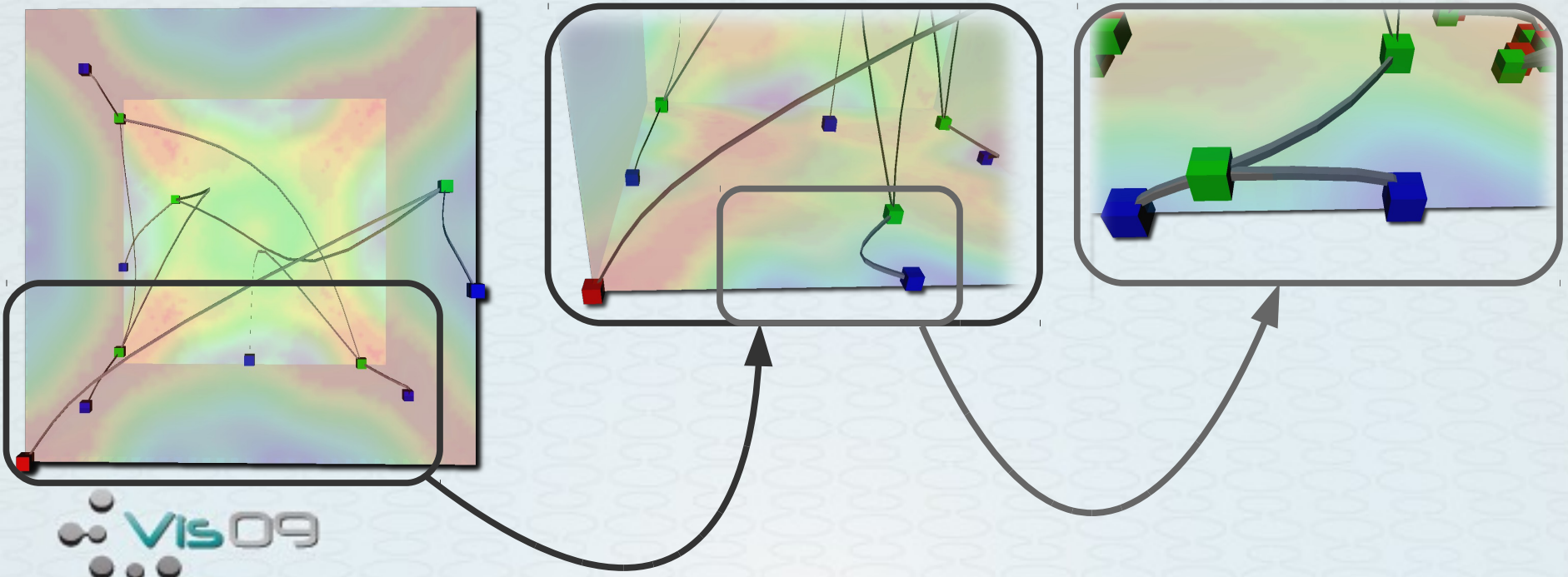
Hierachical Embedding

- View dependent embedding:
  - Select a level in the hierarchy of the RG
  - Limit the set of rendered arcs



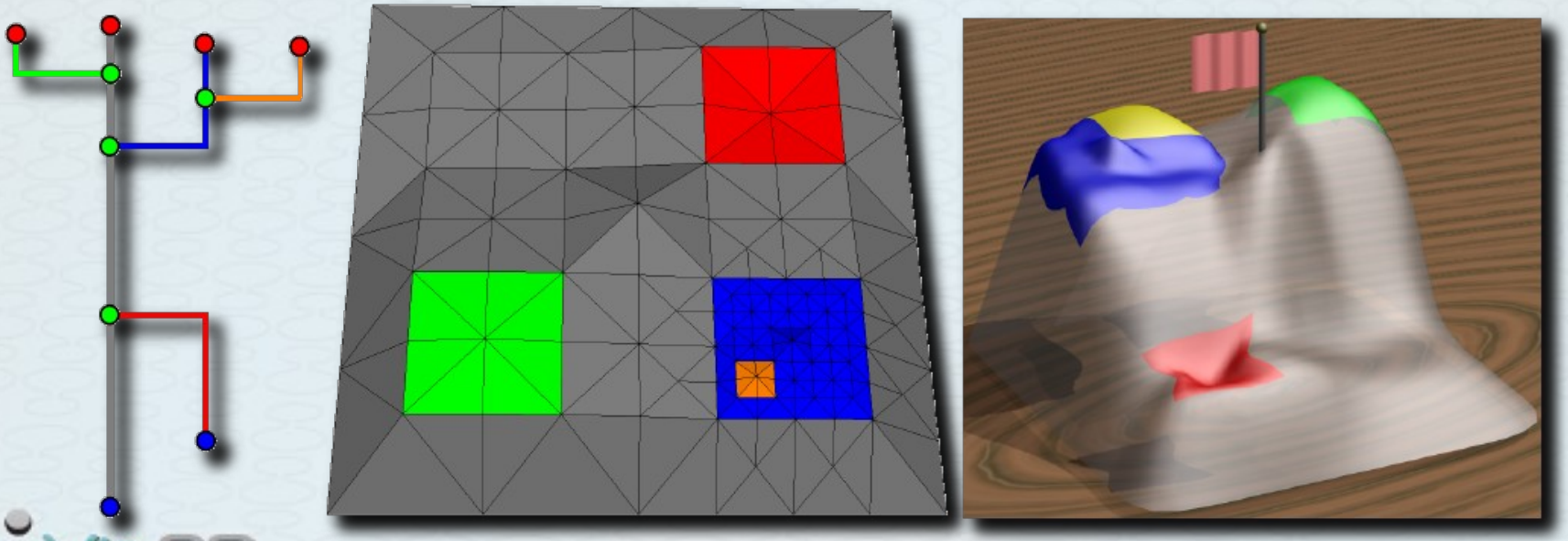
Hierachical Embedding

- View dependent embedding:
  - Select a level in the hierarchy of the RG
  - Limit the set of rendered arcs

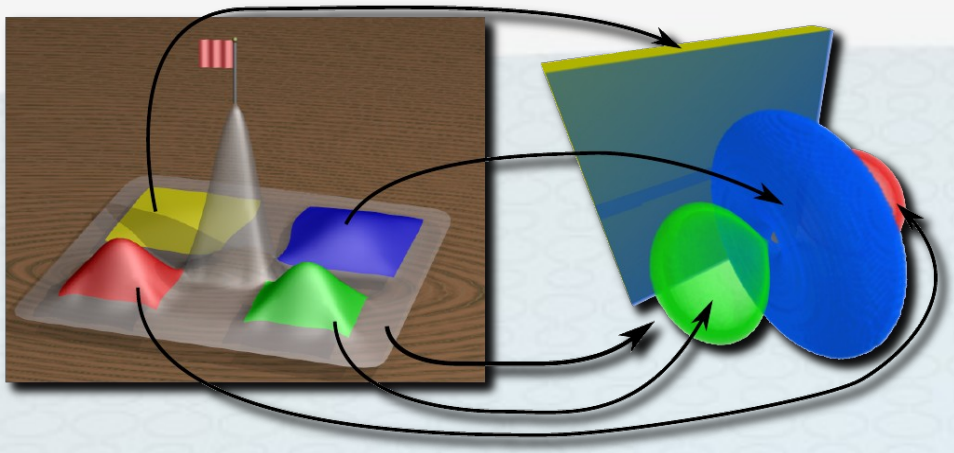


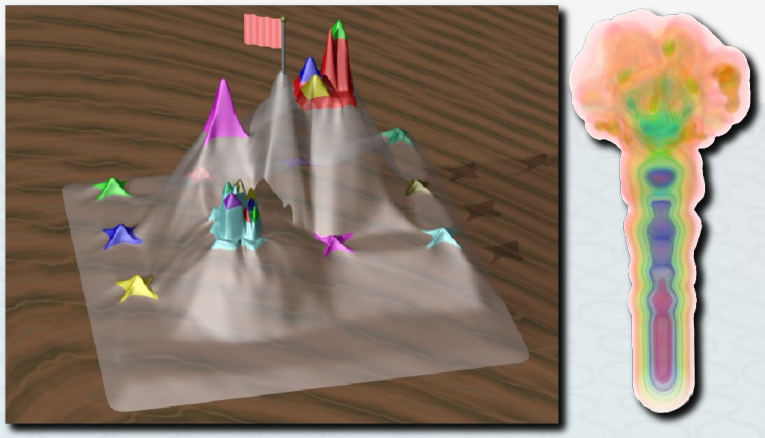
Visualization Metaphors

- Overcome 3D embedding ambiguities (self-occlusion)
- Generate a visual metaphor (for loop-free Reeb graphs only)
  - Construct a 3D **terrain** representing the Reeb graph
  - Segmentation following the elder's rule



Visualization Metaphors

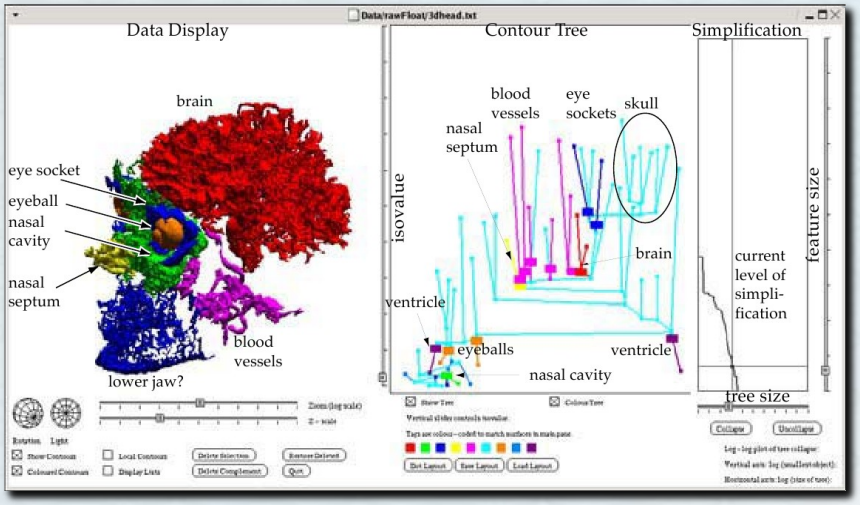


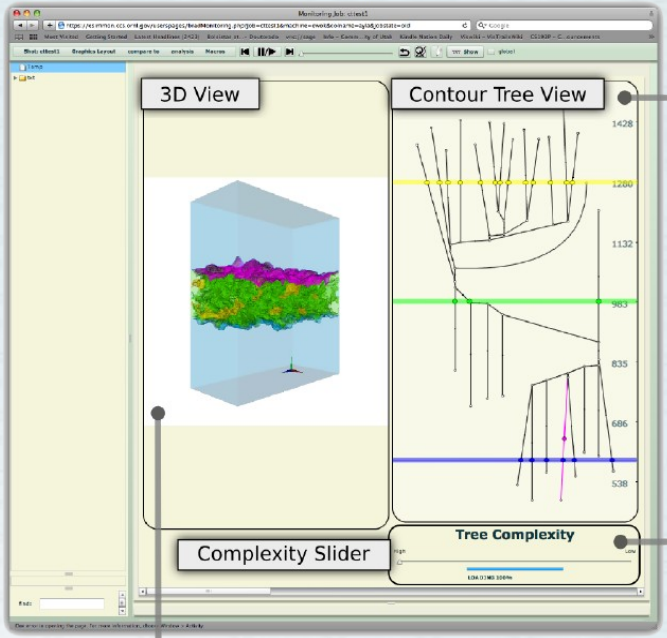


- Intuitive representation of the Reeb graph
- Hierarchy of terrains (hierarchy of Reeb graphs)
- However:
  - Applies an abstraction to an abstraction
  - Not always intuitive

Planar Layouts

- May be simpler to consider the Reeb graph for what it is:
  - … a planar graph
  - Significant litterature for planar graph layouts
  - GraphViz **http://www.graphviz.org**



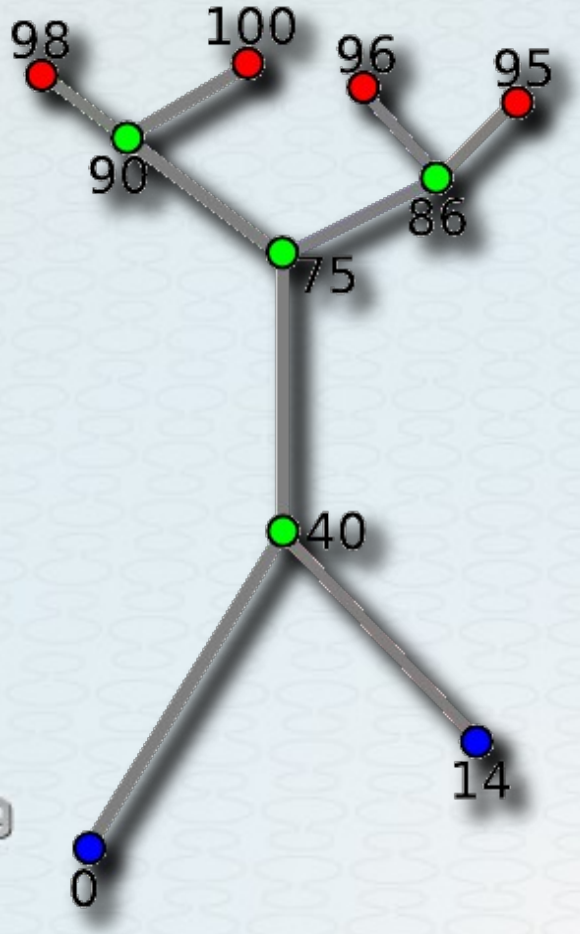


Reeb Graph Layout Problem

- Reeb graphs are a special case of planar graphs
- Properties on the degree of the nodes
- Aesthetic constraints
  - Y-value of each node: actual scalar value
  - Edge crossing should be avoided
  - Notion of parent-child arcs
    - Elder's rule
    - Persistent homology

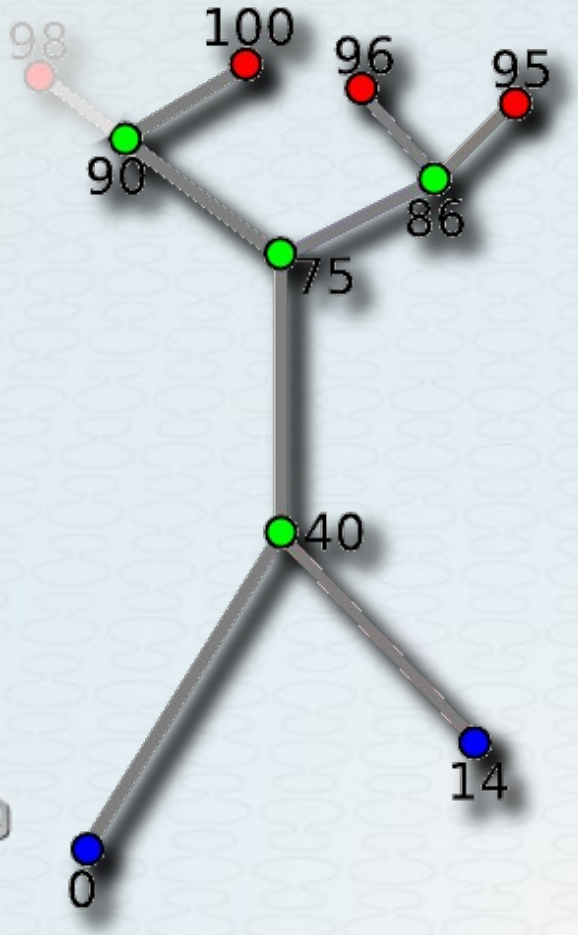
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



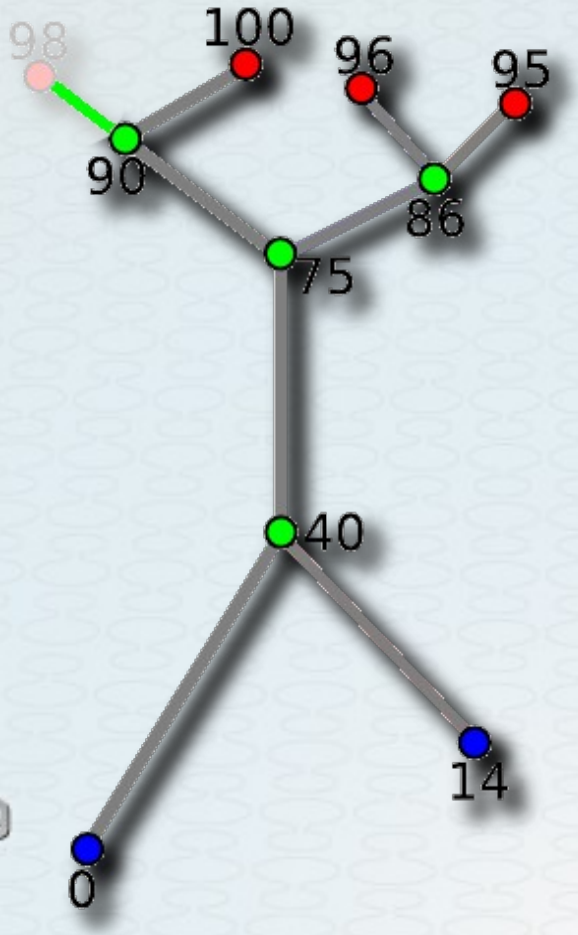
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



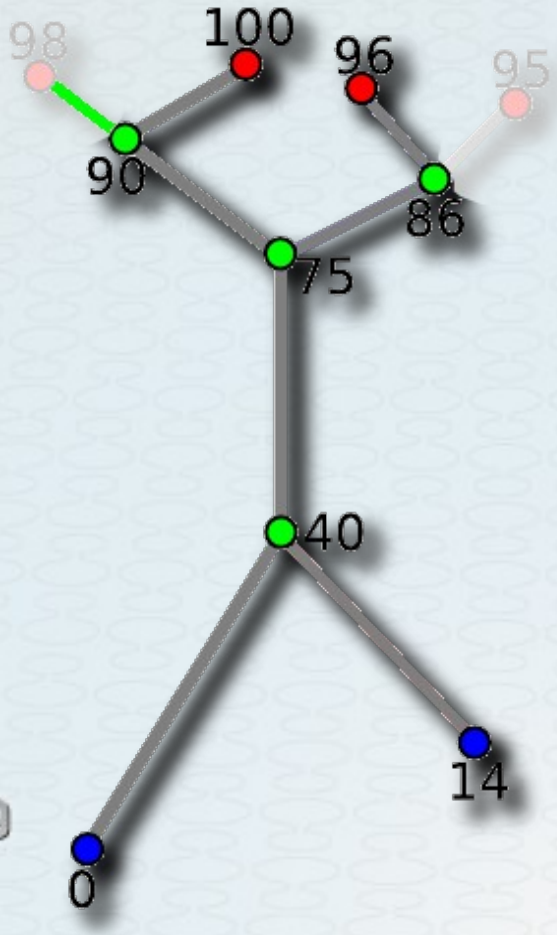
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



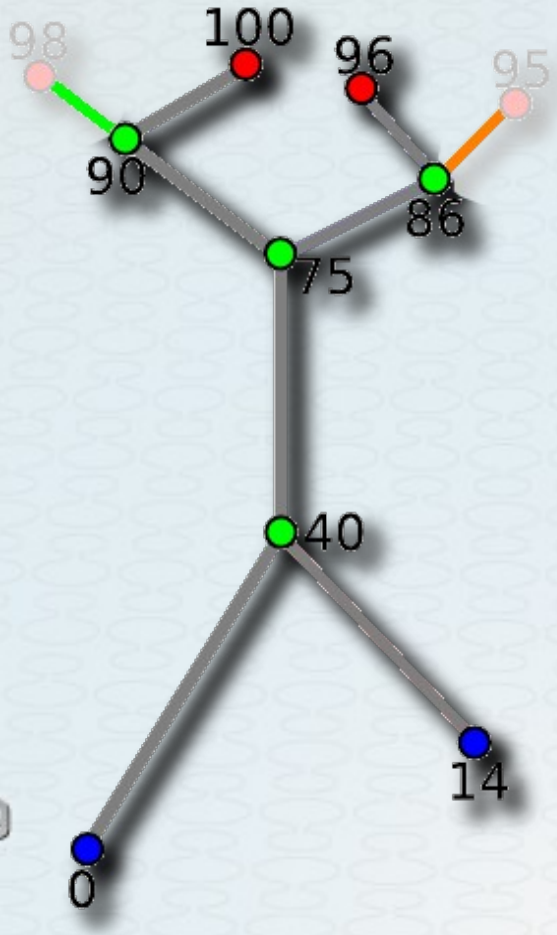
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



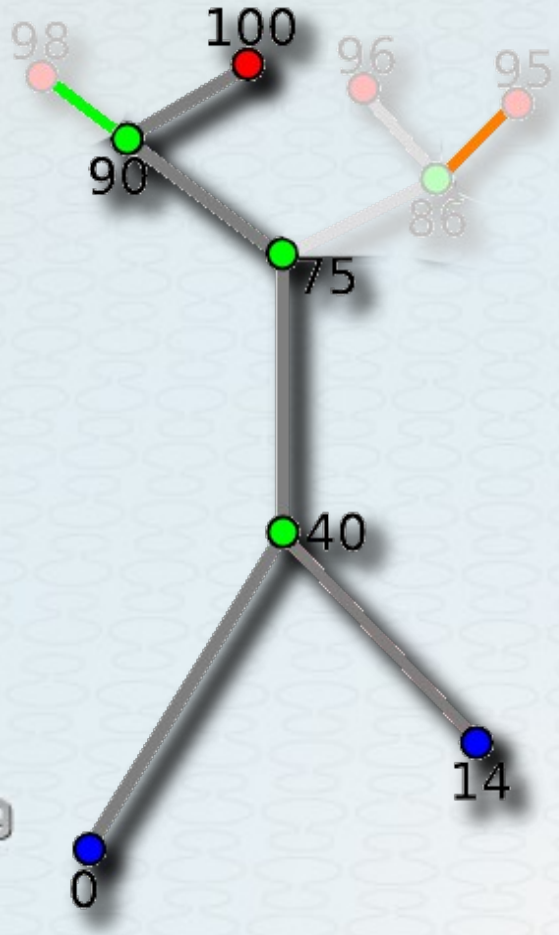
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



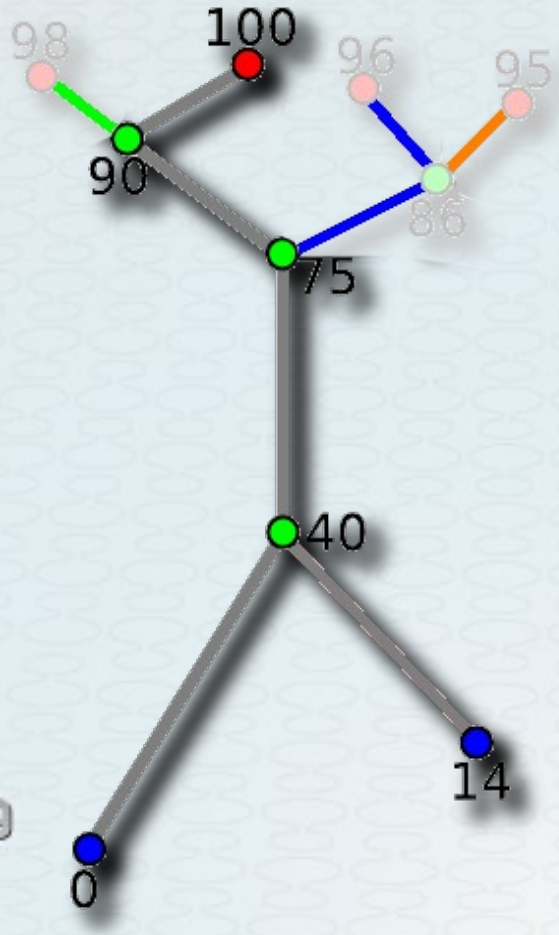
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



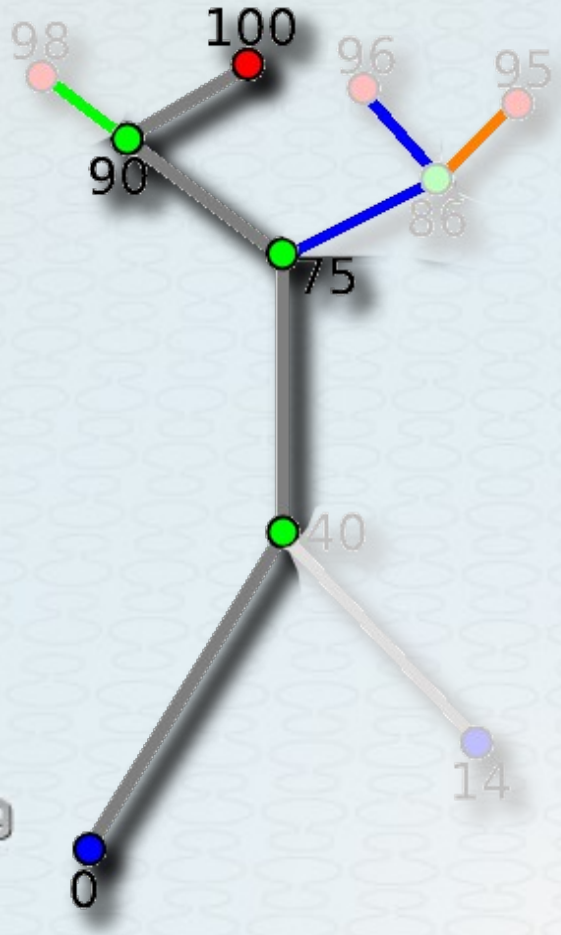
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



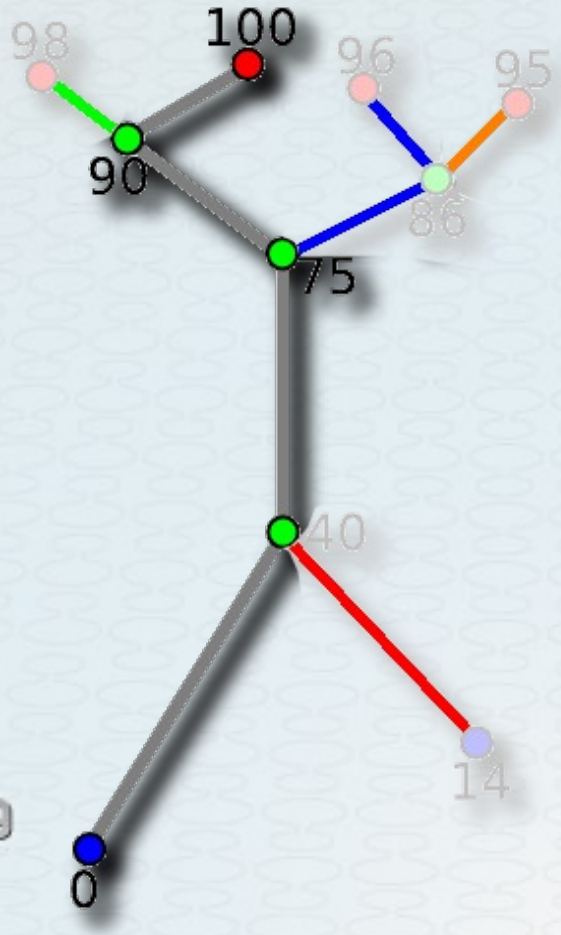
Branch Decomposition

- Decompose the Reeb graph according to the hierarchy construction process



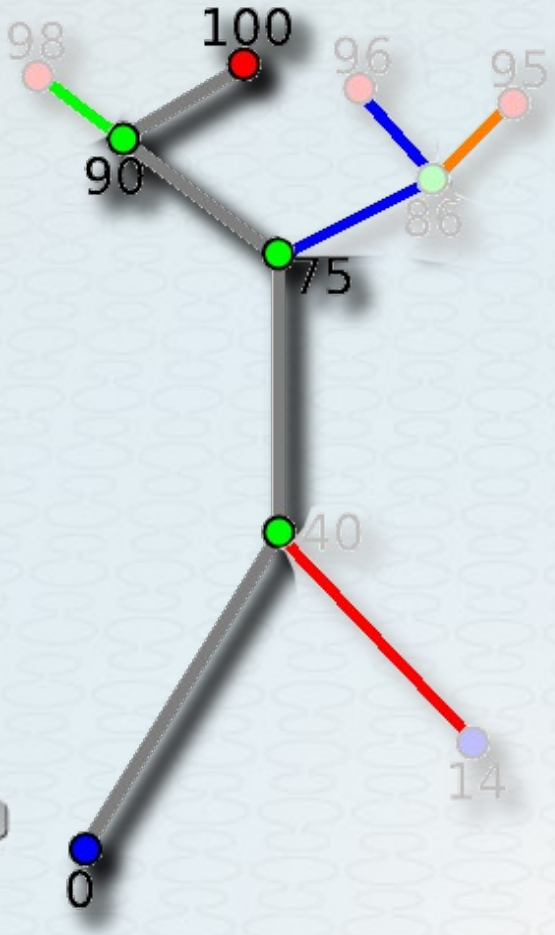
Branch Decomposition

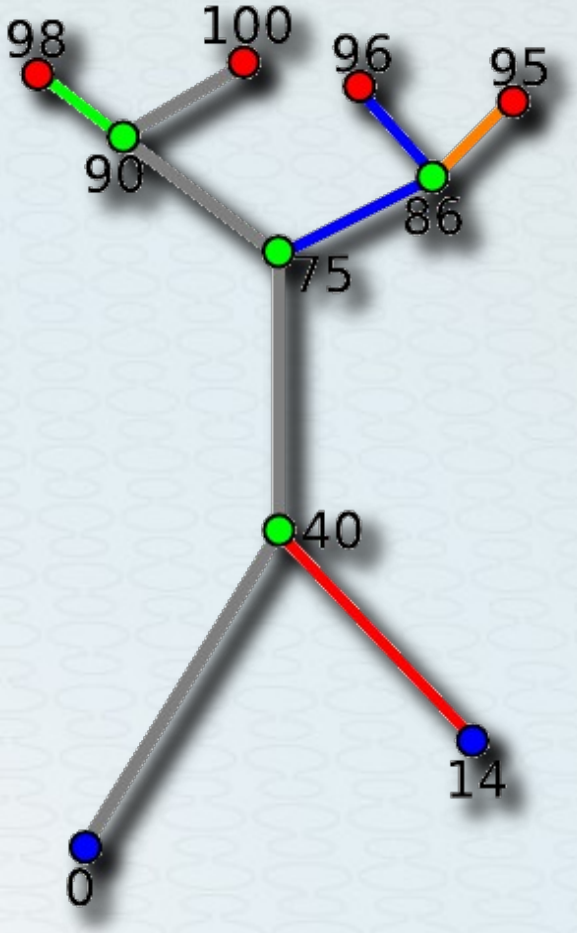
- Decompose the Reeb graph according to the hierarchy construction process



Branch Decomposition

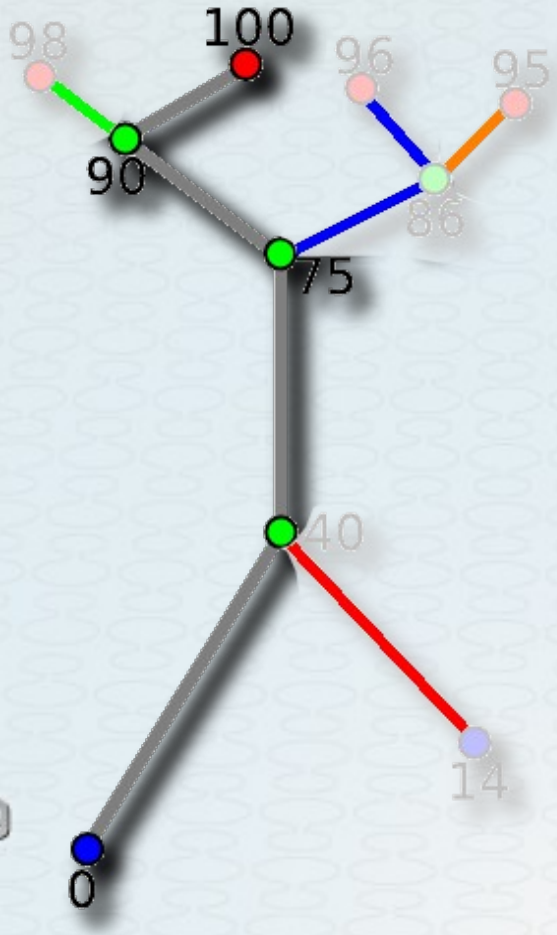
- Decompose the Reeb graph according to the hierarchy construction process

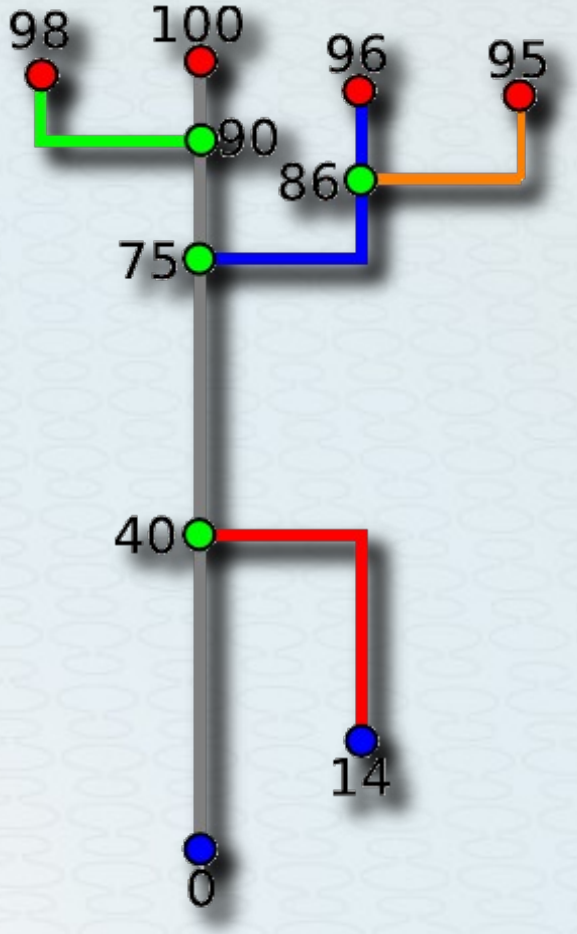




Branch Decomposition

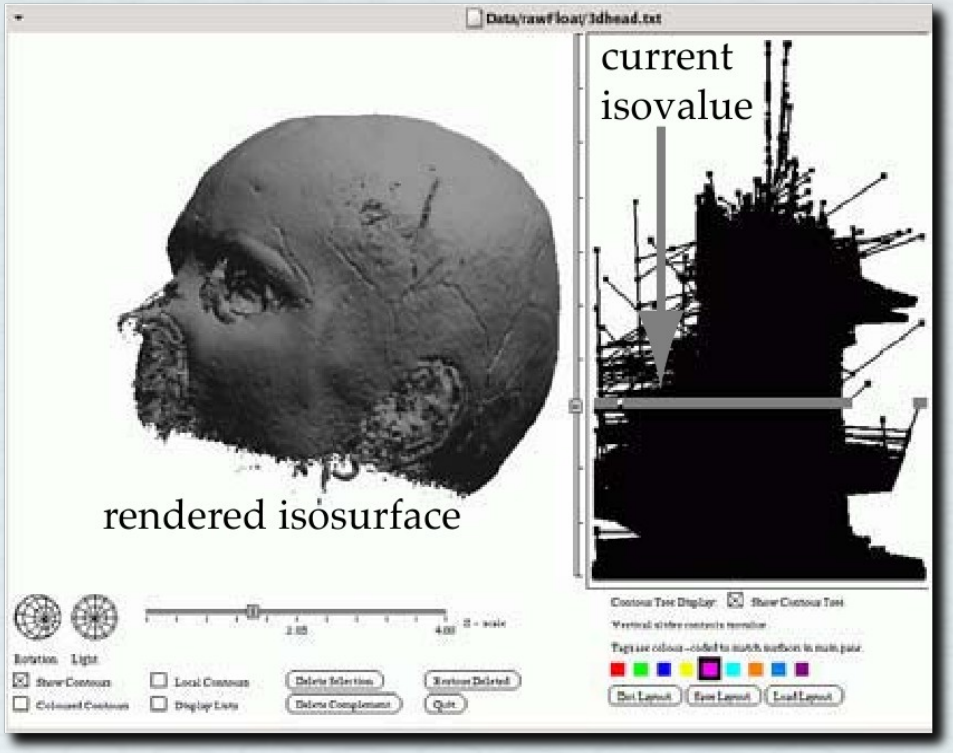
- Decompose the Reeb graph according to the hierarchy construction process





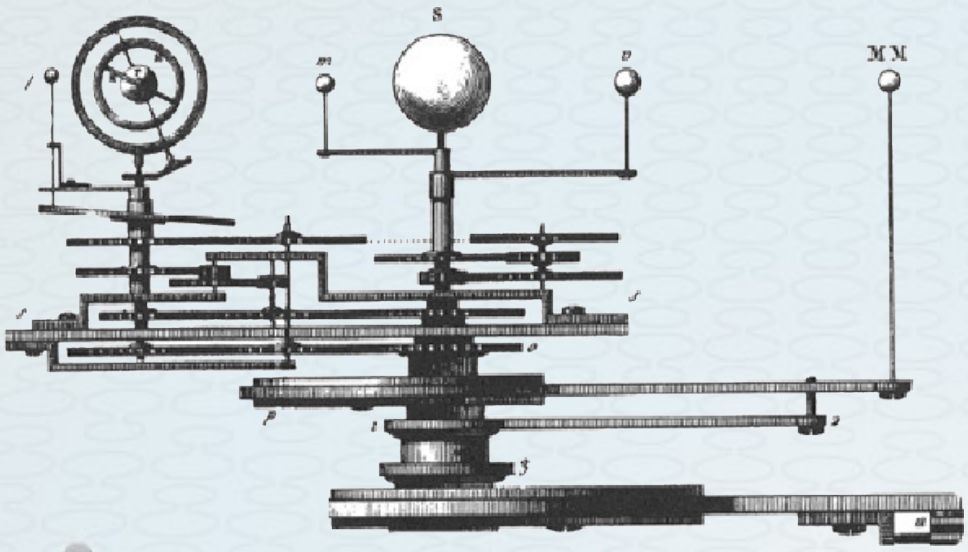
Drawbacks of Planar Layout

- Can get messy rapidly...
- Dimension reduction problem:
  - Nodes: critical points
  - Can be close to each other in 3D
  - Have to be distant in the plane



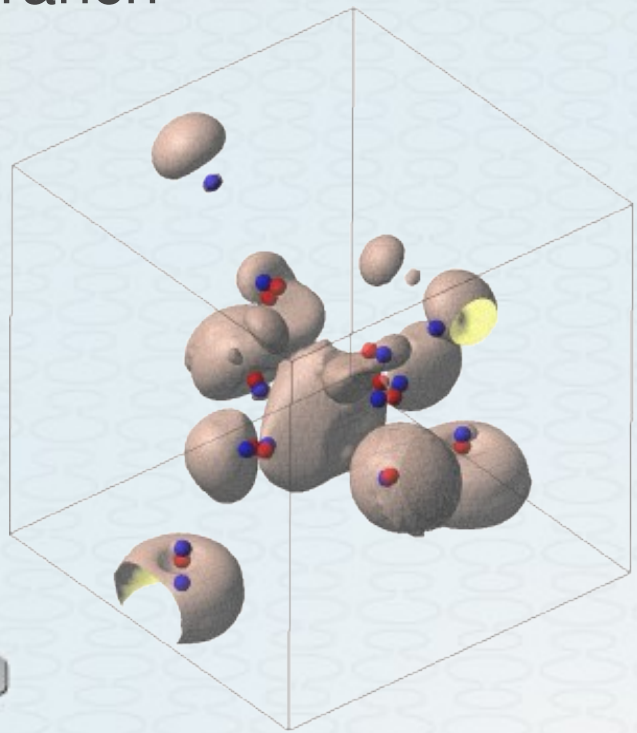
Abstract 3D Layouts

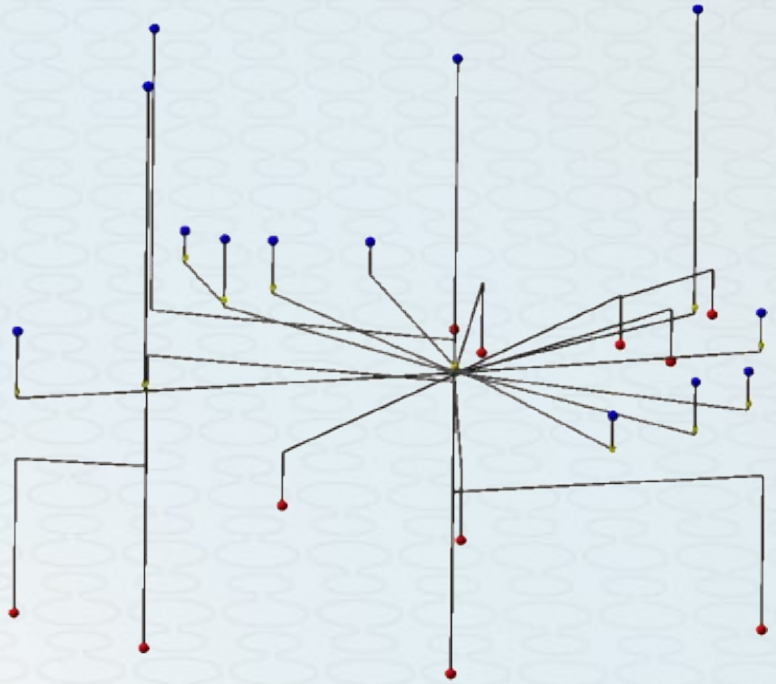
- Overcome planar graph density issues:
  - Overcome the dimension reduction issue
  - **Orrery** metaphor



Toporrery Layout

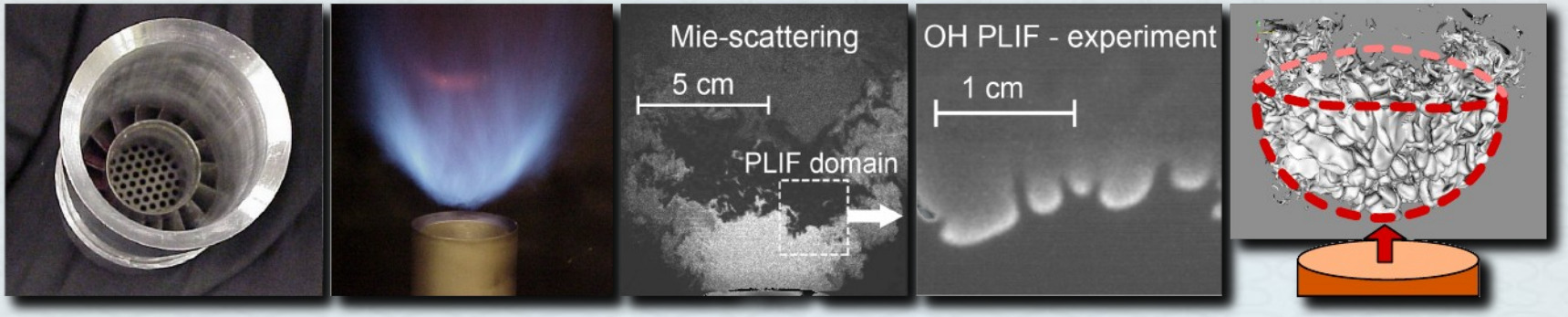
- Represent the Reeb graph as an orrery
- Branch decomposition of the Reeb graph
- Apply and additional **rotation** for each child branch of a parent branch

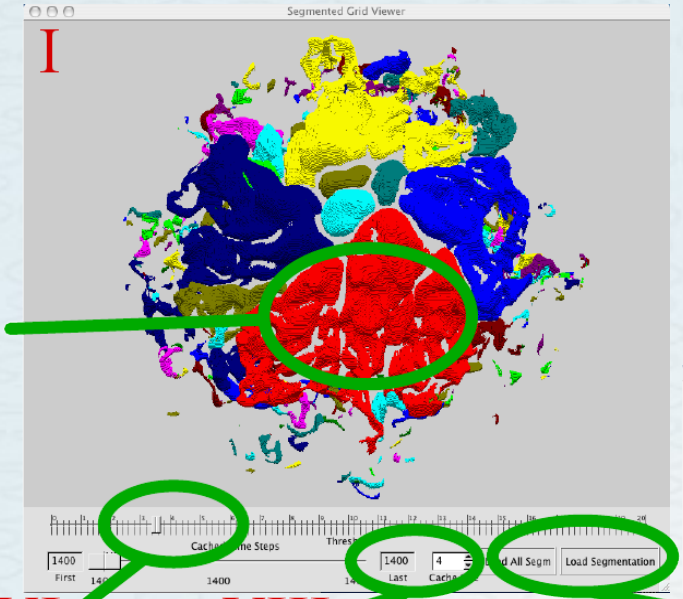


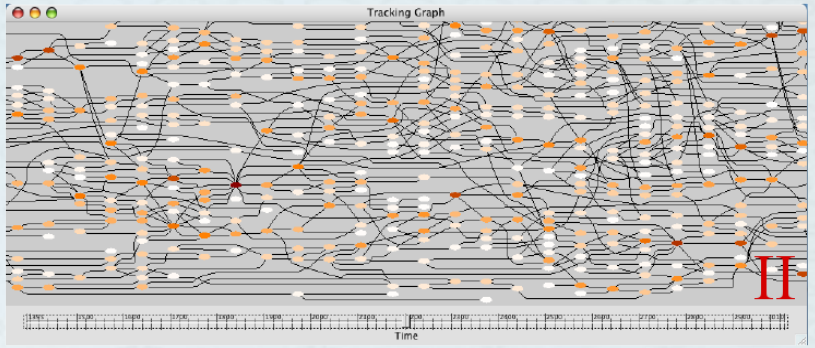


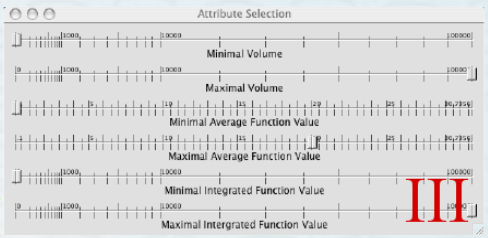
VisInfoVis: ...It Can Get Even Worse

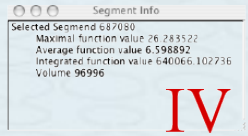
- Especially with time-dependent data...











Conclusion

- A lot of **visualization** techniques compute **abstract** representations of the data (topology)

- To give the user the full potential of those representations:
  - Need to **visualize** these **abstract** representations
  - **InfoVis** problem
- In the future:
  - Large-scale data, time-varying data
  - The Vis community will need the InfoVis community more than ever!