Report Scientific Visualization

Erik Bijl
Emilio Oldenziel

April 2018

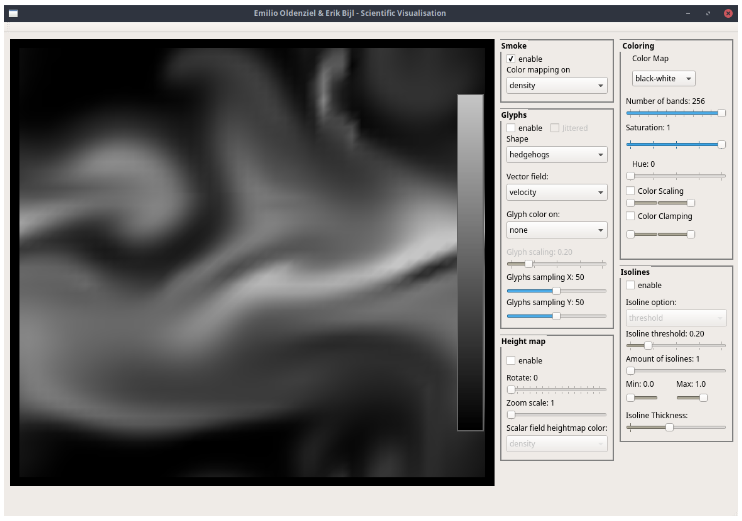
1 Introduction

For the practicals of the course Scientific Visualisation we created an user interface to demonstrate various visualisation techniques. We implemented these techniques for a continuous scientific data set in the form of fluid flow simulated by calculating the Navier-Stokes equation. The code for the implementation and the executable can be found on Github [1]

2 Setup

To easily set the visualisation parameters we chose to implement the provided source code in the Qt framework. Qt has OpenGL support and drag-and-drop; buttons, drop-boxes and sliders. Rewriting the code minimised the time spend on the user interface and focused us on the visualisation. Figure 1 shows the user interface. On the left the result of the data set visualisation is shown and on the right we can see the buttons and sliders that control the parameters of the visualisation. These are divided into groups of parameters for several visualisation techniques.

[1] https://github.com/EmilioOldenziel/ScientificVisualisation


Emilio Oldenziel & Erik Bijl - Scientific Visualisation

Smoke
- [✓] enable
- Color mapping on: density

Glyphs
- [ ] enable [ ] Jittered
- Shape: hedgehogs
- Vector field: velocity
- Glyph color on: none
- Glyph scaling: 0.20
- Glyphs sampling X: 50
- Glyphs sampling Y: 50

Height map
- [ ] enable
- Rotate: 0
- Zoom scale: 1
- Scalar field heightmap color: density

Coloring
- Color Map: black-white
- Number of bands: 256
- Saturation: 1
- Hue: 0
- [ ] Color Scaling
- [ ] Color Clamping

Isolines
- [ ] enable
- Isoline option: threshold
- Isoline threshold: 0.20
- Amount of isolines: 1
- Min: 0.0 Max: 1.0
- Isoline Thickness:

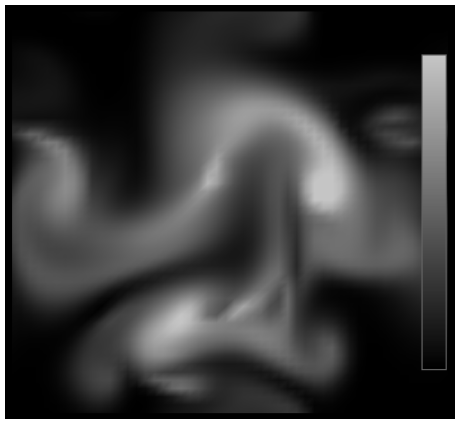
Figure 1: The user interface for the visualisation.

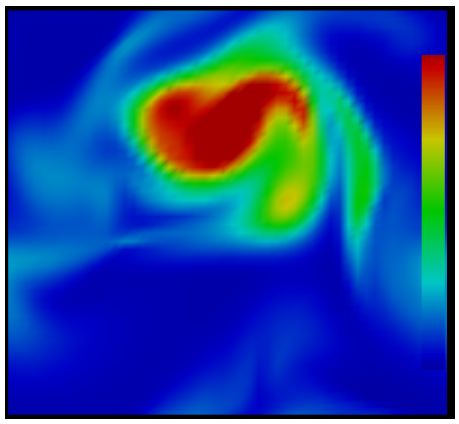
3 Coloring

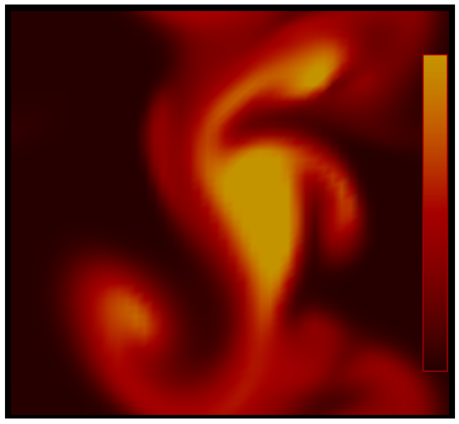
To visualise the scalar values of the data set: density (rho), force field magnitude $||f||$ and velocity field magnitude $||v||$ we implemented several colormaps.
To show the range of colors that are in the visualisation a colorbar is added to the right side of the visualisation.

3.1 Colormaps

The first two colormaps were already implemented in the provided source code. These two colormaps are the rainbow and the black-white colormap as can be seen in figure 2. The rainbow colormap is a intuitive colormap for humans and is commonly used. Still our eyes are attracted towards certain colors when using the rainbow colormap. The black-white colormap enables us to better see details of the visualisation in a natural manner (from black to white). Additionally we added a heat colormap which is shown in figure 2c. The heat colormaps is useful when one want to detect maxima of the visualisation. A zebra colormap (figure 2d) is implemented to enable a user to detect variation in the visualisation. The zebra colormap alternates between black and white for small value changes such that each value is mapped to black or white.


(a) Black-white colormap


(b) Rainbow colormap


(c) Heat colormap

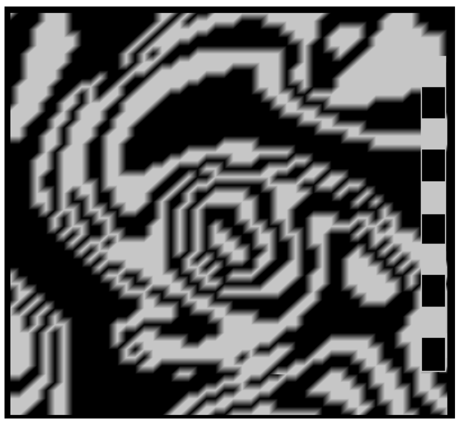
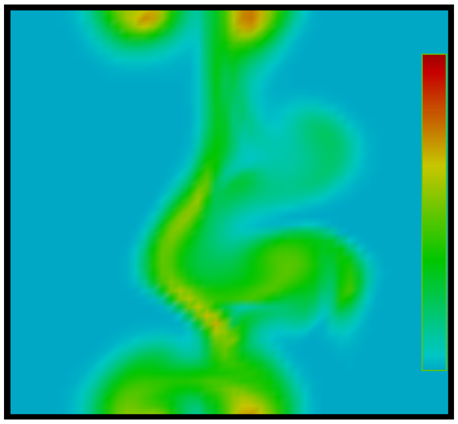
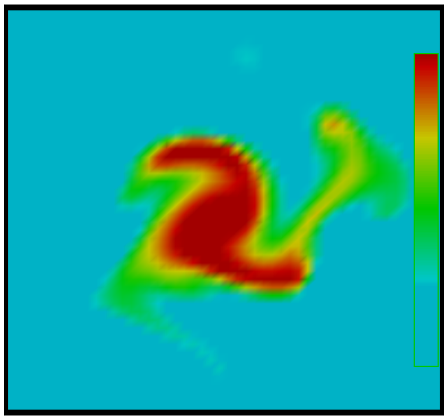

(d) Zebra colormap

Figure 2: Different colormaps

3.2 Scaling and clamping

Modification of the colormap is implemented by scaling and clamping. Scaling can be used to scale a part of the color range up or down, this can give a larger emphasis on a certain range of values. Clamping can 'cut off' a color range at the ends of the colorspace, this can be useful if we are not interested in these values. The results of scaling is shown in figure 5a and clamping in figure 5b.


(a) Color scaling


(b) Color clamping

3.3 HSV

The colormaps also have an option to be adjusted in the HSV space. In his space we can add addition rotation angle to the hue in the HSV space or change the saturation of the colors. Figure 4 shows a colormap where additional angle is added to the colorwheel in the HSV color space and the saturation is decreased.

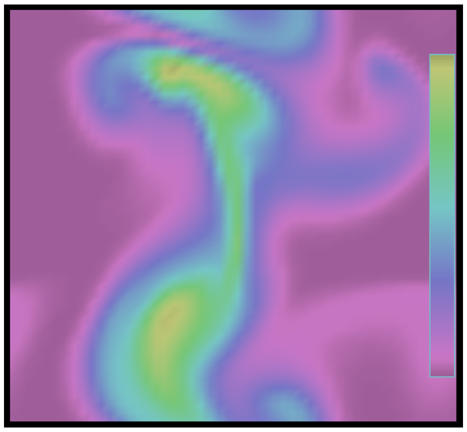


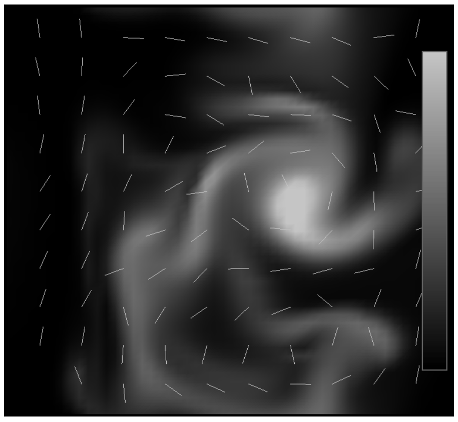
Figure 4: Colormap with lower saturation and adjusted hue

4 Glyph visualisation

In this step we implemented a set of glyph techniques. The glyphs are used to visualise the vector fields: fluid velocity v and the force field f. In this way we are able to visualise higher dimensional fields such as velocity and force. Glyphs show the position, direction and magnitude of a set of vectors. We will describe our implementation of glyphs by these three properties.

4.1 Position

The position of vector field can be seen as the place where a glyph is drawn. This position of the glyphs depends on the amount specified by the user. If a user wants to see few glyphs in detail, the sampling in x and y direction can be reduced. As a result the simulation shows few glyphs which can be thoroughly examined. Another scenario is indicating in which direction the fluid flows by an large amount of glyphs. This can be done by a large sampling amount and results in a cluttered simulation of glyphs. In this way we have enabled the user to set a desired amount of glyphs depending on the goal of the simulation. Two different amount of glyphs are illustrated in figure 5. The glyphs are calculated by linear interpolation of the vertices of the cell that in occurs in. This interpolation is a weighted sum over the four cell vertices.
A problem of vector glyph visualisation can be the regular pattern of sample points. The problem which occurs is called beating artifacts which shows patterns in our simulation which are not real. Beating artifacts can be solved by a method called jitter that adds a small deviation to the position of each cell. In our simulation an user can enable jitter if desired.


(a) Small sampling amount of glyphs

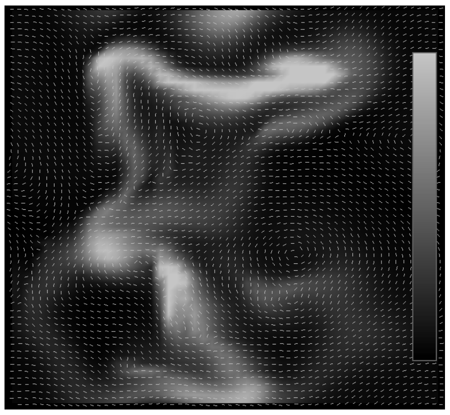
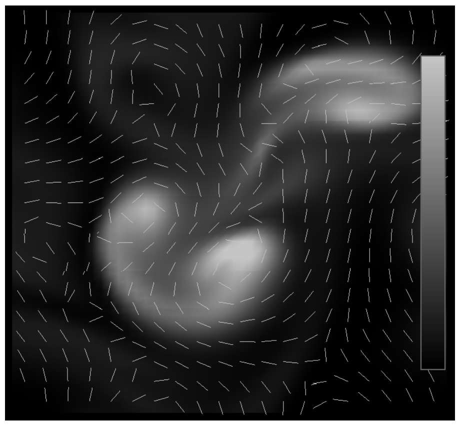

(b) Large sampling amount of glyphs

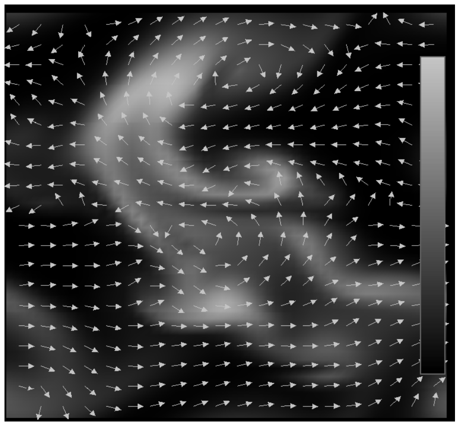
Figure 5: Different amounts of glyphs

4.2 Direction

Our simulation colors the glyphs to show the direction of a vector field. Each direction has its own color. Another method to show the direction of glyphs is using different shapes. These shapes are shown in figure 6 and we will describe them below:

- Hedgehog: this is the simplest form of glyphs. In each cell we draw a line that indicates the position, direction and magnitude of a set of vectors.
- Arrows: this is a more advanced form of glyphs. The advantage of using glyph arrows is that shows a signed direction which was not shown by hedgehog glyphs. The disadvantage is that it takes more space to render which makes the image more cluttered.
- Cones: a different shape that also shows the signed direction of glyphs.


(a) Hedgehogs


(b) Arrows

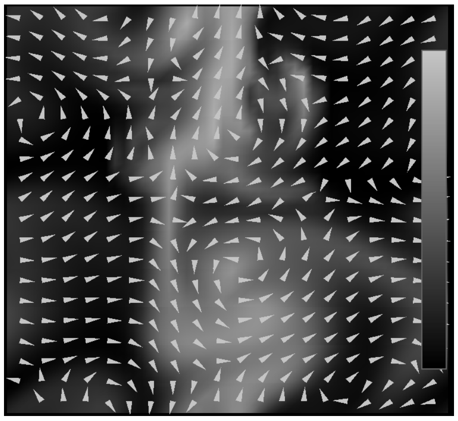
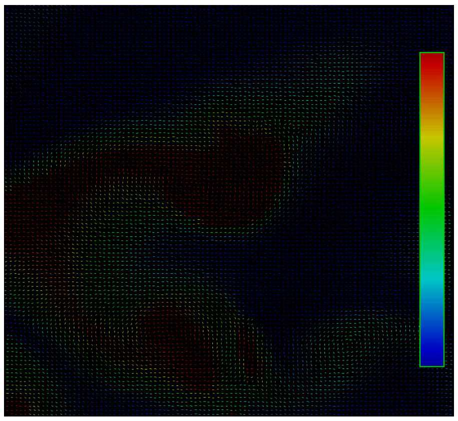

(c) Cones

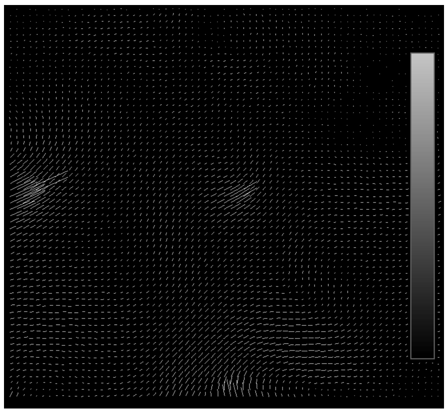
Figure 6: Different shapes of glyphs

4.3 Magnitude

In our simulation the magnitude of a glyph could be shown in two ways. The first manner is that of coloring the glyphs according to its magnitude shown in figure

7a. By doing this a high magnitude results in a high color value. This color value is then mapped to the colormap and the same properties for the smoke coloring are available. Another method is to scale the length of the glyphs according to its magnitude. This method is shown in figure 7b. Now a high magnitude results in a large glyphs. Our simulation also enables combination of these two settings.


(a) Color scaling


(b) Color clamping

5 Divergence

To visualise source and sinks in our dataset we implemented the divergence given a vector $v = (v_x, v_y, v_z)$ to scalar v. This calculation is defined by:

$$div\ \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \tag{1}$$

The result of the divergence implementation show the places where sources and sinks are located. In the simulation these are mostly close to each other.

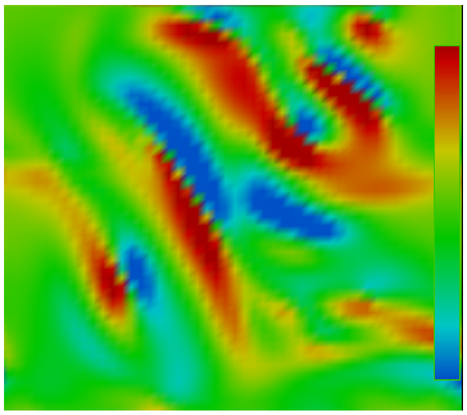
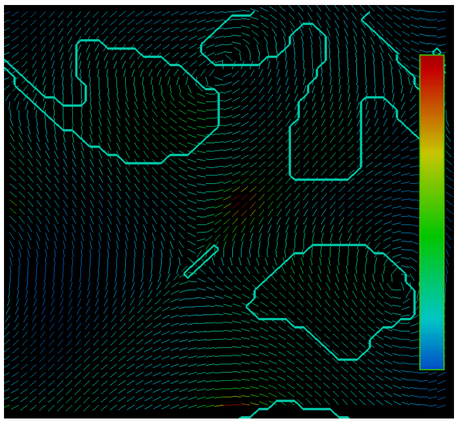


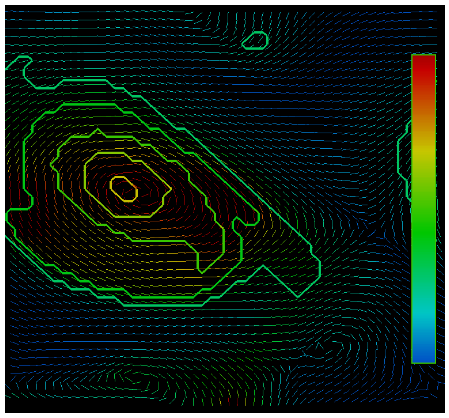
Figure 8: Divergence shows sources (red) and sinks (blue)

6 Isolines

Isolines can be used to denote areas where its values are above a certain threshold. To visualise these isolines we implemented the marching squares algorithm that draws a line in every cell in the simulation. The application has an option to choose one specific threshold as seen in figure figure 9a or to draw n isolines for threshold values in a specified range as shown in figure 9b.



(a) Isolines with treshold 0.5

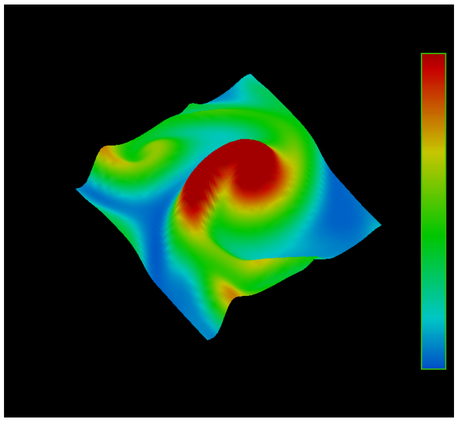


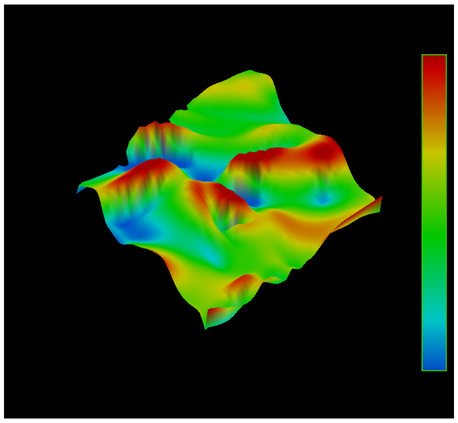
(b) 5 isolines between 0.3 and 0.8

7 Heightplots

To visualise an extra value we can add an extra dimension the visualisation by going to 3D, this can be performed by the implementation on height plots. We implemented a height plot where scalar values are represented by the height.

To do this we changed from triangles to quads to represent a cell. We added the option to select divergence as scalar value for the height as an extension, this results in 'steep hills' where a source is located and and 'deep valley' for sinks. b


(a) Heightplot for density


(b) Heightplot for divergence