



CS 247 – Scientific Visualization

Lecture 3: The Visualization Pipeline; Data Representation, Pt. 1

Markus Hadwiger, KAUST

Reading Assignment #2 (until Feb 7)



Read (required):

- Data Visualization book, finish Chapter 2
- Data Visualization book, Chapter 3 until 3.5 (inclusive)
- Data Visualization book, Chapter 4 until 4.1 (inclusive)

- Continue familiarizing yourself with OpenGL if you do not know it !

Programming Assignments Schedule (tentative)



| | | | |
|---------------|--|-------|---------------|
| Assignment 0: | Lab sign-up: setup piazza + github account, get git repo Basic OpenGL example [we will offer a tutorial!] | until | Jan 31 |
| Assignment 1: | Volume slice viewer | until | Feb 13 |
| Assignment 2: | Iso-contours (marching squares) | until | Feb 27 |
| Assignment 3: | Iso-surface rendering (marching cubes) | until | Mar 15 |
| Assignment 4: | Volume ray-casting, part 1 | until | Mar 31 |
| | Volume ray-casting, part 2 | until | Apr 7 |
| Assignment 5: | Flow vis, part 1 (hedgehog plots, streamlines, pathlines) | until | Apr 21 |
| Assignment 6: | Flow vis, part 2 (LIC with color coding) | until | May 5 |

Programming Assignment #1: Slice Viewer



Basic tasks

- Download data into 3D volume texture
- Display three different axis-aligned slices using OpenGL texture mapping using the 3D volume texture

Minimum

- The slice position should be adjustable for each slice view.
- Make sure the aspect ratio of the shown slices is correct.
- If the window is resized, the slice is resized with the correct aspect ratio (no distortions)

Bonus

- Show all three axis aligned slices at once
- Show arbitrarily aligned slices with an interface to change the arbitrary slice

Programming Assignment #1 Example



The screenshot shows a development environment with three windows:

- Terminal Window:** Displays the command-line output of the application. It includes the OpenGL version information, keyboard commands, dataset loading details, and a series of "increasing current slice" messages from 86 to 90.
- Slice Viewer Window:** A grayscale image showing a cross-section of a head or skull, likely a CT scan or similar medical image.
- Code Editor:** Shows the source code for a function named `printOpenGLError`. The code uses OpenGL error handling to print errors to the terminal.

```
#include <iostream>
G\_VERSION major=4 minor=3
Keyboard commands:
b - Toggle among background clear colors
w - Increase current slice
s - Decrease current slice
a - Toggle viewing axis
1 - Load lobster dataset
2 - Load head dataset
3 - Load hydrogen dataset
loading data ../../Datasets/skewed_head.dat
volume dimensions: x: 184, y: 256, z:170
downloading volume to 3D texture
increasing current slice: 86
increasing current slice: 87
increasing current slice: 88
increasing current slice: 89
increasing current slice: 90

int printOpenGLError(char *file, int line)
{
    // Returns 1 if an OpenGL error occurred, 0 otherwise.
    GLenum glErr;
    int retCode = 0;

    glErr = glGetError();
    while (glErr != GL_NO_ERROR)
    {
        printf("glError in file %s @ line %d: %s\n", file, line, gluErrorString(glErr));
        retCode = 1;
        glErr = glGetError();
    }
    return retCode;
}

#define printOpenGLError() printOpenGLError((char *) __FILE__, __LINE__)
```

Programming Assignment #1 Example



The screenshot shows a Windows desktop environment with three windows open:

- Terminal Window:** Displays the command-line output of the application. It includes a help menu at the top and a log of operations performed on a volume dataset.
- Slice Viewer Window:** Shows a grayscale 3D volume rendering of a biological specimen, likely a head or skull, with internal structures visible.
- Code Editor Window:** Displays the source code for a function named `printOpenGLError`. The code uses OpenGL error handling to print errors to the terminal.

```
b - Toggle among background clear colors
w - Increase current slice
s - Decrease current slice
a - Toggle viewing axis
1 - Load lobster dataset
2 - Load head dataset
3 - Load hydrogen dataset
loading data ../../Datasets/skewed_head.dat
volume dimensions: x: 184, y: 256, z:170
downloading volume to 3D texture
increasing current slice: 86
increasing current slice: 87
increasing current slice: 88
increasing current slice: 89
increasing current slice: 90
toggling viewing axis to: 0
increasing current slice: 93
increasing current slice: 94
increasing current slice: 95
toggling viewing axis to: 1
decreasing current slice: 127
decreasing current slice: 126
decreasing current slice: 125
decreasing current slice: 124

int printOpenGLError(char *file, int line)
{
    // Returns 1 if an OpenGL error occurred, 0 otherwise.
    GLenum glErr;
    int retCode = 0;

    glErr = glGetError();
    while (glErr != GL_NO_ERROR)
    {
        printf("glError in file %s @ line %d: %s\n", file, line, gluErrorName(glErr));
        retCode = 1;
        glErr = glGetError();
    }
    return retCode;
```

Programming Assignment #1 Example



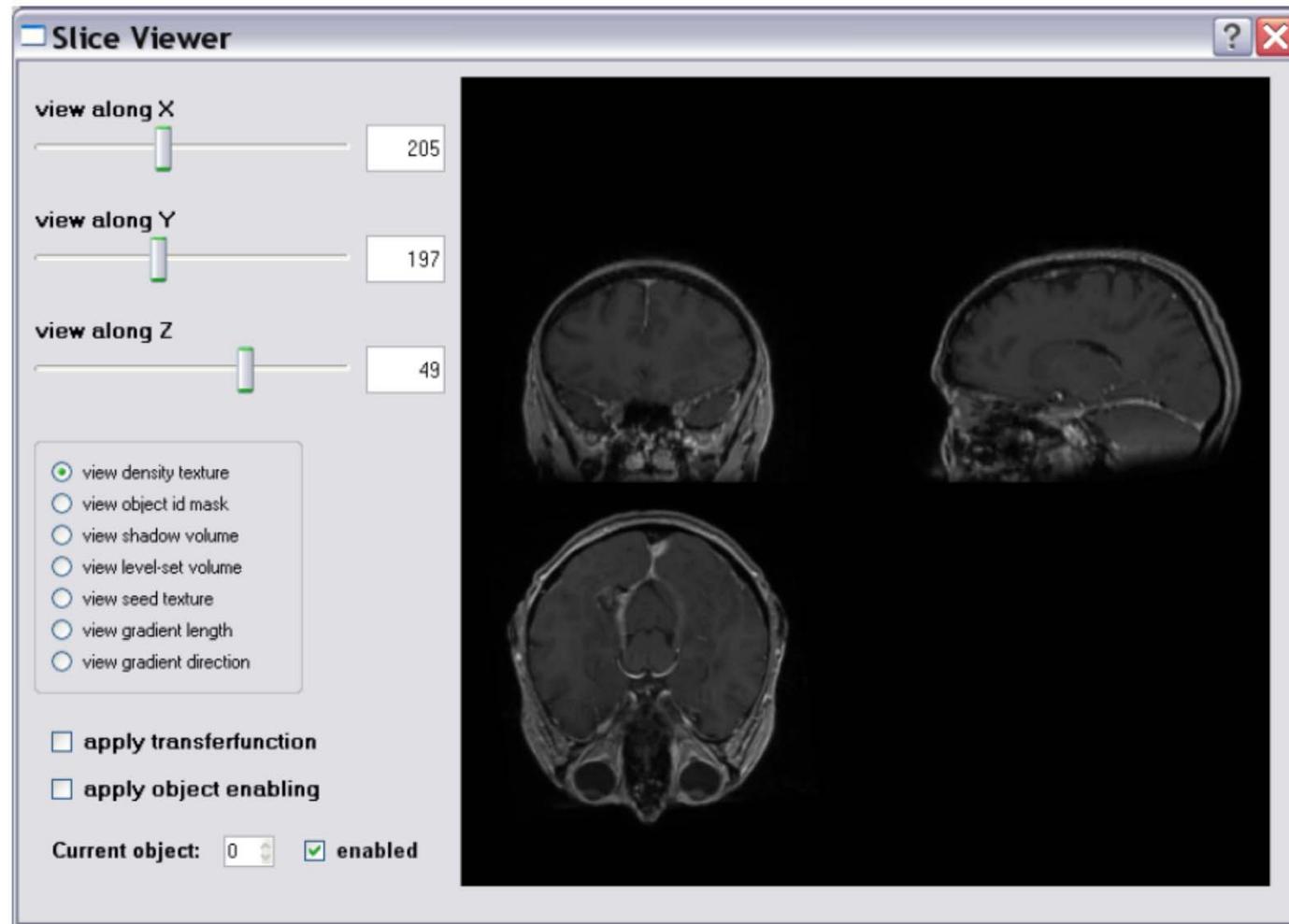
The screenshot shows a software interface for scientific visualization. On the left, a terminal window displays the output of a program named CS247_Assignment1.exe. The output includes the OpenGL version (GL_VERSION major=4 minor=3), keyboard commands, and a series of messages indicating the loading of datasets and the processing of slices. On the right, a window titled "AMCS/CS247 Scientific Visualization - Slice Viewer" displays a grayscale 3D volume rendering of a biological specimen, likely a head or brain, showing internal structures like the brainstem and cerebellum. Below the terminal window, a code editor shows a portion of C++ code for handling OpenGL errors.

```
#include <iostream>
G\Development\git\Teaching\Work\CS247_Assignment1\x64\Debug\CS247_Assignment1.exe
GL_VERSION major=4 minor=3
Keyboard commands:
b - Toggle among background clear colors
w - Increase current slice
s - Decrease current slice
a - Toggle viewing axis
1 - Load lobster dataset
2 - Load head dataset
3 - Load hydrogen dataset
loading data ../../Datasets/skewed_head.dat
volume dimensions: x: 184, y: 256, z:170
downloading volume to 3D texture
increasing current slice: 86
increasing current slice: 87
increasing current slice: 88
increasing current slice: 89
increasing current slice: 90
toggling viewing axis to: 0
increasing current slice: 93
increasing current slice: 94
increasing current slice: 95

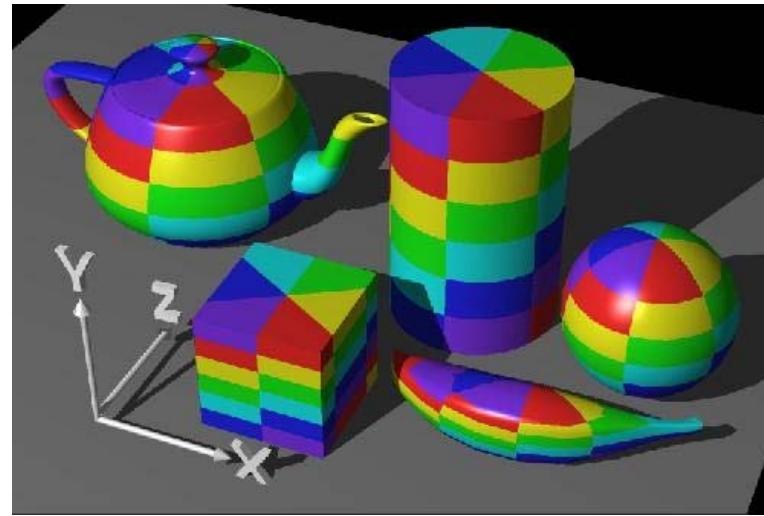
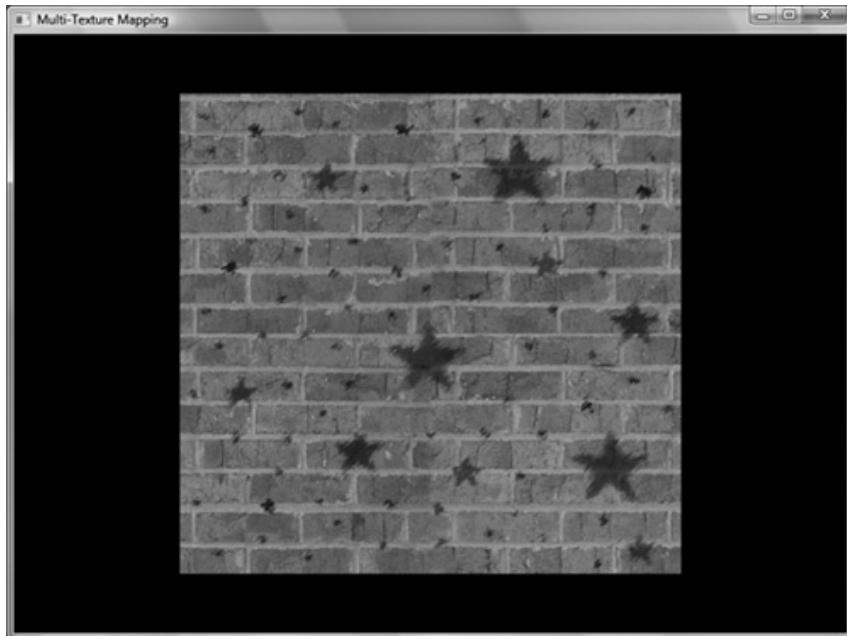
int printOpenGLError(char *file, int line)
{
    // Returns 1 if an OpenGL error occurred, 0 otherwise.
    GLenum glErr;
    int retCode = 0;

    glErr = glGetError();
    while (glErr != GL_NO_ERROR)
    {
        printf("glError in file %s @ line %d: %s\n", file, line, gluErrorName[glErr]);
        retCode = 1;
        glErr = glGetError();
    }
    return retCode;
}
```

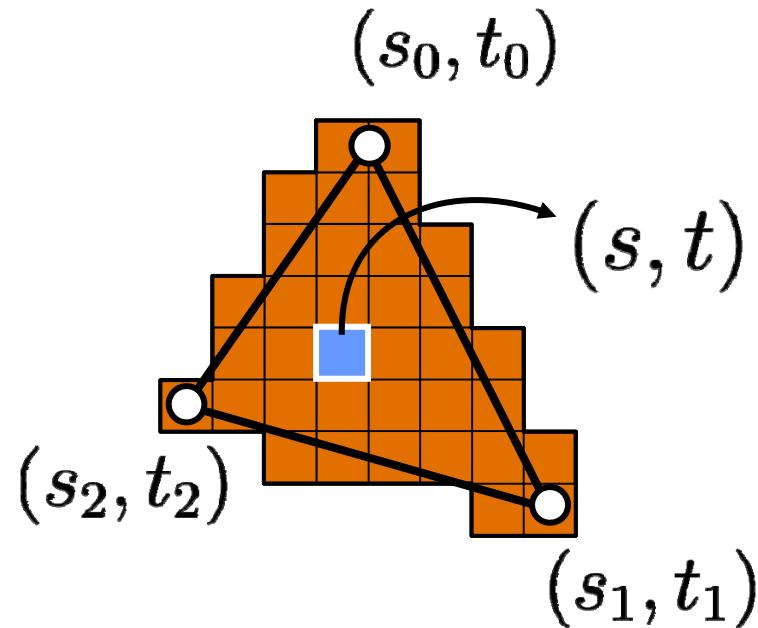
Programming Assignment #1 Example



Texture Mapping

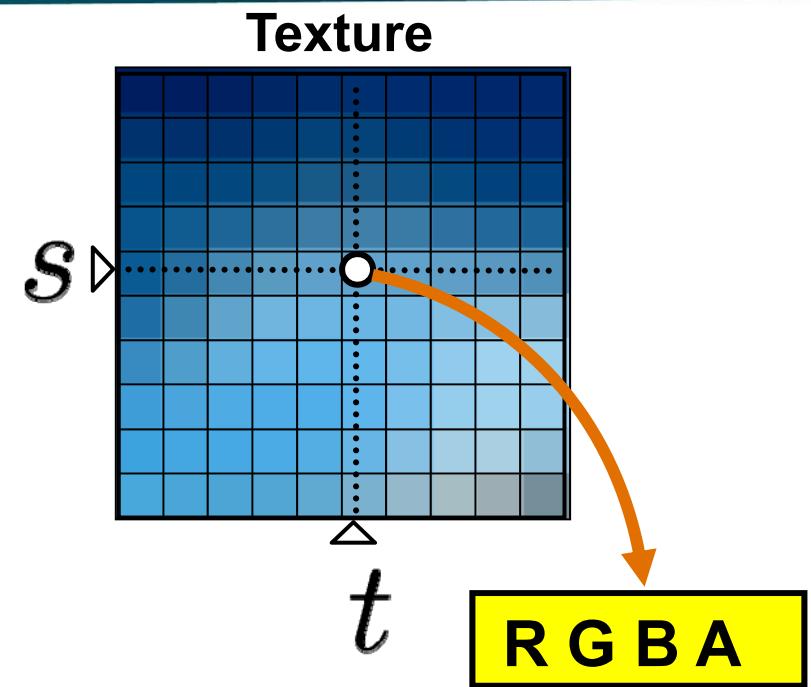


2D Texture Mapping



For each fragment:
interpolate the
texture coordinates
(barycentric)
Or:

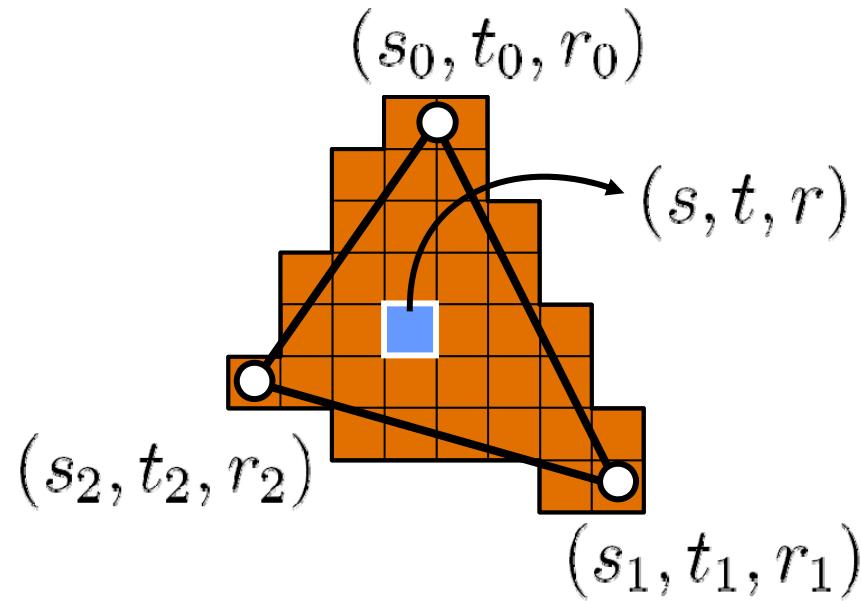
Use arbitrary, computed coordinates



Texture-Lookup:
interpolate the
texture data
(bi-linear)
Or:

Nearest-neighbor for “array lookup”

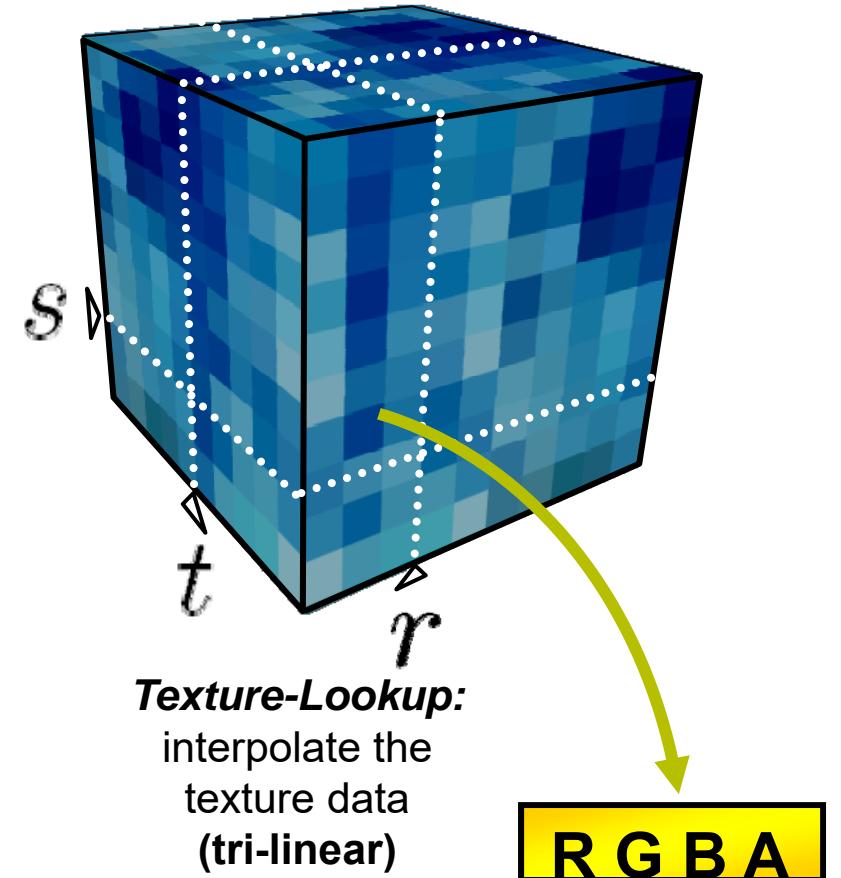
3D Texture Mapping



For each fragment:
interpolate the
texture coordinates
(barycentric)

Or:

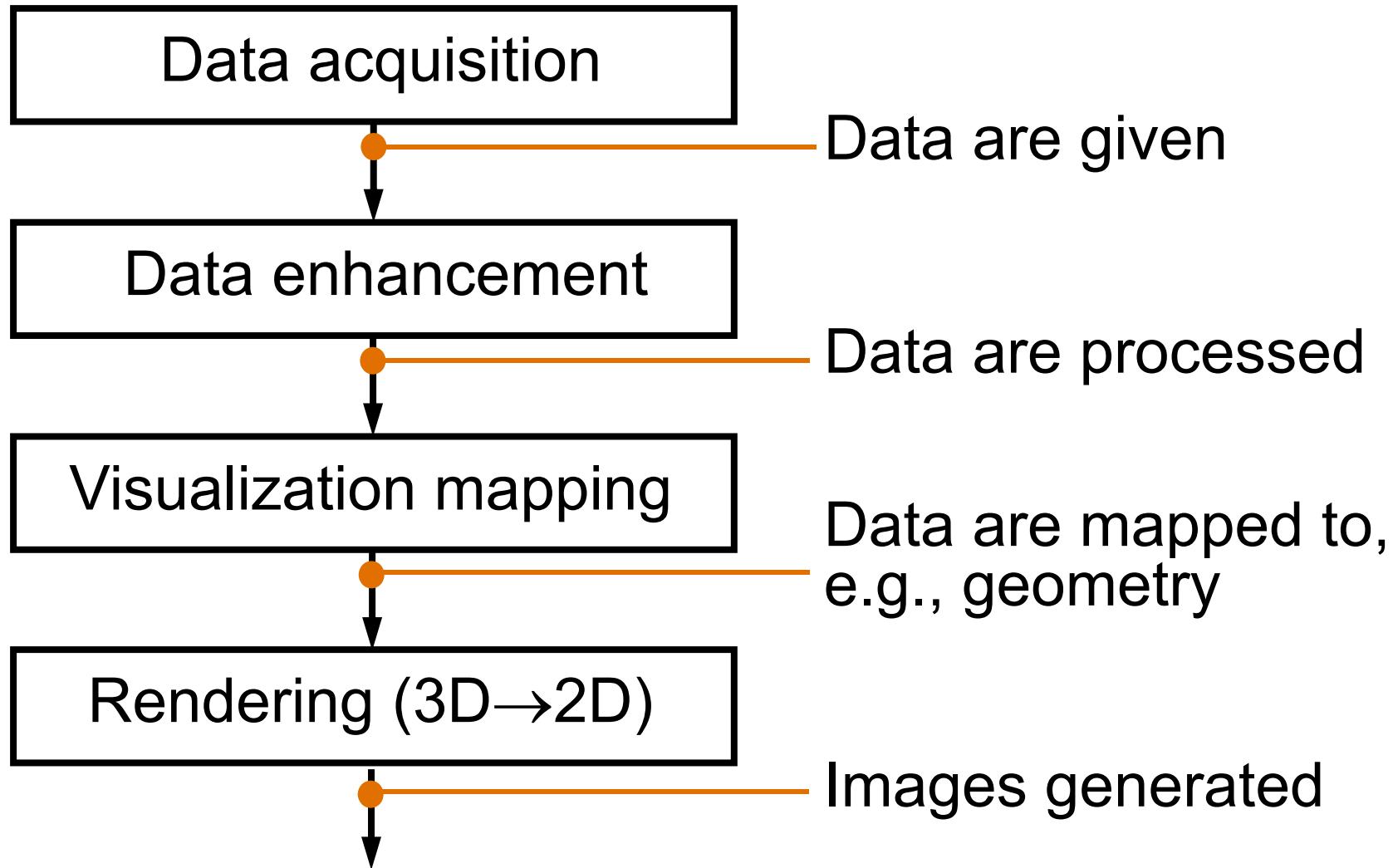
Use arbitrary, computed coordinates



Nearest-neighbor for “array lookup”

The Visualization Pipeline

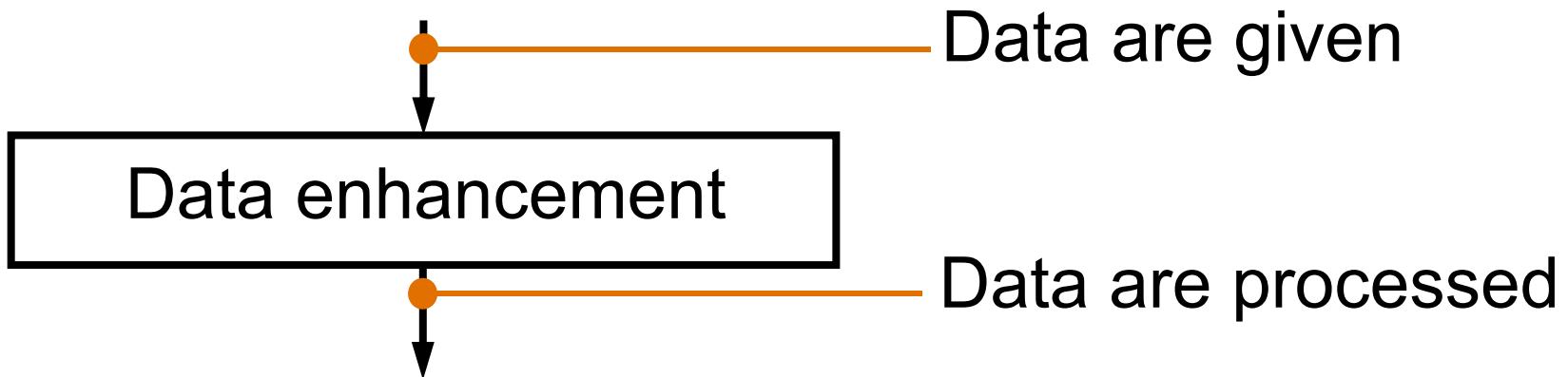
The Visualization Pipeline – Overview



The Visualization Pipeline – Stage 1

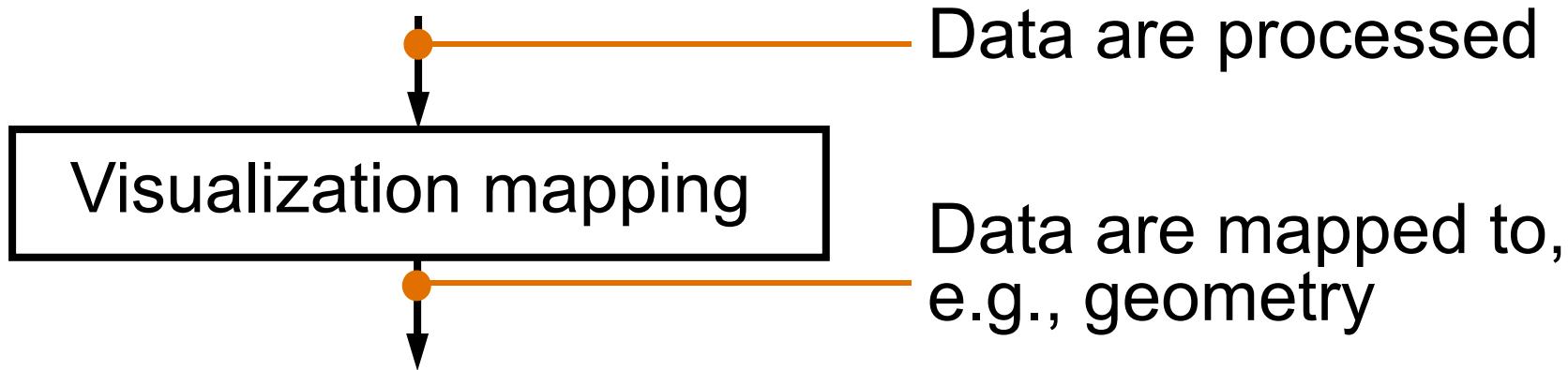


The Visualization Pipeline – Stage 2



- Filtering, e.g., smoothing (de-noising, ...)
- Resampling, e.g., on a different-resolution grid
- Data derivation, e.g., gradients, curvature
- Data interpolation, e.g., linear, cubic, ...

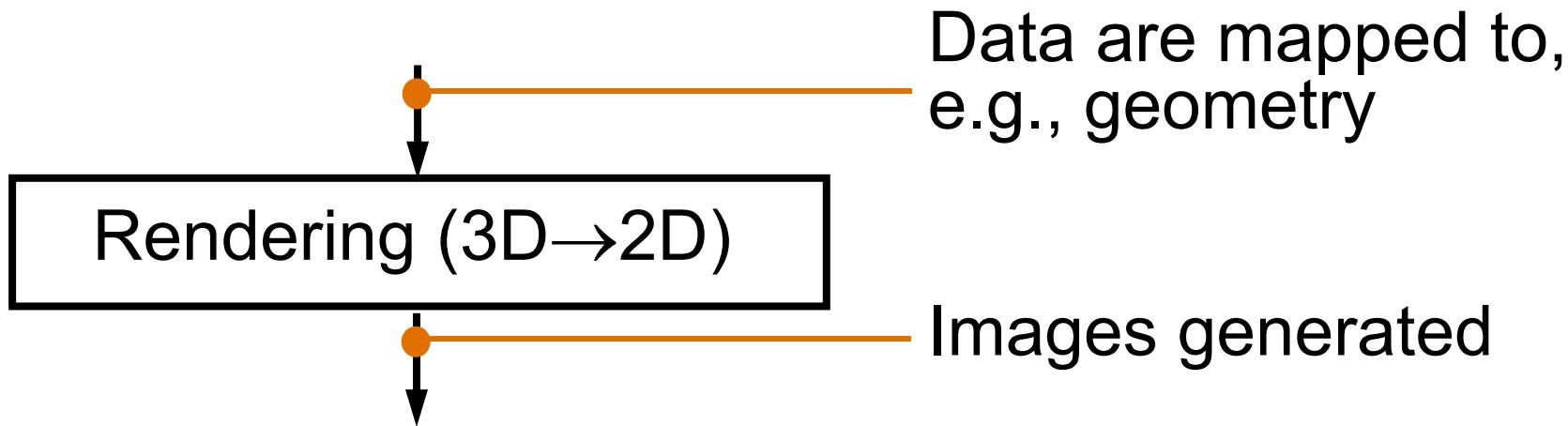
The Visualization Pipeline – Stage 3



Make data “renderable”

- Iso-surface calculation
- Glyphs, icons determination
- Graph-layout calculation
- Voxel attributes: color, transparency, ...

The Visualization Pipeline – Stage 4



Rendering = image generation with computer graphics

- Visibility calculation
- Illumination
- Compositing (combine transparent objects, ...)
- Animation

Data Representation

Data – General Information



Data:

- Focus of visualization,
everything is centered around the data
- Driving factor (besides user) in choice and attribution of the
visualization technique
- Important questions:
 - Where do the data “live” (**data space**)
 - **Type** of the data
 - Which **representation** makes sense
(secondary aspect)

Data Space



Where do the data “live”?

- Inherent spatial domain (**SciVis**):
 - 2D/3D data space given
 - examples: medical data, flow simulation data, GIS data, etc.
- No inherent spatial reference (**InfoVis**):
 - abstract data,
spatial embedding through visualization
 - example: data bases
- **Aspects**: dimensionality, domain, coordinates, region of influence of samples (local, global)

Data Type



What type of data?

- **Data types:**
 - Scalar = numerical value
(natural, integer, rational, real, complex numbers)
 - Non-numerical (categorical) values (e.g., blood type)
 - Multi-dimensional values, i.e., codomain (n-dim. vectors, second-order ($n \times n$) tensors, higher-order tensors, ...)
 - Multi-modal values (vectors of data with varying type [e.g., row in a table])
- **Aspects:** dimensionality, codomain (superset of range/image)

Data == Functions



Mathematical Functions

Associates every element of a set (e.g., X) with *exactly one* element of another set (e.g., Y)

Maps from domain (X) to codomain (Y)

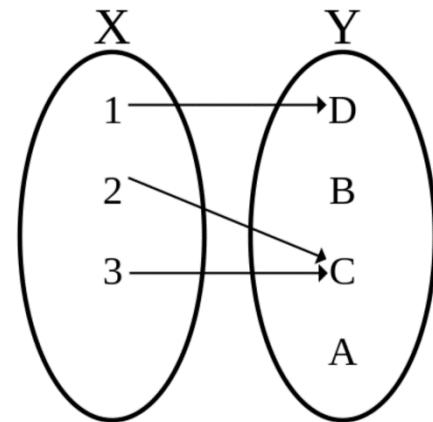
$$f: X \rightarrow Y$$

$$x \mapsto f(x)$$

Also important: *range/image*; *preimage*;
continuity, differentiability, dimensionality, ...

Graph of a function (mathematical definition):

$$G(f) := \{(x, f(x)) | x \in X\} \subset X \times Y$$



Mathematical Functions



Associates every element of a set (e.g., X) with *exactly one* element of another set (e.g., Y)

Maps from domain (X) to codomain (Y)

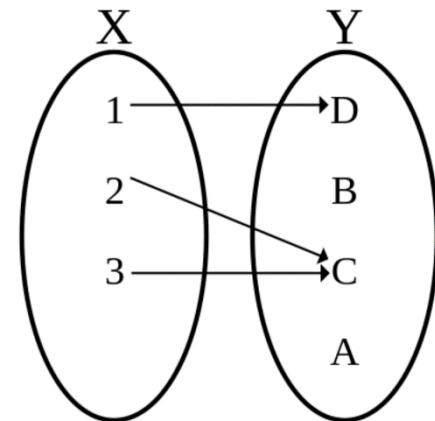
$$f: \mathbb{R}^n \rightarrow \mathbb{R}^m$$

$$x \mapsto f(x)$$

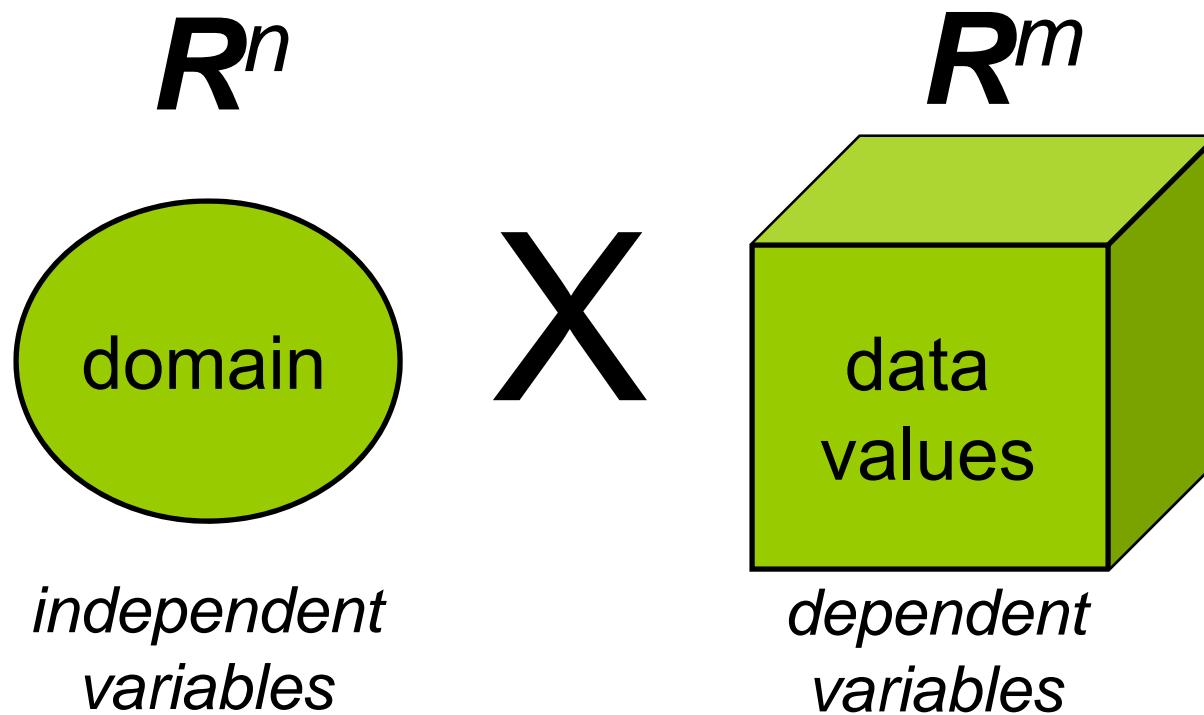
Also important: *range/image; preimage;*
continuity, differentiability, dimensionality, ...

Graph of a function (mathematical definition):

$$G(f) := \{(x, f(x)) | x \in \mathbb{R}^n\} \subset \mathbb{R}^n \times \mathbb{R}^m \simeq \mathbb{R}^{n+m}$$



Data Representation



scientific data $\subseteq R^{n+m}$



Example: Scalar Fields

2D scalar field

$$f: \mathbb{R}^2 \rightarrow \mathbb{R}$$

$$x \mapsto f(x)$$

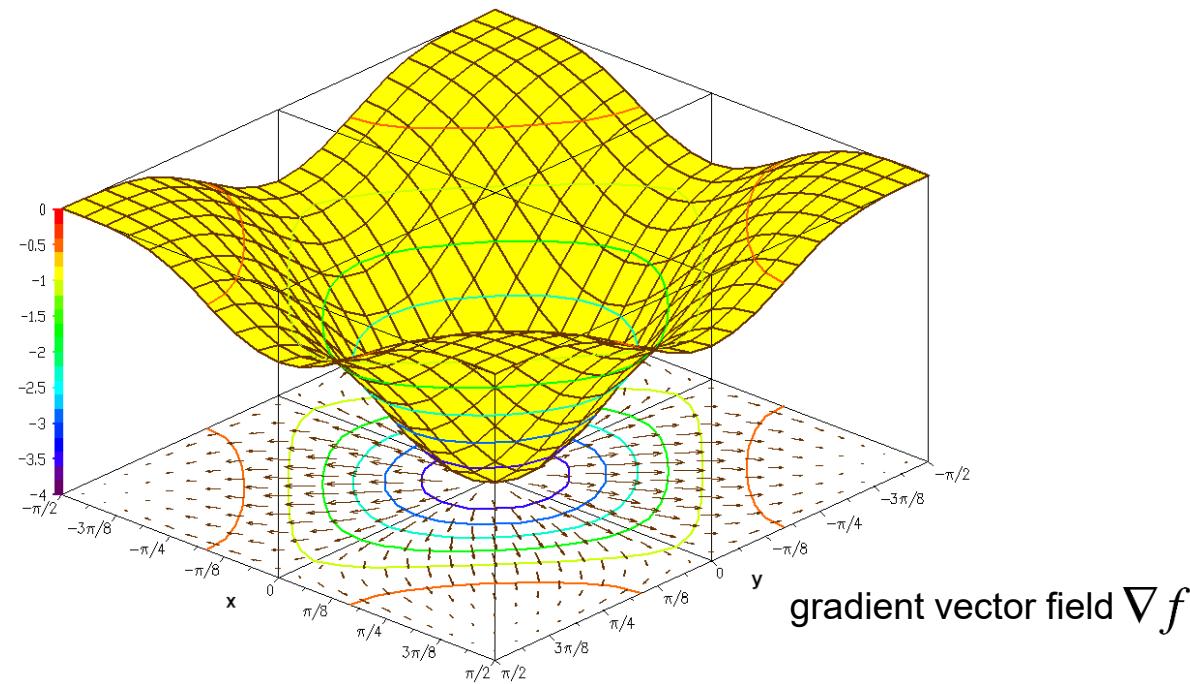
Graph: $G(f) := \{(x, f(x)) | x \in \mathbb{R}^2\} \subset \mathbb{R}^2 \times \mathbb{R} \simeq \mathbb{R}^3$

pre-image

$$S(c) := f^{-1}(c)$$

iso-contour

$$(\nabla f \neq 0)$$





Example: Scalar Fields

3D scalar field

$$f: \mathbb{R}^3 \rightarrow \mathbb{R}$$

$$x \mapsto f(x)$$

Graph: $G(f) := \{(x, f(x)) | x \in \mathbb{R}^3\} \subset \mathbb{R}^3 \times \mathbb{R} \simeq \mathbb{R}^4$

pre-image

$$S(c) := f^{-1}(c)$$

iso-surface

$$(\nabla f \neq 0)$$

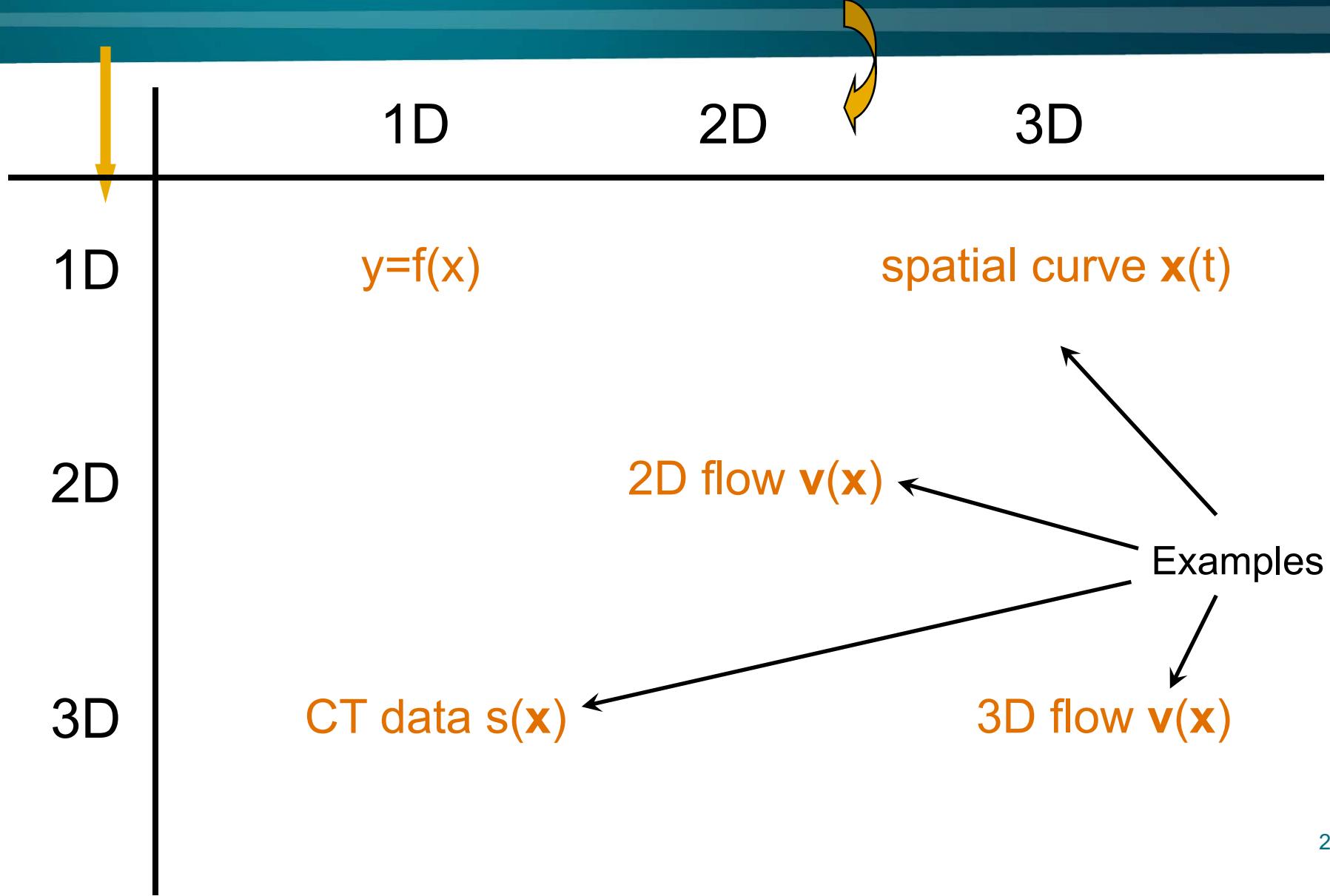
?

Visualization Examples



| data | description | visualization example |
|-----------------------|--|---|
| $N^1 \rightarrow R^1$ | value series | bar chart, pie chart, etc. |
| $R^1 \rightarrow R^1$ | scalar function over R | (line) graph |
| $R^2 \rightarrow R^1$ | scalar function over R^2 | 2D-height map in 3D, contour lines in 2D, false color map |
| $R^2 \rightarrow R^2$ | 2D vector field | hedgehog plot, LIC, streamlets, etc. |
| $R^3 \rightarrow R^1$ | scalar function over R^3 (3D densities) | iso-surfaces in 3D, volume rendering |
| $R^3 \rightarrow R^3$ | 3D vector field | streamlines/pathlines in 3D |

Data Space (Domain) vs. Data Type (Codomain)





Visualization Examples

data

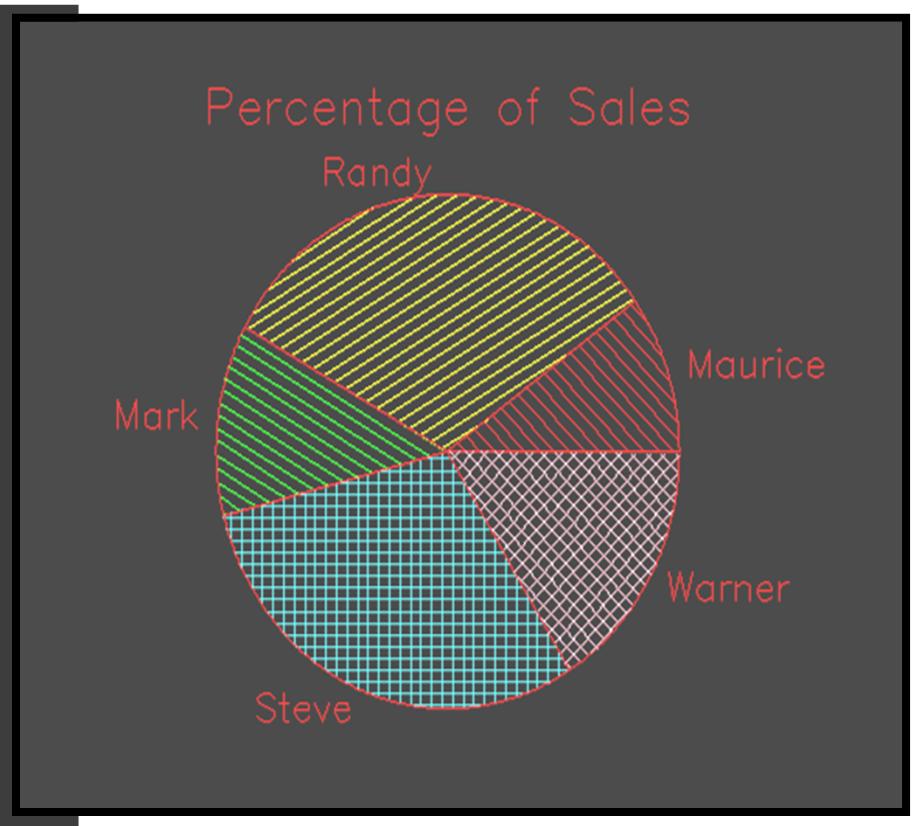
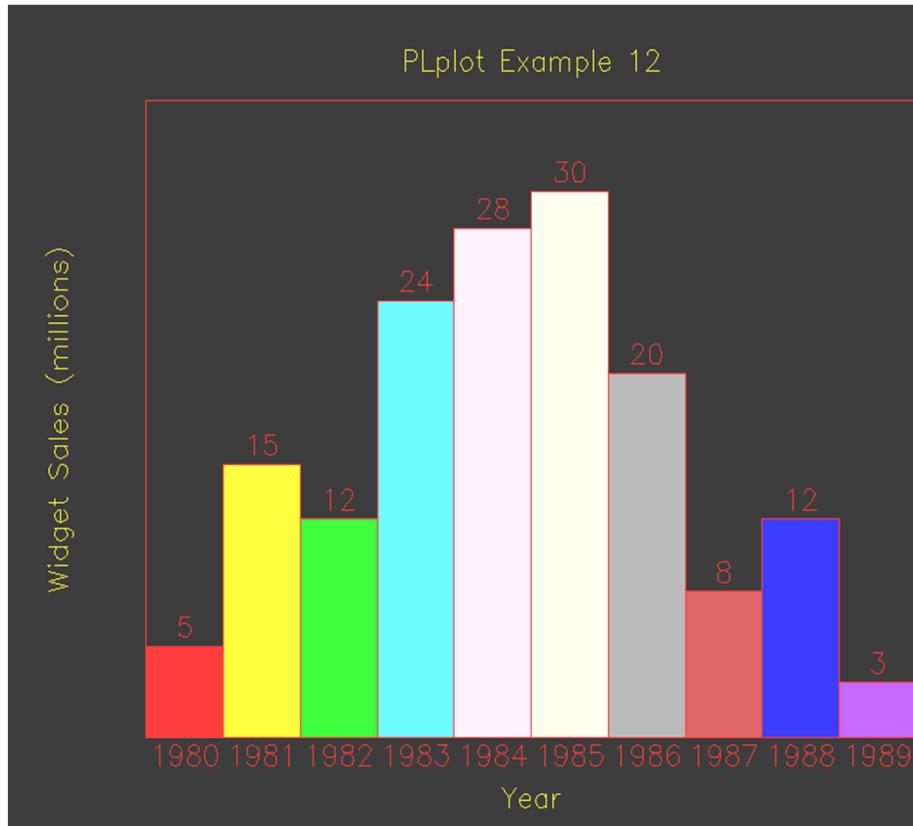
description

visualization example

$N^1 \rightarrow R^1$

value series

bar chart, pie chart, etc.



Visualization Examples



data

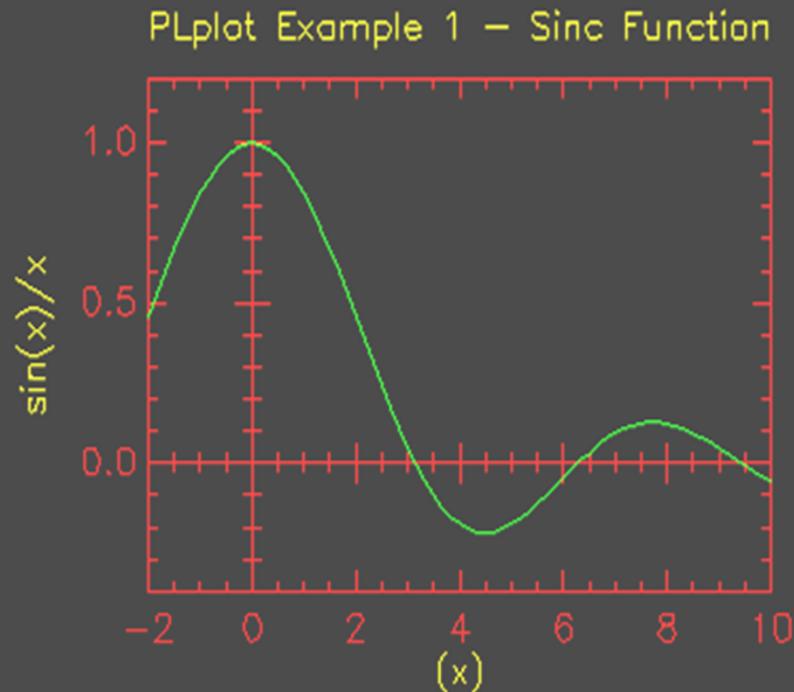
description

visualization example

$R^1 \rightarrow R^1$

function over R

(line) graph





Visualization Examples

data

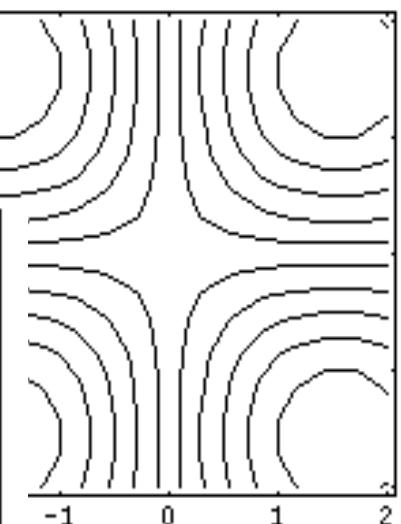
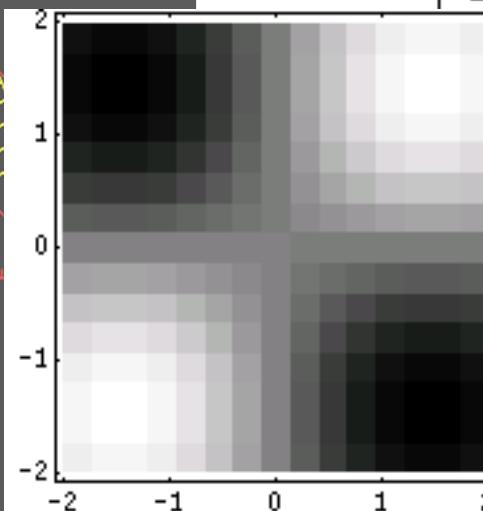
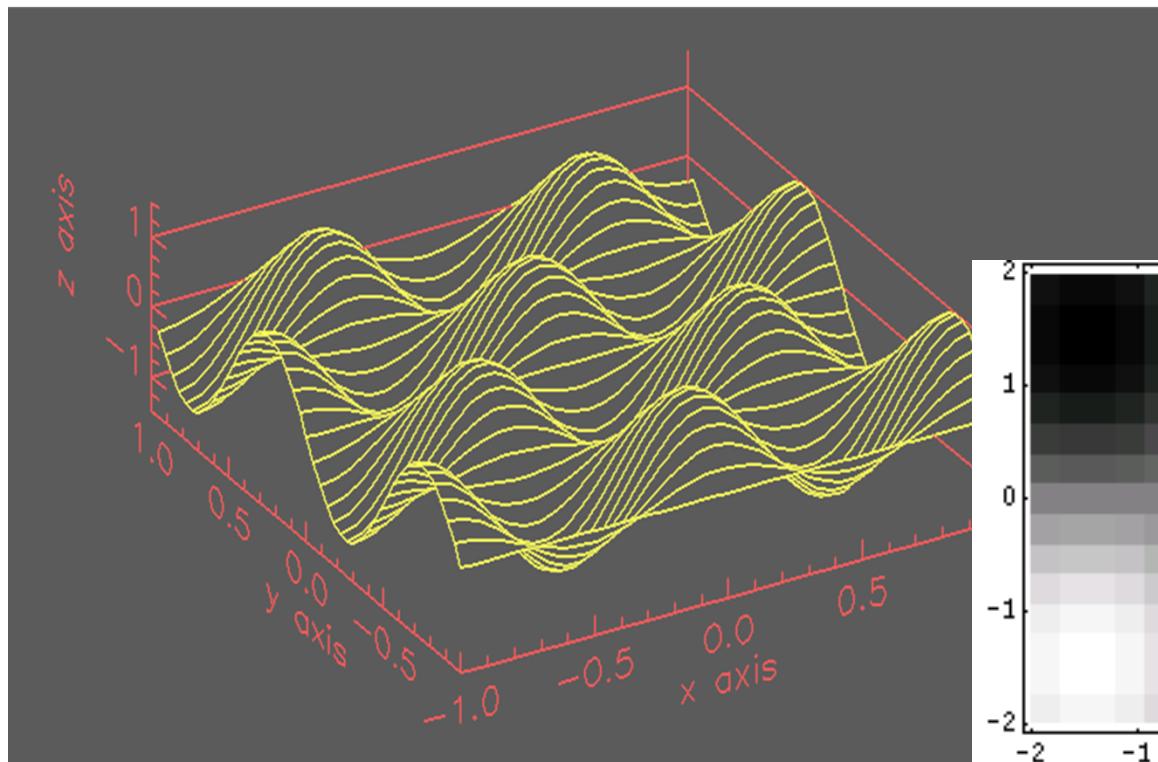
description

visualization example

$R^2 \rightarrow R^1$

function over R^2

2D-height map in 3D,
contour lines in 2D,
false colors (heat map)





Visualization Examples

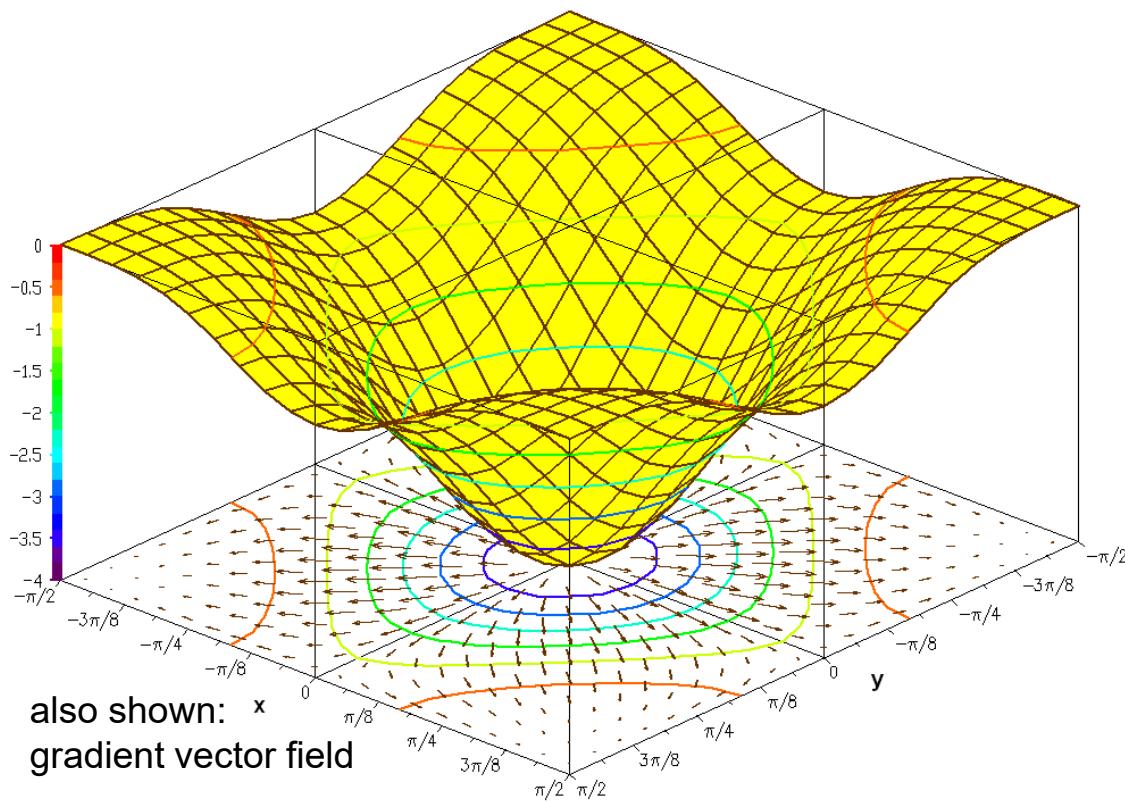
data

description

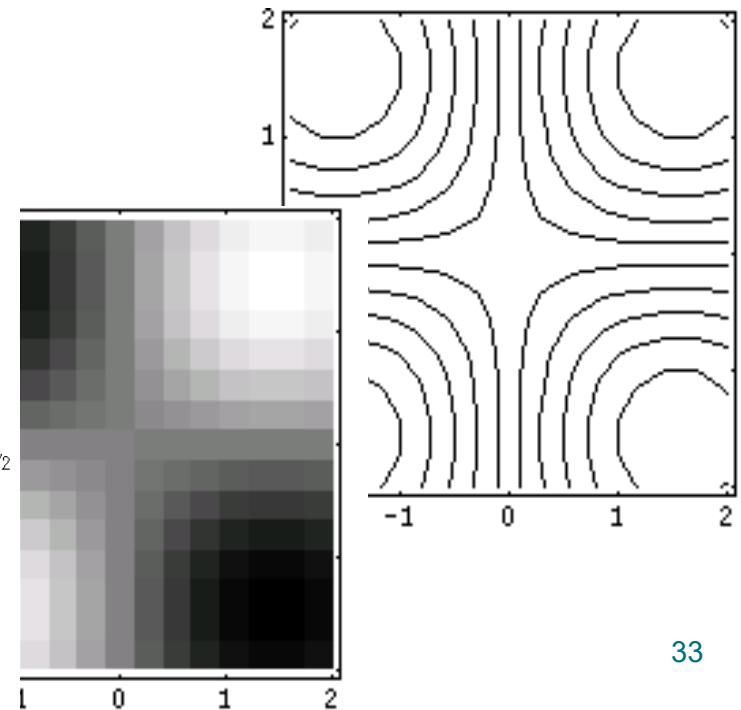
visualization example

$R^2 \rightarrow R^1$

function over R^2



2D-height map in 3D,
contour lines in 2D,
false colors (heat map)



Visualization Examples



data

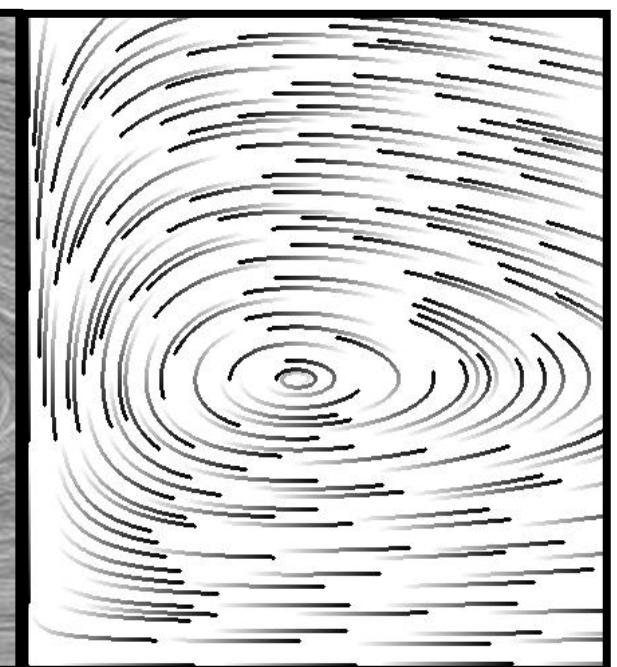
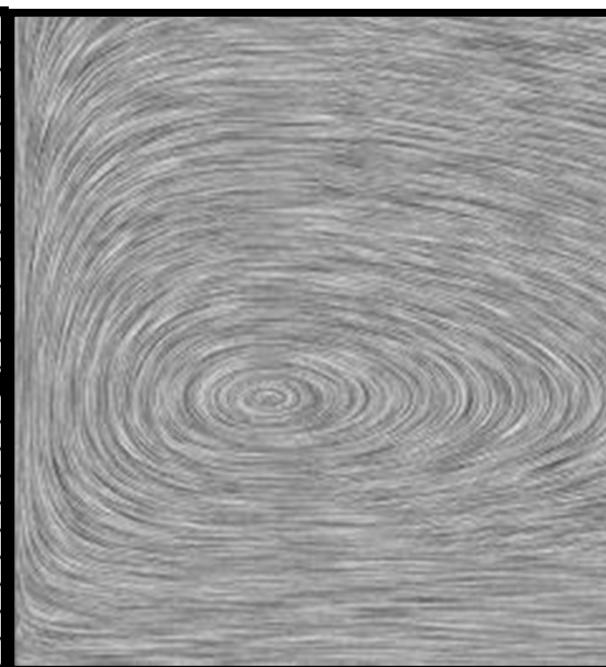
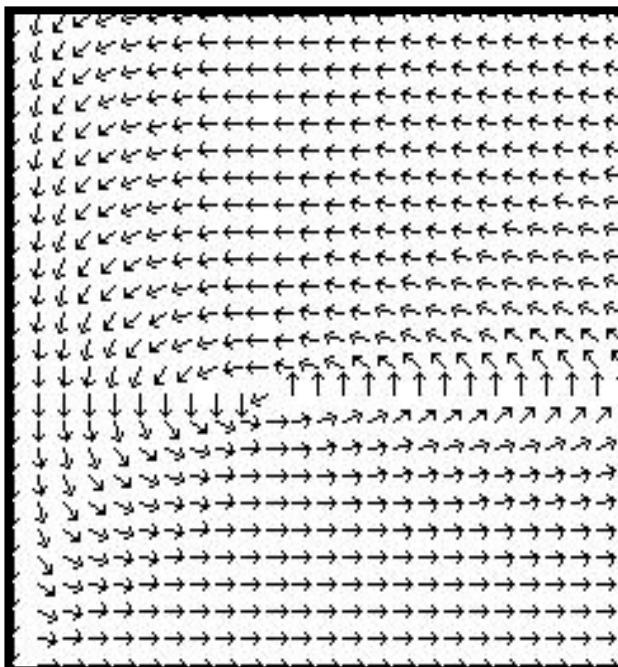
description

visualization example

$\mathbb{R}^2 \rightarrow \mathbb{R}^2$

2D-vector field

hedgehog plot, LIC,
streamlets, etc



Visualization Examples



data

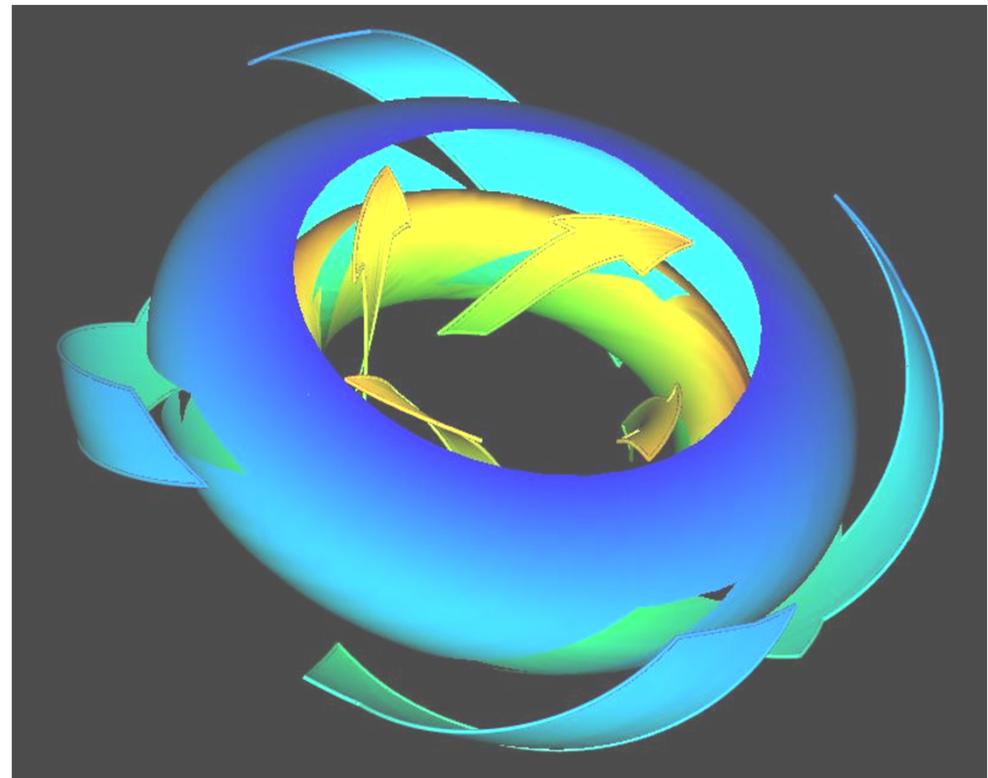
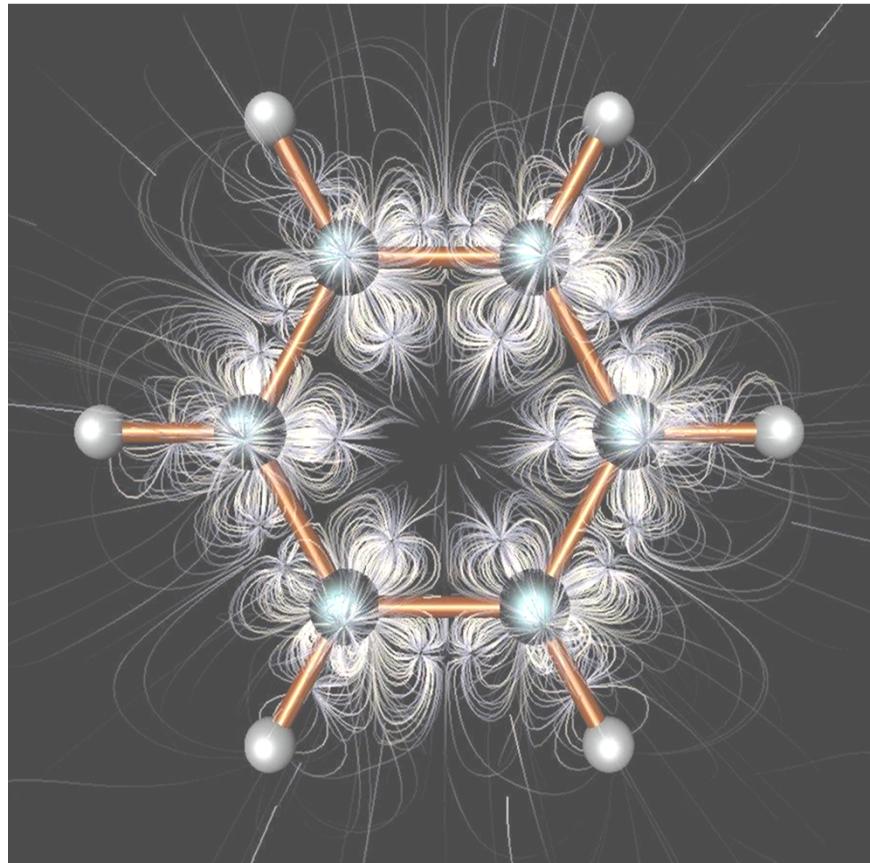
$\mathbb{R}^3 \rightarrow \mathbb{R}^3$

description

3D-flow

visualization example

streamlines,
streamsurfaces



Visualization Examples

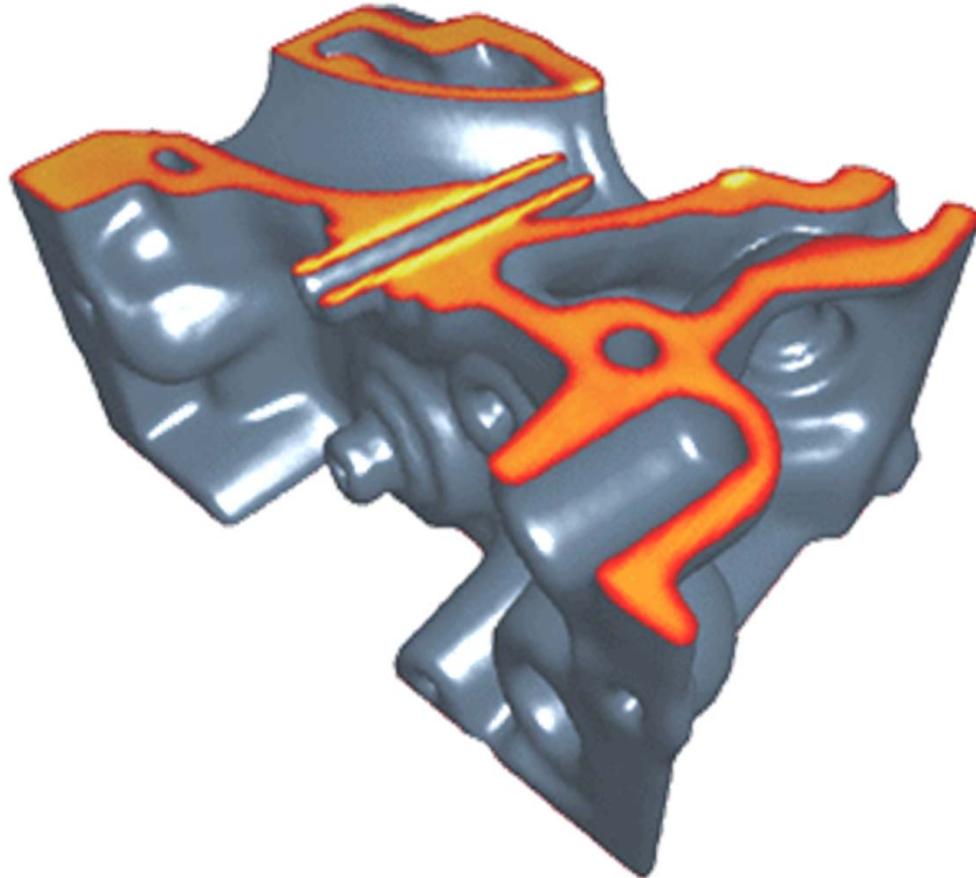


data

$$\mathbb{R}^3 \rightarrow \mathbb{R}^1$$

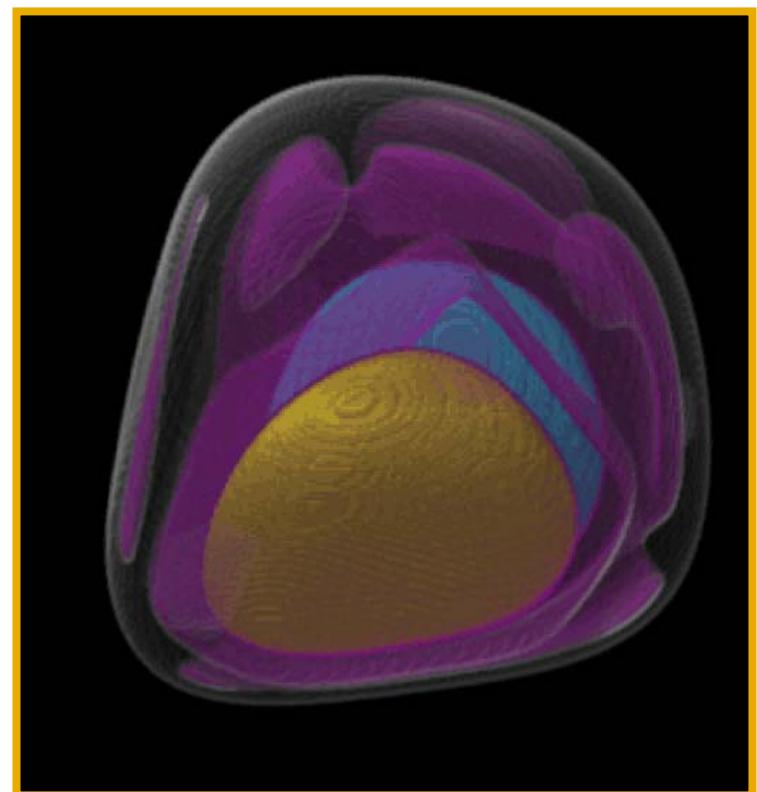
description

3D-densities



visualization example

iso-surfaces in 3D,
volume rendering



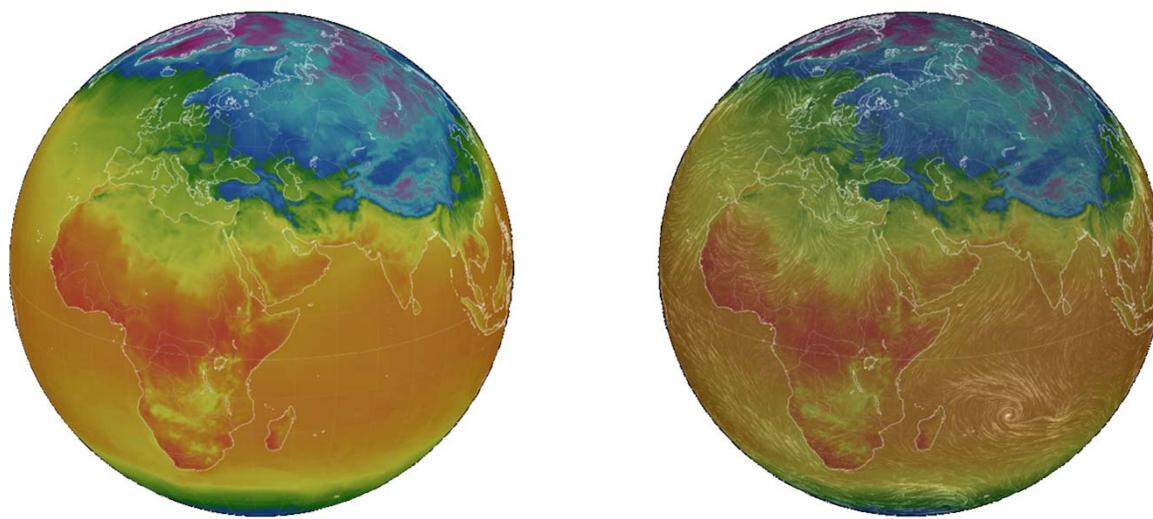


Domain Not Always Euclidean

Manifolds



- Scalar, vector, tensor fields on manifolds



Thank you.

Thanks for material

- Helwig Hauser
- Eduard Gröller
- Daniel Weiskopf
- Torsten Möller
- Ronny Peikert
- Philipp Muigg
- Christof Rezk-Salama