A Case Study in Selective Visualization of Unsteady 3D Flow

Dirk Bauer ETH Zürich
Ronald Peikert ETH Zürich
Mie Sato ETH Zürich
Mirjam Sick VA Tech Hydro Zürich

Abstract

In this case study, we explore techniques for the purpose of visualizing isolated flow structures in time-dependent data. Our primary industrial application is the visualization of the vortex rope, a rotating helical structure which builds up in the draft tube of a water turbine. The vortex rope can be characterized by high values of normalized helicity, which is a scalar field derived from the given CFD velocity data. In two related applications, the goal is to visualize the cavitation regions near the runner blades of a Kaplan turbine and a water pump, respectively. Again, the flow structure of interest can be defined by a scalar field, namely by low pressure values. We propose a particle seeding scheme based on quasi-random numbers, which minimizes visual artifacts such as clusters or patterns. By constraining the visualization to a region of interest, occlusion problems are reduced and storage efficiency is gained.

CR Categories and Subject Descriptors: I.3.8 [Computer Graphics]: Applications; I.4.7 [Image Processing and Computer Vision]: Feature Measurement - Feature Representation; J.2 [Applications]: Physical Sciences and Engineering - Engineering.

Additional Keywords: Flow Visualization, Feature Extraction, Particle Tracing

1 INTRODUCTION

Displaying moving particles is a standard technique for the visualization of time-dependent 3D vector fields. Nevertheless, for that type of data, it is hard to find better techniques. Moving streamlines are not physically meaningful, while path lines should be reserved to static visualizations. Streak lines are an adequate technique, but by discretizing them into a series of particles, some additional information can be conveyed. Particles also have the advantage of a small glyph size which minimizes occlusion in projected views. And finally, particle visualizations compare naturally to some types of flow experiments. The hydraulic CFD simulations which led to this case study are often accompanied by flow experiments where the flow is measured and visually examined in scaled models. In such experiments, cavitation bubbles can be observed (even in rotating machine parts if viewed under stroboscopic lighting).

While steady flows can be explored by scanning through the data domain, e.g. by taking slices or by moving streamline seed points, this is not an option for time-dependent data. But then, visualization has to be sparse, i.e. it can not simultaneously depict the flow everywhere in the data domain. If a certain level of detail is expected, visualization has to be constrained to a region, and as a consequence, data must be explored repeatedly. In this case study, we use the approach of defining flow regions in a data-guided way.

Computer Graphics Lab, Dept. of Computer Science, ETH Zürich, CH-8092 Zürich, Switzerland.
VA Tech Hydro, Hardstr.319, CH-8023 Zürich, Switzerland.
{bauer,peikert,sato}@inf.ethz.ch Mirjam.Sick@vatech-hydro.ch

Flow regions of our interest are vortices and cavitation regions, but other types of flow regions, such as recirculations, could be treated similarly.

We experienced that typical regions of interest consist of a few percents of all grid cells. Hence, it makes sense to generate and trace particles only where needed. A common practice is to release particles from locations evenly spaced along lines or circles and at fixed time intervals. One can then observe how these geometric shapes are deformed over time. A different approach, which we pursued, is to randomly spread particles, aiming at a uniform density while avoiding patterns and clustering.

Instead of tracing discrete particles, 3D textures could be used to visualize the flow field, e.g. by extending the Lagrangian-Eulerian Advection algorithm proposed by Jobard et al. [5] from two to three spatial dimensions. However, we did not choose this approach because hardware support for 3D texture is not generally available. Also, for time-dependent 3D textures, texture loading time becomes an issue.

The goal of evenly spaced glyphs has led to the streamline placement techniques [8] which, in principle, could be extended to four dimensions for time-dependent flow. Our approach is instead to exploit the conservation of mass property of physical flow fields. For incompressible flows, conservation of mass means zero divergence, therefore the particle density remains constant if it was initially constant. For compressible flows (which we are not concerned with in this case study), an initial particle distribution can be made to reflect the density of the medium.

A common problem of particle-based flow visualization is the need for injecting new particles during the time evolution. This is necessary to maintain a uniform particle density. Texture-based methods accomplish this by continually adding noise. Noise is smeared enough by the large filter kernels typically used in LIC to avoid visible artifacts. But this solution is obviously not applicable to discrete particles. Instead, our strategy is to extend the region of interest by a few layers of “buffer” cells where particles are kept invisible. When particles enter or leave the region of interest, visibility is changed smoothly. In the buffer cells, the correct particle density can now be maintained by adding and deleting invisible particles. And because of mass conservation, this carries over to the visible particles, too.

2 FLOW REGIONS OF INTEREST

The industrial case under investigation is the flow field in the draft tube of a water turbine. The flow in a water turbine passes through the vaneless spiral casing and enters the stationary stay and guide vanes which accelerate the flow. The flow decelerates through the runner and enters the draft tube, which is the last component of a water turbine. A draft tube generally is a diffuser with or without a bend whose objective is to convert the available kinetic energy into pressure rise. Depending on the operating conditions, the swirl of the flow at runner exit (and thereby at draft tube inlet) is strong enough to cause a flow instability, which is called the draft tube vortex. The typical shape of a draft tube vortex is that of a rotating helix. Due to the low pressure in the center of the vortex, the flow is often cavitating for a wide range of operating conditions, so that on the test rig, the vortex rope is naturally visualized by means of the cavitation bubbles, see Fig. 1.

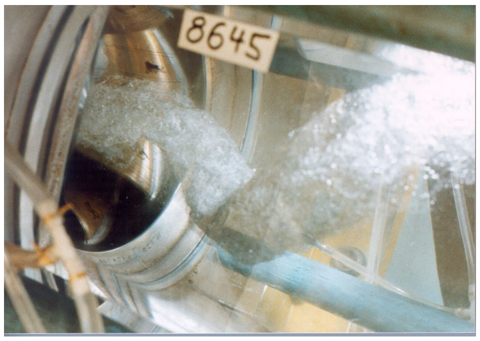

Figure 1: Photograph of a cavitating vortex rope.

The part-load vortex rope is of technical relevance for two reasons. Firstly, it causes vibrations within the runner and on the hub and the shaft. Secondly, it leads to an unstable throughflow and unstable power output. The shaft vibrations caused by the draft tube vortex can cause severe damage of the machine. Therefore in some plants the operating range of the machine is restricted in order to avoid a strong vortex and the resulting damage.

The design of the runner blades and the hydraulic contour of the draft tube determine the onset and the strength of the draft tube vortex. It is therefore important to understand the details of the vortex rope flow in order to define physically well founded design rules for a runner design which is safe with respect to the draft tube vortex.

The visualization methods demonstrated in this paper are based on instationary CFD simulations of the draft tube vortex in a pump turbine. The 3D Navier Stokes equations were solved by using the commercial code CFX-TASCflow [9] with a circumferentially averaged inlet velocity profile resulting from a CFD simulation of the runner flow.

We will generally assume that regions of interest (ROI) can be specified by a scalar field. In many cases, such a scalar exists among the CFD data channels or can easily be derived from them. For example, cavitation regions are characterized by pressure values falling below the vapor pressure. And vortices can be characterized by high values of either helicity or normalized helicity.

When trying to define a region by a threshold of a scalar field, the result is often more than one connected component. It can then be necessary to perform a selection among these components.

In the case of a vortex, there is an alternative way to define a ROI based on an extracted vortex core. In an earlier paper [2] we described how to track a vortex core in time-dependent data. The idea is now to radially extend the core line as long as the "vortex strength" is above a given threshold. We define vortex strength as the absolute value of the imaginary part of the complex eigenvalues of the velocity gradient tensor. Similar vortex tubes, but based on pressure values, have been computed by Banks and Singer [1]. Fig. 2 shows the principle of our algorithm, which consists of the following steps:

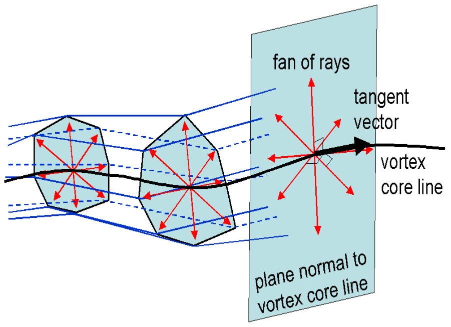

Figure 2: Principle of the vortex hull algorithm.

```
for each point on the vortex core line
  Project the velocity field onto a  plane perpendicular
  to the vortex core line.
  Construct a fan of rays which spreads over the plane.
  for each ray
    Generate points on the ray at small intervals.
    for each point on the ray
      Evaluate the vortex strength.
      if vortex strength < threshold break
    next point on the ray
  next ray
  Draw a star-shaped polygon connecting the last points.
next point on the vortex core line.
Connect star-shaped polygons to a prismatic tube around
the vortex core line.
```

Figure 3: Pseudo-code of the vortex hull algorithm.

Fig. 4 shows a vortex tube resulting from this algorithm, as well as the vortex core line and some streamlines indicating the vortical structure. With a slight modification, this algorithm produces a scalar field containing the vortex strength for nodes inside the vortex tube and zero values outside the vortex tube.

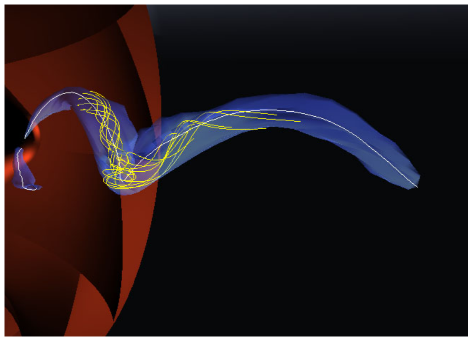

Figure 4: Vortex core and computed vortex extent.

3 THE VISUALIZATION TECHNIQUE

For the purpose of selectively visualizing the flow in regions of interest, we modify the standard particle-based technique in a way to achieve these goals:

- Particles are visible exactly in the ROI with a smooth transition (fading) if they enter or leave the ROI.
- Particle density represents the density of the medium during the whole simulation. For incompressible flow, this means particle density is initially constant and remains constant.
- Particles are distributed in a way that no regular patterns or other artifacts occur.

3.1 Visibility

In our application, the ROI is defined by high values of a scalar field *s(x)*, which is either directly given in the datasets (e.g. the negative pressure) or has previously been derived from them (e.g. the helicity, which can be computed from the velocity). Particles become invisible if, at their location, the field *s(x)* drops below a threshold s_0. Above a higher threshold s_1, they become visible. Between the two thresholds, a smooth transition is made by using either semitransparency or reduced size glyphs.

Invisible particles can be generally treated as nonexistent by the implementation. However, in the vicinity of the ROI it is good to trace invisible particles, too (see Section 3.3).

3.2 Particle seeding

The goal of the particle seeding is to generate a uniform distribution while avoiding clusters and regular patterns. In multi-dimensional spaces, quasi-random sequences give better results than pseudo-random sequences or jittered regular samples. The latter two methods lead to clustering (Fig. 5a, 5b) or patterns (Fig. 5c).

Sobol' quasi-random sequences were proposed [7] and have recently been used [6] for LIC methods. An advantage of the Sobol' points is that they can be tiled with no visual seams, as is shown in Fig. 5d and 5e. This allows us to tile the physical space and to create quasi-random points only for a single tile, which is repeatedly used wherever it covers the ROI (extended by a few layers of grid cells, see next section).

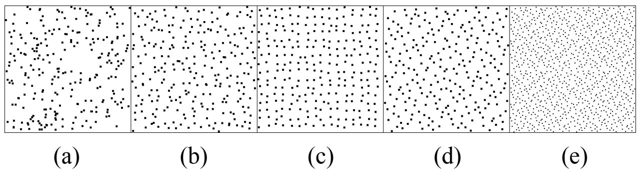

Figure 5: (a): Pseudo-random samples, (b),(c): jittered regular samples, (d): Sobol' points, (e): four tiles of Sobol' points.

3.3 Particle tracing

Initially and after each time step, the grid cells are classified as "inner cells", "buffer cells" or "outer cells". Inner cells are those where at least one corner satisfies $s(x) > s_0$, the partial visibility criterion. Buffer cells are the cells which are in a topological k-neighborhood of inner cells. Additionally, all inner cells within the k-neighborhood of an inflow boundary are reset to buffer cells. The remaining cells are classified as outer cells (see Fig. 6).

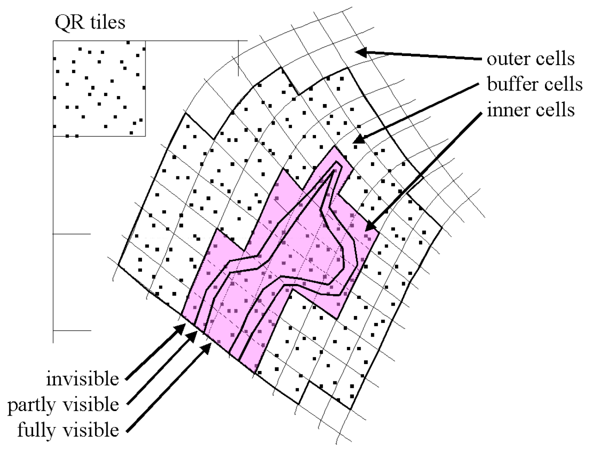

Figure 6: Classification of cells, particle visibility, and tiles of quasi-random points (2D representation)

The particles are initially distributed over inner and buffer cells. After a few time steps, particles are advected and now cover a different area. As long as this area completely covers all inner cells, everything is fine. But this condition can fail, also because the set of inner cells is dynamic.

When this situation occurs, all particles in the buffer cells are cleared and new particles are generated in the buffer cells. Since these particles are invisible by definition of the buffer cells, this replacement has no visual impact. Nevertheless, it is important that old and new particles join seamlessly at the border between buffer and inner cells, since the particles may later become visible. By using Sobol' quasi-random points, we can meet this requirement sufficiently for the purpose of visualization. And by making the ring of buffer cells wide enough, we don't have to replace particles too often.

Based on these considerations, we designed a particle tracer of the following structure:

```
Initialize a particle list.
currentTime := startTime
while currentTime < endTime
  Get current velocity field and scalar field.
  if update interval has expired
    Update the classification for each cell
    (according to the scalar values of its nodes).
    Delete all particles in OUTER and BUFFER cells.
    Generate new particles in BUFFER cells.
  end if
  currentTime := currentTime + timeStep
  for each particle in particle list
    Get current velocity of the particle.
    Calculate new position of the particle.
    Find new cell and local coordinates of the particle
    (delete the particle if it has left the grid).
  next particle
  if drawing interval has expired
    Draw the particles.
  end if
end while
```

Figure 7: Pseudo-code of the particle tracer

Replacing particles in the buffer cells can be done at fixed time intervals, although a safer method would be to release marked particles at the outer boundary of the buffer cells. A marked particle entering an inner cell would trigger the replacement.

Instead of generating a new set of quasi-random points for each replacement operation, we experimented with a cheaper solution, namely to just randomly offset the old points. This corresponds simply to a translation of all tiles, so there is no need to actually modify point coordinates.

Integrating particle paths in unstructured grids requires frequent incremental point searches [3]. A global point search is only necessary for newly generated particles, mostly because there is no spatial coherence in quasi-random sequences. To optimize the global point search, we organize particles in a regular grid which is a refinement of the tiling grid.

3.4 Conservation of mass

When velocity (and density in the case of a compressible flow) is interpolated from the node data, the resulting fields are expected to be mass-conservative. However, this is not the case if the standard techniques are used, namely trilinear interpolation in hexahedral cells and linear interpolation in tetrahedral cells. By comparing influx and outflux of each cell, we observed that typically 1-5% of the mass is "lost" numerically due to trilinear interpolation of node data. This loss can be reduced (to about 0.5-2%) if the dual grid is regarded (dual cells are the control volumes of the finite volume method used by solvers such as CFX-TASCflow).

In principle, mass-conservative interpolation could be used [4]. However, this would require the computation of two global stream functions for every simulated or interpolated time step.

We decided to use special interpolation only in cells next to solid (no-slip) boundaries, where most numerical mass loss occurs. In these boundary cells, the main effect of trilinear interpolation is that particles tend to stick at the boundary, eventually compromising the postulated uniform distribution. To prevent this, we don't allow particles to get closer to a solid wall than a certain fraction of a cell. This is done by clamping the appropriate local coordinate after each integration step, which is common practice [3] and can be justified by the assumption of particles with spatial extent.

3.5 Particle rendering

Besides the obvious techniques of rendering spheres (Fig. 8) or wide points, a good alternative are streamlets, i.e. short pieces of the local path line (Fig. 9). The streamlets can be rendered as alpha-textured quadrilateral strips (ribbons) made up of billboard polygons. Billboard polygons are directed toward the viewer and give the illusion of 3D rather than flat objects. Moreover, they can be lit with diffuse and specular reflection more effectively.

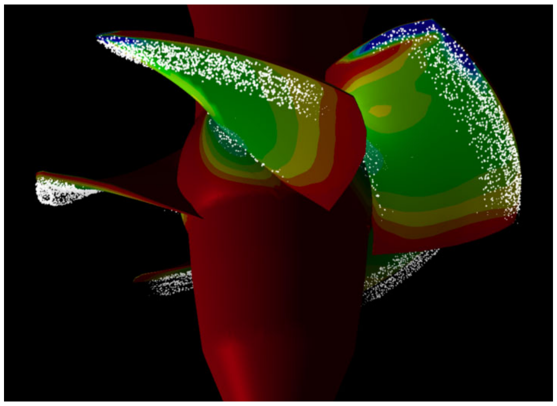

Figure 8: Cavitation bubbles near Kaplan runner blades (ROI specified by low pressure).

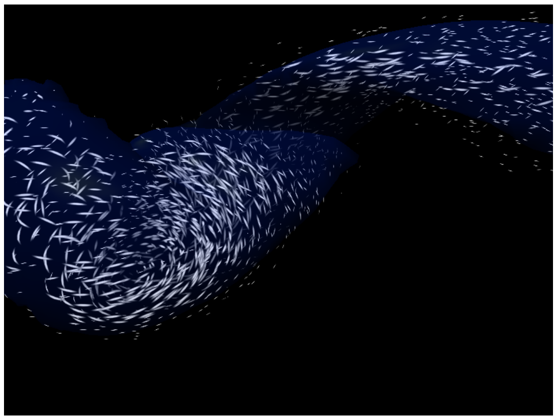

Figure 9: Vortex rope in Francis draft tube (ROI specified by high normalized helicity).

4 CONCLUSION

We proposed a modified particle tracer for selectively visualizing time-dependent 3D vector fields. Unlike previous representatives of its kind, our particle tracer is focused on special regions of interest in the flow, which can reduce the amount of data to be processed by one order of magnitude. By using quasi-random sequences when seeding new particles, a uniform particle density is maintained over time, and regular patterns and clusters are avoided. Due to the use of buffer cells, particles fade in and out smoothly, which is favorable for animations. We applied our method to various datasets from our industry partners, visualizing the vortex rope in the draft tube of a Francis turbine, and cavitation on the suction side of various turbine and pump runner blades. The concept of selectively visualizing relevant flow structures has proven to give additional insight into their complicated dynamic behavior.

Acknowledgment

This work has been supported by the Swiss Commission for Technology and Innovation, by VA Tech Hydro, Zürich, and by Sulzer Markets and Technology, Winterthur.

References

[1] D. Banks and B. Singer, Vortex Tubes in Turbulent Flows: Identification, representation, reconstruction. In *Proceedings of IEEE Visualization '94,* 1994.

[2] D. Bauer and R. Peikert, Vortex Tracking in Scale-Space, *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, D. Ebert, P. Brunet, I. Navazo (Editors), Barcelona, May 2002.

[3] P. Buning, Numerical Algorithms in CFD Post-processing, *Computer Graphics and Flow Visualization in Computational Fluid Dynamics,* von Karman Institute for Fluid Dynamics Lecture Series 1989-07, September, 1989.

[4] D. Feng, X. Wang and W. Cai and J. Shi, A mass conservative flow field visualization method, *Computers & Graphics,* 21 (6), pp. 749-756 (November 1997). Pergamon Press / Elsevier Science. ISSN 0097-8493.

[5] B. Jobard, G. Erlebacher and M. Y. Hussaini, Lagrangian-Eulerian advection for unsteady flow visualization, *IEEE Visualization 2001*, pp. 53-60 (October 2001). ISBN 0-7803-7200-x.

[6] A. Sanna, B. Montrucchio and R. Arina, Visualizing Unsteady Flows by Adaptive Streaklines, *Proc. WSCG 2000, The 8-th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media'2000,* February 2000, Plzen, Czech Republic, pp. I84-I91.

[7] D. Stalling and H.C. Hege, Fast and Resolution Independent Line Integral Convolution. *Computer Graphics Proceedings '95, ACM SIGGRAPH,* 1995, pp. 249-256.

[8] G. Turk and D.C. Banks, "Image-Guided Streamline Placement," *Computer Graphics Proceedings '96, ACM SIGGRAPH,* 1996, pp. 453-460.

[9] CFX-TASCflow Theory documentation, Version 2.11.1, AEA Technology, Chapter 4: Turbulence Closure Models, Harwell, U.K. (2000), http://www.software.aeat.com/cfx/ .