Doppler Speckles - A Multi-Purpose Vectorfield Visualization Technique for Arbitrary Meshes

Werner Benger[1] and Georg Ritter[3] and Simon Su[4] and Dimitris E. Nikitopoulos[2] and Eamonn Walker[2] and Sumanta Acharya[2] and Somnath Roy[2] and Farid Harhad[1] and Wolfgang Kapferer[3]

[1]Center for Computation and Technology, Louisiana State University, USA

[2]Department for Mechanical Engineering, Louisiana State University, USA

[3]Institute for Astro- and Particle Physics, University of Innsbruck

[4]Princeton Institute for Computational Science and Engineering

`werner@cct.lsu.edu`, `georg.ritter@uibk.ac.at`, `simonsu@princeton.edu`, `meniki@me.lsu.edu`, `ewalke1@tigers.lsu.edu`, `acharya@me.lsu.edu`, `sroy13@tigers.lsu.edu`, `fharhad@cct.lsu.edu`, `wolfgang.e.kapferer@uibk.ac.at`

Abstract

We present a technique that allows a comprehensive view to volumetric vector fields on arbitrary mesh structures. The technique is inherently independent from the topological structure of the underlying grid by using vertex-based splats that are parameterized in color, shape and transparency by the vector field. It can be easily attached to volumes, surfaces or lines to depict the properties of a vector field, yielding more insight than standard techniques such vector arrows or streamlines. We demonstrate this technique upon datasets from computational fluid dynamics and astrophysics, comprising uniform grids, curvilinear multiblocks and particle systems.

1 Introduction

Visualization methods that emphasize variations in the vector fields - for instance for critical point analysis and feature extraction [Theisel et al., 2003] - require knowledge of the field's derivative, which to compute requires topological neighborhood information. Such is trivial for a uniform grid, but effortsome for grid structures build from multiple blocks, curvilinear coordinates or particle systems. While such specialized methods are required for detailed analysis of a particular case, they are well complemented with a generic approach giving a coarse overview of the entire structure first. Even with advanced vector field analysis tools available, the simple "draw vector as arrow" approach serve its purpose, since it is simple, intuitive, and applicable to any kind of data set.

The objective in our approach is to handle vector fields on arbitrary meshes, whereby we conceptually separate between the base space (the grid structure and its topological properties) and the fiber space (the properties of the fields given at each point), as demonstrated in [Benger, 2008]. The use of splatting techniques[Westover, 1990] is not new and has long been used a generic visualization technique for scalar and vector fields [Crawfis & Max, 1993]. The art lays in the way how these splats are drawn and parameterized by the vector field, especially in a volume. Approaches to apply line integral convolution techniques to three dimensions[Interrante & Grosch, 1997] are known to be difficult.

2 The Speckle Rendering Approach

2.1 Projected Vector Speckles

The approach here is to draw speckles via OpenGL point sprites, employing a fragment shader depicting the vector field properties. This method is very fast since there is no geometry to be constructed like for vector arrows. It is therefore suitable for large datasets consisting of many vertices and we used it to inspect 4 million points on a laptop GeForce 8600M GT (512MB GPU RAM) and 16 million points on Quadro FX 5600 (1.5MB GPU RAM) desktop workstation. The idea is to render the image of a point sprite representing the vector field, such as an arrow projected into the view plane. The image drawn on the point sprite must therefore be view-dependent and has to be computed by the fragment shader. Instead of drawing the outline of an arrow we chose to draw an elongated Gaussian spot, intentionally building on the work of [Crawfis & Max, 1993]. Using Gaussian spots are suitable for volume rendering and with additional texturing resemble Gabor Patterns that are suitable for tensor field visualization [Benger et al., 2006].

The main purpose of the vertex shader is to compute the projected vector, the size of the point sprite and to determine the overall color of the splat. While we could map any color to these monochrome splats, including a color-mapped scalar field, we will primarily make use of the component of the vector field in direction of the observer, as described later in 2.2. The vertex shader (Fig. 1) computes the projected vector, color and point sprite size, the (Fig. 2) computes texture coordinates for the point sprite as rotated by the vector projected into the view image plane.

2.2 The Doppler Effect Color Map

At cosmological scales where the velocity of objects is comparable to the speed of light, the colors under which an object is seen are shifted due to the Doppler effect. See [Weiskopf et al., 1999] for a review of this effect. [Kapferer & Riser, 2008]

```
uniform float MaxSize;       // maximal size of a splat
uniform float size;          // user-defined scaling parameter
uniform vec3 observer;       // location of the camera
uniform float velocityscale; // scaling factor for the vector field
uniform sampler1D colormap;  // color map as 1D texture

varying vec4 vector;         // projected unit vector for fragment shader
varying float r;            // magnitude of the vector field

void main(void)
{
      // Project vector field in image space
vec4  dest = gl_Vertex + vec4(gl_Normal,0);
      vector      = gl_ModelViewProjectionMatrix*dest;
      gl_Position = ftransform();
      vector     -= gl_Position;
      vector.w    = 0.0;

      // Compute size of the point sprite
float dist = length(observer-gl_Vertex.xyz);
float diameter = clamp(size/dist, 0.1, MaxSize);\n"
      gl_PointSize = diameter;

      // Compute color from view component of vector, scaled
      //to fit into the 1D colormap via the atan() function
float z = vector.z;
float zlen = atan(velocityscale*z)/(3.141592/2);
      gl_FrontColor = texture1D(colormap, 0.5 - 0.5*zlen);

      // normalize vector, and provide length as separate value
      r = length(vector);
      vector /= r;
}
```

Figure 1: Essence of the vertex shader in GLGS computing the projected vector, point sprite size and color. Note that the vector field is expected to be available through `gl_Normal`.

```
uniform float velocityscale, velosensitivity;
uniform sampler2D PSFTexture;
varying vec4 vector;
varying float r;

void main (void)
{
   if (r==0.0) discard;
   // (0,1) --> (-1,1)
vec2  splat_crd = 2.0*(gl_TexCoord[0].st - vec2(0.5,0.5));
float splat_radius2 = splat_crd.x*splat_crd.x + splat_crd.y*splat_crd.y;
float sin_p = vector.x,cos_p = uvector.y;
// compute point sprite coordinates rotated by the vector direction
vec2  splat_rot = vec2( splat_crd.x*cos_p + splat_crd.y*sin_p,
                       -splat_crd.x*sin_p + splat_crd.y*cos_p);

// stretch along vector x-direction, scaled by the vector magnitude, limited by atan
// scale factor will remain between [2,10]
   splat_rot.x *= 6 + 4*atan(velocityscale*r/velosensitivity)*velosensitivity / (3.141592/2);

float cos_th = vector.z, sin_th2 = 1.0-cos_th*cos_th;

// shorten splat by projection effect toward viewer plane
   splat_rot /= sin_th2;
// get intensity value for splat_rot 2D coordinates
float I = texture2D( PSFTexture, splat_rot);

// multiply by sin_th2 to make direct views to vectors invisible
   gl_FragColor.a = I*gl_Color.a*sin_th2;
   gl_FragColor.rgb = gl_Color.rgb;
}
```

Figure 2: Essence of the fragment shader in GLGS computing the projected vector, point sprite size and color. Actual code might provide more shaping functionality of the point sprite texture, support more thresholding features and support texture animation by modifying the `splat_rot.y` parameter.

used a color scheme resembling the Doppler effect to interactively colorize galaxies in astrophysical simulation, blueish depicting objects moving toward the camera, red depicting objects moving from the observer. This color scheme is intuitive once one is familiar with the Doppler effect and allows to read off projected velocities directly from the colors. Changing the view point will change the colors, such that the full three-dimensional structure of a vector field can be explored during interaction.

Here we employ a similar scheme to map the z component of the vector field to an rgb-tripel. This color map is parameterized by a central intensity value I_c, a line intensity value I_l and a deviation parameter d such that

$$\begin{array}{ccc} 0.0 & \longmapsto & (0,0,0) \\ d & \longmapsto & (I_l,0,0) \\ 0.5 & \longmapsto & (I_c,I_c,I_c) \\ 1-d & \longmapsto & (0,0,I_l) \\ 1.0 & \longmapsto & (0,0,0) \end{array} \tag{1}$$

Hereby the z-values of the projected vector field need to be transformed into the range $[0,1]$ before applying the color map. The vertex shader Fig. 1 ensures such via applying the arcus tangens. This color transfer function as described here allows for fine tuning to enhance certain properties. For instance, when setting the line intensity $I_l = 0$, then the speckles will be monochrome white with highest intensity when the vector field is parallel to the view plane, i.e. the projected z component is zero. This results in the same visual effect as illuminated streamlines [Zöckler et al., 1996].

3 Results

3.1 Couette Flow

The Couette Flow is a vector field solution describing a fluid between two counter-rotating cylinders at radii $r = 1$ and $r = 4$. The field is two-dimensional with axial symmetry and given by the analytic formula

$$\vec{v}(P) = \begin{cases} r \notin [1,4]: & \{0,0\} \\ r \in [1,4]: & \{-P_y(r-2)/r, P_x(r-2)/r\} \end{cases} \tag{2}$$

where $P = P_x, P_y$ is a spatial location with coordinates, $r = \sqrt{P_x^2 + P_y^2}$ is the radial distance of this point and $\vec{v}(P)$ the vector field at each point. The vector field vanishes for $r > 4$, $r < 1$ and at $r = 2$. For $r \in [1,2]$ it describes a counter-clockwise rotation, for $r \in [2,4]$ it describes a clockwise rotation.

We have chosen this vector field for analytic studies because of its simplicity. Fig. 3 shows a comparison when applying vector arrows, streamlines as seeded from a certain region of interest and colored by their length, and the colored speckles as described here. While the streamlines yield the most clear information, they require a proper choice of the seeding points, which is non-trivial in complex vector fields [Weinkauf et al., 2002]. In contrast, vector arrows and speckles show the structure of the entire dataset, whereby the colored speckles are visually superior. Note that the colors change depending on the view point (according to the of the relative velocity), as demonstrated in Fig. 4.

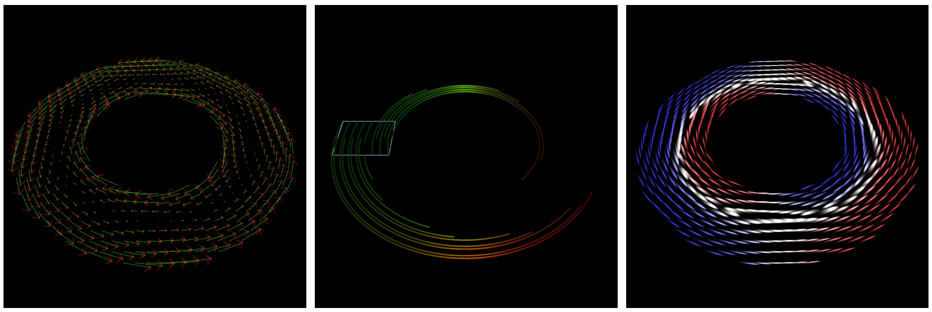

Figure 3: Couette Flow visualized via arrows, streamlines (the box in the left center indicates the location of the seeding points) and Doppler speckles.

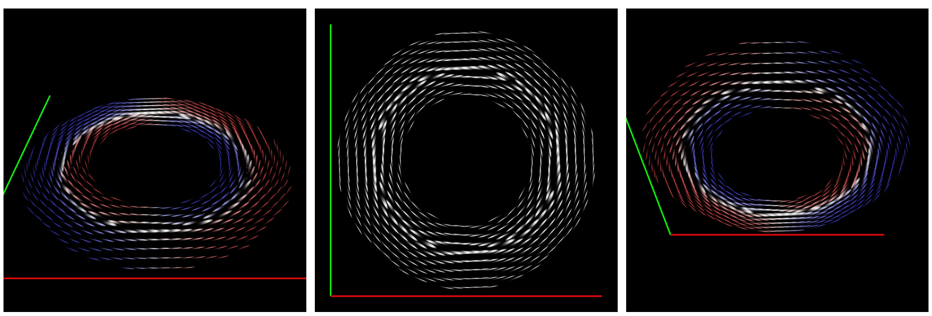

Figure 4: Doppler speckles of the Couette Flow as seen from different views.

3.2 Uniform Grid

We apply the described technique to a dataset stemming from a computational fluid dynamic simulation modeling a bubble rising in a micro channel. The data is given on a uniform grid of size $64 \times 64 \times 257$; we may render a Doppler speckle at each of the ca. 1 million data points, resulting in a volume rendering-like appearance as demonstrated in Fig. 5 left row. However, in contrast to classical volume rendering of a scalar field, the visible structures and colors will vary with the view point (upper row, lower row). The dominant direction of the fluid flow is prominently visible as well regions where the flow is reversed. These regions can be investigated in detail using streamlines; the Doppler speckles can now be attached to the vertices of the streamlines. We achieve a rendering performance of about 5fps using a GeForce 8600M GT for the entire volumetric data set.

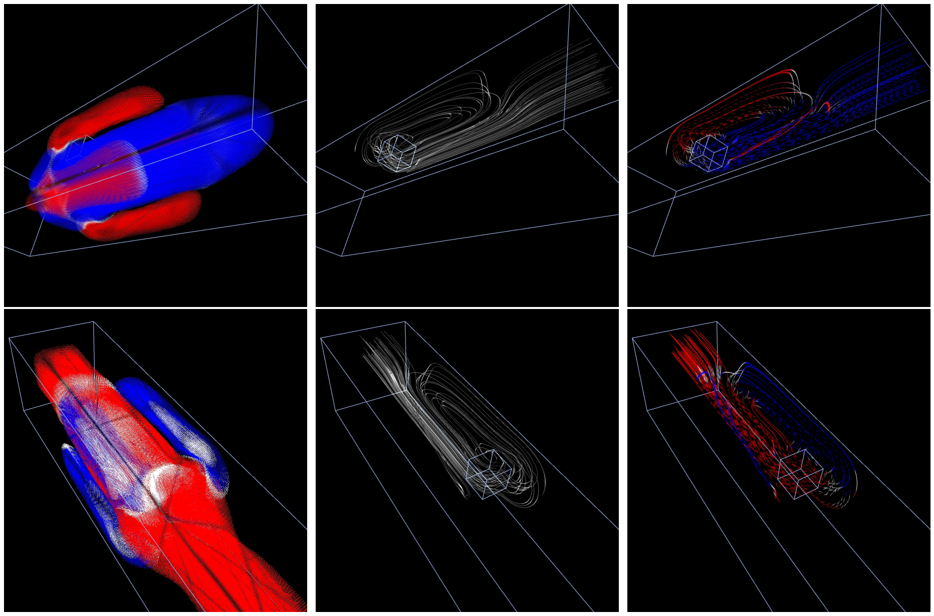

Figure 5: Flow field of a bubble rising in a micro channel, depicting Doppler speckles in the entire volume (left), streamlines (center) and speckles applied to stream lines. Lower row shows appearance from opposite view direction.

3.3 Curvilinear Multiblock

The CFD simulation of a stir tank produced a dataset consisting of 2088 blocks of the sizes varying from 9^3 to 32^3, resulting in ca. 3mio vertices with explicit coordinates (curvilinear grid). We have successfully computed streamlines in this complex dataset [Benger et al., 2009], which is now complemented by this vector field volume rendering technique using Doppler speckles. A certain difficulty in computing streamlines is finding points of interest; the Doppler speckle technique may serve the purpose of investigating the entire dataset at once, as demonstrated in Fig. 6. We achieve a rendering performance of about 0.3fps using a GeForce 8600M GT.

3.4 Particle Systems

The cosmological simulation by [Kapferer et al., 2005] is based on Gadget 2 [Springel, 2005] and implements an N-body gravitational model coupled with smoothed particle hydrodynamics (SPH). It is given as a particle data set without explicit neighborhood information. The Doppler speckle technique can be directly applied to this volumetric data set, providing information about the velocity field around galaxies formed in this process. Color reveals velocity orthogonal to the view plane, the structure of the speckles depicts velocity in the

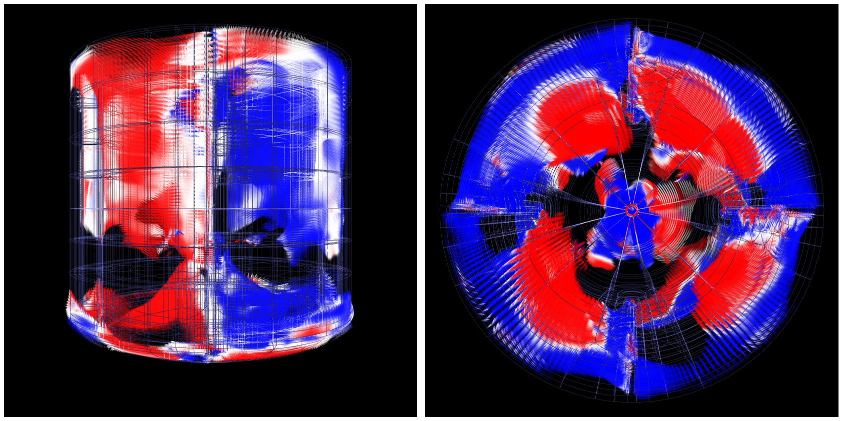

Figure 6: Flow field of a stir tank visualized using the Doppler speckles; we employed various threshold parameters to suppress regions of high velocity, therefore enhancing boundaries of changing flow direction.

view plane itself Fig. 7. For a dataset of 4 million points we achieve a rendering performance of about 2fps using a GeForce 8600M GT.

4 Conclusion

An universal algorithm to visualize volumetric vector fields on arbitrary mesh structures has been presented. It provides a rich parameter space to emphasize particular regions of a vector field in a volume, and can also be applied to one-dimensional base spaces such as lines, two-dimensional base spaces such as surfaces and three-dimensional volumes, yielding a novel approach for the volume rendering of vector fields.

5 Acknowledgements

This research employed resources of the Center for Computation & Technology at Louisiana State University, which is supported by funding from the Louisiana legislature's Information Technology Initiative. Portions of this work were supported by NSF/EPSCoR Award No. EPS-0701491 (CyberTools).

References

[Benger, 2008] Benger, W. (2008). Colliding galaxies, rotating neutron stars and merging black holes - visualising high dimensional data sets on arbitrary meshes. *New Journal of Physics*, 10. URL: http://stacks.iop.org/1367-2630/10/125004.

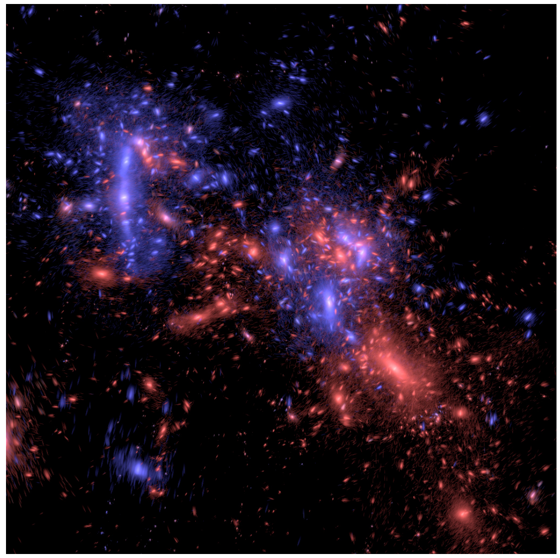

Figure 7: Velocity field of galaxies depicted by Doppler shift and geometric shape. Rendering of 4 million vertices.

[Benger et al., 2006] Benger, W., Bartsch, H., Hege, H.-C., Kitzler, H., Shumilina, A., & Werner, A. (2006). Visualizing Neuronal Structures in the Human Brain via Diffusion Tensor MRI. *International Journal of Neuroscience*, 116(4), pp. 461–514.

[Benger et al., 2009] Benger, W., Ritter, M., Acharya, S., Roy, S., & Jijao, F. (2009). Fiberbundle-based visualization of a stir tank fluid. In *WSCG 2009, Plzen*.

[Crawfis & Max, 1993] Crawfis, R. A. & Max, N. (1993). Textured splats for 3d scalar and vector field visualization. In Nielson & Bergeron (Eds.), *Visualization '93, San Jose* (pp. 261–266).: IEEE CS Press.

[Interrante & Grosch, 1997] Interrante, V. & Grosch, C. (1997). Strategies for effectively visualizing 3d flow with volume lic. In *Visualization '97* (pp. 421–424).: ACM Press.

[Kapferer et al., 2005] Kapferer, W., Knapp, A., Schindler, S., Kimeswenger, S., & van Kampen, E. (2005). Star formation rates and mass distributions in interacting galaxies. `http://arxiv.org/abs/astro-ph/0503559v1`.

[Kapferer & Riser, 2008] Kapferer, W. & Riser, T. (2008). Visualization needs and techniques for astrophysical simulations. *New Journal of Physics*, 10(12), 125008 (15pp). URL: `http://stacks.iop.org/1367-2630/10/125008`.

[Springel, 2005] Springel, V. (2005). The cosmological simulation code gadget-2. *Mon.Not.Roy.Astron.Soc.*, 364, 1105–1134. URL: `http://www.mpa-garching.mpg.de/gadget`.

[Theisel et al., 2003] Theisel, H., Weinkauf, T., Hege, H.-C., & Seidel, H.-P. (2003). Saddle connectors - an approach to visualizing the topological skeleton of complex 3d vector fields. In G. Turk, J. J. van Wijk, & R. Moorhead (Eds.), *Proc. IEEE Visualization 2003* Seattle, U.S.A.

[Weinkauf et al., 2002] Weinkauf, T., Hege, H.-C., Noack, B., Schlegel, M., & Dillmann, A. (2002). Flow around a backward-facing step. Winning Entry from the 20th Annual Gallery of Fluid Motion exhibit, held at the 55th Annual Meeting of the American Physical Society, Division of Fluid Dynamics, Dallas, November 2002. URL: `http://www.zib.de/weinkauf/`.

[Weiskopf et al., 1999] Weiskopf, D., Kraus, U., & Ruder, H. (1999). Searchlight and doppler effects in the visualization of special relativity: a corrected derivation of the transformation of radiance. *ACM Transactions on Graphics*, 18(3), 278–292. URL: `http://www.tat.physik.uni-tuebingen.de/~weiskopf/publications/acmtog99.pdf`.

[Westover, 1990] Westover, L. (1990). Footprint evaluation for volume rendering. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques* (pp. 367–376).: ACM Press, NY, USA.

[Zöckler et al., 1996] Zöckler, M., Stalling, D., & Hege, H.-C. (1996). Interactive visualization of 3d-vector fields using illuminated streamlines. In *Visualization '96* (pp. 107–113). URL: `http://www.zib.de/visual/papers/islVis96/`.