Transportation-based Visualization of Energy Conversion

Oliver Fernandes[1], Steffen Frey[1] and Thomas Ertl[1]
[1]*VISUS, University of Stuttgart, Allmandring 19, Stuttgart, Germany*
{*oliver.fernandes,steffen.frey,thomas.ertl*}*@visus.uni-stuttgart.de*

Keywords: energy transport, glyph-based visualization, glyph placement, volume rendering, energy flux, flow dynamics

Abstract: We present a novel technique to visualize the transport of and conversion between internal and kinetic energy in compressible flow data. While the distribution of energy can be directly derived from flow state variables (e.g., velocity, pressure and temperature) for each time step individually, there is no information regarding the involved transportation and conversion processes. To visualize these, we model the energy transportation problem as a graph that can be solved by a minimum cost flow algorithm, inherently respecting energy conservation. In doing this, we explicitly consider various simulation parameters like boundary conditions and energy transport mechanisms. Based on the resulting flux, we then derive a local measure for the conversion between energy forms using the distribution of internal and kinetic energy. To examine this data, we employ different visual mapping techniques that are specifically targeted towards different research questions. In particular, we introduce glyphs for visualizing local energy transport, which we place adaptively based on conversion rates to mitigate issues due to clutter and occlusion. We finally evaluate our approach by means of data sets from different simulation codes and feedback by a domain scientist.

1 INTRODUCTION

With the ever growing computational power available, flow simulations performed by scientists and engineers become more realistic and detailed. Therefore, the analysis of occurring flow phenomena becomes increasingly difficult, and available tools may not be adequate to the task. New visualization techniques are necessary to be able to explore the different aspects of phenomena occurring in the data. In particular, the consideration of conserved quantities proves to be a valuable asset in both mathematics and physics, and is ultimately fundamental for our understanding of nature. Used as a basis for modeling flow dynamics, the conservation laws of physics are employed to establish a set of partial differential equations to describe the dynamics of a fluid in motion. Basically, this is done by first determining the composition of a conserved quantity, and identifying its constituents. The different ways the quantity can change over time is then examined by modeling appropriate physical interactions. The quantity needs to be analyzed in a temporal context, as only examining a given state of the simulation, even when considering all defining variables, may not reveal all dynamics involved. For example, the visualization of the velocity does not suffice to fully understand the dynamics (i.e. transport) of momentum for the simple case of an incompressible steady laminar flow simulation. In this particular case, the solution is fairly simple: by taking into account the deviatoric stress tensor, the true momentum flux field can be derived for each component, and analyzed via standard vector field visualization.

However, this is not the case in general. Usually, all dynamics calculated by solving the partial differential equations cannot be extracted directly from the state of the fluid. In a more interesting example of a compressible flow simulation, one cannot directly derive, given the initial and final state of the fluid, how an energy change has occurred. The final state can, in theory, be reached by various combinations of exchanging heat or having work done on its volume.

In this paper, we propose an approach to visualize this process for arbitrary flow data by examining the exchange of total energy between control volumes. The resulting energy flux is in turn used to compare the composition of energy types in the initial state with the final state. Based on this comparison, a visual representation is generated showing the spatial distribution for the conversion between different forms of energy.

In Sec. 2 we give an overview on related work. After providing a quick outline of our overall approach and describing the underlying fundamentals regard-

ing energy densities occurring in flow (Sec. 3), we then discuss and evaluate the individual components of our approach, which we consider to be the main contributions of our paper:

- We introduce our approach to estimate energy conversion for visualization by reformulating and solving it as a transportation problem (Sec. 4).
- We then discuss several mapping approaches for the visualized quantity to generate meaningful renderings from the obtained energy data (Sec. 5).
- We evaluate the utility of our approach with different data sets and an expert study with a domain scientist (Sec. 6).

We finally conclude our work in Sec. 7.

2 Related Work

Since the visualization of energy and energy transport can lead to significant insights of the explored data set, a few approaches to the problem have been suggested. However the intricate nature of the problem, both theoretically and technically, proves it to be a difficult task to produce a concise and comprehensive visualization of all the information contained.

Many visualization techniques to explore flow data have been developed over the years. Relevant for the approach discussed in this paper are several prior works, relating to various steps of our approach.

Already in 1905, visualizing the participation of energy in an experiment or a simulation, often so-called Sankey diagrams were (Sankey, 1905) (and still are) used, to give an abstracted overview of energy exchange. Of course, any further information regarding the spatial distribution is lost.

Many geometry-based techniques have been derived processing the underlying fields of the data to generate geometry representing its characteristics. Among these are the well-known streamlines and similar integration techniques resulting in geometry, which resemble the velocity vector field of the flow. For an overview of geometric visualizations, see (McLoughlin et al., 2010).

Often a mixture of texture-based and geometry-based approach already reveals much information about the flow, for example done with LIC on isosurfaces in (Laramee et al., 2004). An extension of these are stream tubes, which are densely seeded streamlines on a closed curve, therefore creating a tube-like structure (Schroeder et al., 1991). Various improvements have been made to enhance the usage of stream tubes, for example (McLoughlin et al., 2013). An interesting property of these is that there is no flow across the tube mantle (per definition). In this context, Meyers and Meneveau (Meyers and Meneveau, 2012) suggested so-called energy transport tubes to visualize the transport of kinetic energy in statistically steady flows. Again, defining an energy tube mantle as the two-dimensional manifold where energy flux does not have a normal component (i.e. total energy stays the same summed up across the cross-section of such a tube), they give a good visualization of the propagation of kinetic energy. However, they do not help in exploring phenomena beyond transport of an energy form. Also, arbitrary sources and sinks cannot be included as easily. They also do not consider the different forms of energy present in a flow simulation, and how the composition of the total energy can reveal additional information about the data.

With our approach aggregating energy transport information to yield a scalar value for the conversion, direct visualization techniques need to be considered. Very common are techniques like volume rendering which are used for scalar fields of flow data such as density. Still being an active part of research, volume rendering has many applications in scientific visualization, e.g. (Kroes et al., 2011). Also more lighting-based techniques can help to present the data more intuitively, as for example shading cues, employed in (Boring and Pang, 1996) for vector field fields, such as the velocity. A good overview of direct visualization for higher order data, and possible visualization approaches is given in (Sadlo et al., 2011).

Glyphs provide a very good means of visualizing multivariate and higher order data. Our approach results in multivariate data represented by a dense graph of the data set with both nodes and edges having relevant scalar values applied (node count on the order of cell count). A concise yet intuitive way of interpreting the data is needed. However, for multivariate data, there is no straightforward standard technique to visualize involved quantities directly. An overview of using glyphs to represent multivariate data can be found in (Borgo et al., 2013) and (Ropinski et al., 2011). Custom glyphs incorporating all dimensions and variables of the data can be designed, as is done for example in vector uncertainty visualizations (Wittenbrink et al., 1996), where the classic "hedgehog" vector field visualization is enhanced with additional information on the uncertainty. A novel approach of using the glyph in a multi-scale fashion, where different zoom levels of the same dense glyph layout revealed different information about the data set, was presented in (Hlawatsch et al., 2011). This also encompasses customizing glyphs to fit the underlying data, as shown in (Kraus and Ertl, 2001) or (Tong et al., 2016). Glyphs also work well for flow data when considering the

tensors (Jacobian, Hessian) of flow quantity derivatives, and are a popular tool for visualizing higher order quantities (Schultz and Kindlmann, 2010). An example of combining direct and glyph visualization in an integrated approach is shown in (Üffinger et al., 2011).

Our technique determines the flow of energy between subsequent time steps to visualize the conversion processes. Conceptually, this is related to the earth mover's distance (EMD). Basically, it determines the minimum cost of turning one distribution into the other. First, it constructs a complete graph, taking each cell of the data as a node (i.e., every pair of nodes is connected by an edge). In our approach, we use a minimum-cost flow solver to compute the distance (e.g., (Goldberg and Tarjan, 1990)). In more detail, we employ a widely used technique in this category: the cost-scaling push-relabel algorithm (Goldberg and Tarjan, 1986). It maintains a preflow and gradually converts it into a maximum flow by moving flow locally between neighboring vertices using push operations under the guidance of an admissible network maintained by relabel operations. This algorithm will find an optimal solution (i.e., one with minimal associated cost). In this paper, we use Google's implementation of this algorithm[1]. Its complexity is $O(n^2 \cdot m \cdot log(n \cdot c))$, where n denotes the number of nodes in the graph, m denotes the number of edges, and c is the value of the largest induced edge cost in the graph. Due to this comparably high complexity, we exploit certain properties, particularly to significantly reduce the number of edges to make its direct application to cells of a data set computationally feasible. The EMD is particularly popular in the computer vision community where distances are typically computed between color histograms, e.g., for image retrieval (Rubner et al., 2000). To interpolate between commonly used distributions in computer graphics (like BRDFs, environment maps, stipple patterns, etc.), Bonneel et al. (Bonneel et al., 2011) decompose distributions into radial basis functions, and then apply partial transport that independently considers different frequency bands.

3 Motivation and Fundamentals

In this section, we give a quick overview of the problem determining the transport of total energy for visualization within a given domain, and outline our approach to solve this problem. We then give a short introduction to the relevant fundamentals of energy in fluid dynamics. The section concludes with a brief summary of the minimum cost flow algorithm.

3.1 Motivation for Visualizing Energy Transport

Any process can be divided into variables describing the state, e.g. velocity, pressure etc., and variables describing the dynamics as for example various fluxes. The energy transport we aim to visualize in this paper, can be classified as pertaining to the dynamics of any flow. However, these dynamics involved propagating the system between states cannot be easily extracted in an arbitrary flow, but enable useful insights, as for example, predictions for the evolution of the system. In real life phenomena, dynamical data often cannot be captured, and is unavailable for analysis (e.g. momentum exchange in global weather air flow data). Using computational fluid dynamics, the evolution equations (Navier-Stokes equations, see section 3.2) modeling the flow are known. Even so, there are various approaches to discretize and solve these equations, possibly leading to the resulting dynamics being difficult to extract from the calculation. Even if possible, doing this highly depends on the type of simulation and solver (e.g., implicit or explicit), as well as the respective implementation. In all cases, retrieving the state variables is fairly easy.

Our approach estimates the energy dynamics of any arbitrary system, for which the state variables are known at different time steps (Sec. 4). One of the two prerequisites necessary for the employed algorithm is to know how the amount of energy in a given energy form is related to the state variables of the system. The other prerequisite is information regarding all additional external sources and sinks (or an approximation thereof), to satisfy the energy conservation law. The composition of the total energy, i.e., the types of energy involved, may be chosen arbitrarily for our calculation. This is an important property of our approach in that it is widely applicable to data from many kinds of sources. In particular, using only an initial and final state, knowledge of the actual evolution equations, which, as explained above, might not always be available, are not necessary to compute the energy dynamics. However, any additional knowledge of the domain, or knowledge of the transport mechanisms, can be included to improve the accuracy of the estimated energy fluxes.

To render the myriad of energy flux information yielded by our visualization, we discuss several mappings for the data in Sec. 5). These include a volume rendering for exploration and overview, as well as a glyph-based approach. The glyph approach is

[1] https://developers.google.com/optimization

further enhanced by automatically placing the glyphs for minimal occlusion, while retaining the most important information.

3.2 Energy in Fluid Dynamics

Several forms of energy are present in a compressible flow data set. First, there is kinetic energy, for any mass moving at a non-zero velocity. Other kinds of energy include internal energy, a thermodynamical quantity dependent on the pressure and temperature. While we focus on these two types of energy, various other forms of energy can effortlessly be considered and implemented in a straight forward fashion. In the following, we give a brief overview of the partial differential equations governing compressible flow. We also give a quick summary of the energy types involved and their calculation.

Being independent of underlying governing equations, we only need to consider the forms of energy present in a given data set, regardless how the data was actually produced. For the data sets in this paper, this specifically means considering the internal and kinetic energy forms. For all known phenomena in compressible flow, the Navier-Stokes equations (NSE, 1) prove a very accurate and general model (Hasert, 2014; Landau and Lifshitz, 1987).

Derived from basic conservation equations for mass, momentum and energy (yielding (1a), (1b) and (1c) respectively), the compressible NSE take the following form:

$$\partial_t \rho + \nabla \cdot (\rho u) = 0 \tag{1a}$$

$$\partial_t (\rho u) + \nabla \cdot (\rho u \otimes u + pI) = \nabla \cdot \sigma' \tag{1b}$$

$$\partial_t (\rho \varepsilon) + \nabla \cdot (\rho u (\varepsilon + p)) = \nabla \cdot (\sigma' \cdot u) - \nabla \cdot q - B \tag{1c}$$

$$p = \rho R T, \tag{1d}$$

where ρ is the mass density, u the velocity, p the pressure and T the temperature. ε denotes the specific energy per unit mass (i.e. the energy density), and σ' the deviatoric stress tensor. q is the the heat flux. The final equation closing the set is the equation of state for an ideal gas (1d), using the gas constant R. B is the sum of conservative external fields. However, for numerically solving this set of partial differential equation, various terms need to be approximated and modeled accordingly. In some cases, the exact modeling may not be available, as for measured data. The energy conservation is expressed in terms of the specific energy per mass unit ε:

$$\varepsilon = e + \frac{1}{2}|u|^2. \tag{2}$$

$e = c_v T$ is the internal energy. Multiplication with ρ defines an energy density e_{tot}. In the following, the term energy will be used synonymous to energy density for brevity.

Using the NSE as a basis, the types of energy occurring in compressible flow can be derived. A given energy distribution can be split into several distinct parts. For one, there is the internal energy given by $e_{\text{int}} = \rho c_v T$ with the heat capacity c_v at constant volume. The heat capacity c_v can be expressed with a material parameter, the adiabatic index constant γ, as $c_v = R/(\gamma - 1)$, finally yielding an internal and kinetic energy of

$$e_{\text{int}} = \frac{\rho T}{\gamma - 1} \tag{3}$$

$$e_{\text{kin}} = \frac{1}{2}\rho |u|^2 \tag{4}$$

Other forms could also be taken into consideration, for example potential energy in external fields B, given with $e_{\text{pot}} = \rho g z$.

The NSE as expressed in (1), do not encompass all possible flow physical phenomena. Various extension (e.g. Magnetohydrodynamics, which also consider the Maxwell equations (Goedbloed and Poedts, 2004)) can be found, possibly adding further forms of energy to the composition of the total energy. The data sets considered in this paper focus on the internal and kinetic energy, and assume no external fields. In any case, the conserved quantity is the total energy e_{tot}, given by the sum of all the occurring energy types. For the cases considered in this paper, this is simply given by $e_{\text{tot}} = e_{\text{kin}} + e_{\text{int}}$.

To fully describe the flow problem, or more specifically the energy conservation problem, the system needs to be closed, or the interaction with the surrounding environment must be considered. This is usually done by restricting measurements or simulation to a certain domain, for which the boundary conditions are known or can be modeled sufficiently.

3.3 Minimum Cost Flow Problem

On the basis that energy only gets exchanged in a close neighborhood of a cell, we formulate our energy exchange with a minimum cost flow problem. In this subsection, we give a brief summary of this algorithm and its terminology.

A minimum cost flow problem examines the transport of a scalar quantity through a network, defined by a graph G. Two sets of nodes are given: a source set S and a destination set D. All nodes s, d in S, D are assigned an *integral* scalar value s_Q, d_Q called supply and demand respectively. The supply and demand for each node in S, D must be chosen so that

$$\sum_S s_Q = -\sum_D d_Q. \tag{5}$$

Equation 5 expresses that the quantity in question is preserved. A graph connecting the two sets is established by creating a set E of directed edges between source nodes and destination nodes. Only connections from S to D are allowed, connections back or within S or D are disallowed. S, D and E make up the graph G which defines the input for a minimum cost flow algorithm.

The problem now consists of moving supply from S to D, that is, assigning a set of so-called 'flow' values f_e for each edge e, so that:

$$\begin{aligned} \forall s \in S: & -\sum_{f \in I} f = s_Q,\ I = \{f_e \mid e \text{ starts at } s\} \\ \forall d \in D: & \sum_{f \in I} f = d_Q,\ I = \{f_e \mid e \text{ ends in } d\} \end{aligned} \tag{6}$$

In simple terms, this means that all the supply needs to be transported to cover all the demand, and can only be moved along an edge defined in E. No source node can distribute more than it has supply, and no destination node can receive more than it has demand.

This condition alone potentially still allows a myriad of solutions. To further limit the possibilities, a more specific one can be selected by introducing a secondary condition. This requires assigning a cost c_e to every edge e in E, which can be arbitrarily adjusted. With the assignment of c_e, the selected solution must additionally minimize the so-called total cost c_T:

$$c_T = \sum_{e \in E} f_e c_e \tag{7}$$

The cost value c_e can be lowered for certain edges to favor these edges being chosen to move the supply quantity.

4 Extraction of Energy Transport

To enable a comprehensive visualization of the energy dynamics, we first need to extract the appropriate quantities. The second part is to map these quantities to a renderable representation, explained in Sec. 5. Initially, the value for the two relevant energy densities from the state variables, employing equations 3 and 4, is calculated. After determining the energy contained in each cell for two consecutive time steps, we calculate an approximation of the flux for the total energy e_{tot} between the two time steps. Since the quantity is conserved, the total change of the energy in the domain between time steps can be determined by taking the initial total energy and considering all additions and subtractions at the domain boundary (Reynolds transport theorem (Leal, 2007)).

Obviously energy changes can only be transported by a mechanism (i.e. energy itself does not have a 'speed'), and therefore the limiting speed of this mechanism also limits the distance energy can be moved. For a sufficiently small time step, this transport can be assumed to only occur within a small neighborhood for each cell. This extraction step "splits off" the energy transport information and prepares it to be mapped to an appropriate visual representation. In the following paragraphs, we reformulate our problem of determining the energy flow as a transportation problem. The outcome can then be directly processed by a standard minimum cost flow solver to eventually calculate the energy flux.

For the rest of this discussion, the term energy refers to the total energy density. Matching the energy flux problem to be solvable by a minimum cost flow algorithm involves the following steps:

1. Define all cells of the input grid in the first time step as nodes in S and associate the correct energy representing the supply s_Q. Analogously, define the cells in the second time step as nodes in D with demand d_Q.
2. Model the correct boundary conditions for the nodes in both S and D.
3. Convert the floating point values to integers (required by the solver), and normalize the energy to fulfill equation (5) in a distribution-preserving fashion.
4. Determine an edge set E connecting the two time steps and solve the defined minimum cost flow problem. In our setup, we employ the google ortools solver to evaluate the minimum cost flow problem.

In the following paragraphs, we explain these steps in more detail.

1. Preparing the Energy Data. As an example for given input data without compromising generality, we consider the fluid state to be defined by velocity u, pressure p and temperature T in the following subsections. The step 1 can be implemented in a straightforward manner. With the velocity u, the pressure p and the temperature T provided for each cell by the simulation, the mass density ρ can be calculated using equation (1d). Using the mass density, the energies can be determined with equations (3) and (4), leading to an energy e_{tot} (see section 3.2). With the equation of state, similar calculations can be done for any set of state variables.

2. Handling the Boundary Conditions. Modeling the correct boundary conditions for step 2 is more complex. In particular, the source and destination time steps cannot be treated symmetrically: Any amount of quantity *entering* the domain has to be

considered in the source step, while quantity *leaving* the domain only applies to the destination time step. In order to keep the energy conservation assumption valid, the exchange of energy over the boundaries must be fully accounted for. This will also ensure condition (5) is automatically fulfilled.

To correctly map the boundary effects, we need to identify domain boundaries where energy exchange is possible. For brevity, we only discuss a few important exemplary boundary conditions occurring in our data sets. A correct modeling for missing (or even completely different) conditions can be easily derived based on these ideas.

For each of the state variables u, p and T a boundary condition can be defined as explained in section 3.2. Periodic boundaries need to be constructed with the effect that an addition of energy is modeled in the source time step S, while the exact same removal is implemented in the destination D for the periodic neighbor. Dirichlet boundary conditions allow for both adding and removing energy from the domain, depending on the prescribed value and the actual value of the quantity in the adjacent control volume. For example an inflow boundary condition (u, p and T prescribed) can be modeled by adding a cell (called boundary node) to the source set S for all grid cells at a location fulfilling the condition (i.e. touching the inflow boundary), and supplying the state variable according to the prescribed values.

Neumann boundary conditions also may change the energy contained in the domain. As a prominent example, consider the outflow boundary condition, e.g. where derivatives of u, and T are zero, while the value of p is fixed. The mass flux itself can transport energy out of the domain, and the difference in p may exchange energy too. The removed energy can be calculated taking the amount of mass crossing the boundary (scalar product of mass flux and boundary normal), while the pressure is set to the fixed value of the simulation parameter. Now the associated energy can be calculated and, just as for the inflow condition, a set of additional cells (i.e. boundary nodes) need to be added into the destination set D, to be used with the calculated energy. For different combinations of Periodic, Dirichlet and Neumann conditions for the state variables, the calculation of energy exchange must be accounted for individually. Some boundary conditions do not allow for an exchange in the first place, and therefore do not require a special treatment.

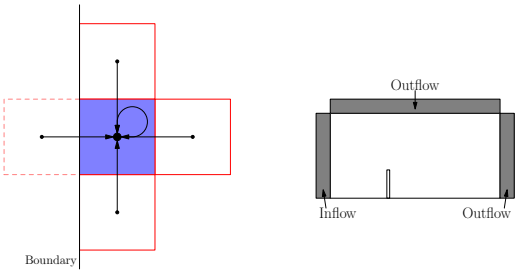
Fig. 1(b) shows a schematic of a cross section of our beam in cross flow data set. The gray boxes indicate areas, to which energy exchange is possible, and needs to be handled by boundary nodes. Note that only the connectivity to the data set and energy density of these boundary nodes is needed, the geometry is irrelevant.

3. Normalization and Quantization. After properly setting up the energy distributions in the previous steps, the rescaling and quantizing step 3 is performed, to fulfill equation 5 and to turn the floating point energy to an integral supply/demand value for the minimum cost flow algorithm. This involves simply multiplying all energy values by a fixed value and rounding down the result to the closest integer.

This quantization, and numerical precision issues, or even insufficiently accurate modeling or estimation of the boundary condition, can cause the precondition (5) to fail. To remedy the problem, a normalization is done to adjust the higher total domain energy to the lower one. Simply scaling all values by a constant factor (the ratio of the total domain energies) is not possible, since the energy values are already quantized. Instead, a unit of supply/demand from a node in the set with the higher total domain energy is removed in a random fashion, until both time steps contain the same accumulated total energy. To keep the overall distribution approximately the same, cells get selected for removal proportional to their energy content.

4. Building and Solving the Graph. Now that the nodes S, D, and associated supply and demand s_Q, d_Q, of the graph have been established, the edge set E can be constructed in step 4. Following the assumption at the beginning of this section, that energy can only propagate to cells within a certain neighborhood, the edges can be constructed accordingly. For each node in the source S, a neighborhood set is selected in the destination set D based on spatial proximity. Fig. 1(a) shows the most simple subgraph for a single destination cell (blue). This cell can potentially receive an energy flow from the source cells (framed in red) located in the neighborhood. The figure only shows immediate neighbors in two dimensions for simplicity, and may actually be an arbitrary unstructured grid in three dimensions with connection to further neighbors in addition to the nearest. The neighborhood to which edges are constructed, is determined by a face-neighbor relationship, and recursive collection of nearby cells, until a certain maximum distance from the original cell is achieved.

This distance is determined by the maximum speed at which a mechanism can transport energy, multiplied by a constant factor, to account for errors introduced in the quantization/normalization step. This can happen when the algorithm is forced to move a unit of energy further than it could have propagated physically, and can be controlled by the edge cost c_e (see below). Note that a cell's source node has an



(a) Simplified minimum cost flow subgraph for a single cell (blue) and connected source cells (red).

(b) Boundary conditions example, here for the beam in cross flow data set.

Figure 1: Establishing the minimum cost flow network graph, and how boundaries are considered.

edge to the same cell's destination node. A boundary node, depicted in Fig. 1(a) with a dashed red line, is only connected to it's adjacent "spawning" node.

All edges e in the set E get assigned a cost c_e based on Euclidian distance between the cell centers. Therefore, the actual geometry of boundary nodes is arbitrary, since the boundary node only has a single edge, connecting it to the source/destination cell in the data domain to which it injects, or from which it removes energy, respectively, and must fulfill equation (6). We promote the use of shorter edges, based on the fact that the physical energy transport is continuous, and the assumption that the time interval between time steps is very small. We do this by defining the edge cost c_e using the squared Euclidian norm on the vector connecting the cell centers s,d: $c_e = \|s-d\|^2$.

With construction of the edge set E the energy transport has been fully mapped to a minimum cost flow problem. To solve the problem, we employ a generic minimum cost flow algorithm with the property of finding an optimal solution, in the sense that the total cost c_T (see equation (7)) for this solution is a global minimum for all possible solutions.

By executing the minimum cost flow algorithm on the graph G a flow value f_e for each edge in E is determined. These flow values can be interpreted as the energy needed to be transported from starting (source) nodes to ending (destination) nodes, to achieve the energy distribution in the second time step starting from the first. In addition, the total cost c_T for the generated solution can be interpreted as a coarse indicator for the accuracy of the approximation.

5 Visual Mapping of Energy Transport Quantities

Obviously, the representation provided by the minimum cost flow algorithm cannot be displayed in a straightforward manner, since the graph G is usually very dense. Even when considering the fact that the outflow values from source nodes contain the same information as the inflow values from destination nodes (only sorted differently) there is still too much information per cell. For each destination cell for example, there can be several (incident) edges, of which each in turn carries two relevant scalar values, the energy distribution s_r (e.g. encoded in the ratio $e_{\text{int}}/e_{\text{tot}}$) in the source cell, and the energy flow amount f_e. Additionally, the destination cells energy ratio d_r has to be considered to extract actual conversion information. We show several approaches which help to explore the energy conversion under different aspects. These mappings may also be combined to complement each other.

5.1 Energy Conversion

First, we define a scalar field giving an overview of the energy conversion. This field incorporates the transported amount, as well as the ratios and their spatial distribution over the entire domain. The propagation direction of energy is lost, however. To calculate an expressive scalar value C for the conversion, given the input as explained in the introduction part of this section, we use a point-based intermediate data representation mapping each point to a single scalar.

For our intermediate representation, we construct a point p_e for each of the incident edges in the graph G. The geometry of p_e is simply the cell center of the destination node. To define a scalar value $C_{\max}$ for each point, we calculate s_r and d_r for the source and destination cells. We then take the absolute value of the difference between s_r and d_r and weight it with the amount of energy f_e transported over the edge:

$$C_{\max} = \|s_r - d_r\| \cdot f_e \tag{8}$$

This yields a scalar for the conversion ratio between consecutive times in each cell, regardless of cell geometry or absolute values, while still encoding all important information.

5.2 Volume Rendering

With Volume Rendering being a good tool to show spatial distributions, we use it to generate overview approaches of the energy transport.

Total Energy Flow Amount. The simplest visualization of the energy transport data is summing up the inflow of energy f_e over all incident edges of the graph for each cell. Displaying this scalar in a volume rendering can give a quick overview of where energy was transported between cells, and the amount. However, any directional information, as well as energy composition information is lost (e.g. Fig. 3(a)).

Direct Conversion Rendering With the Conversion quantity defined above (Sec. 5.1) being a scalar, a direct volume render can be performed.

Both representations can then be rendered as a common scalar field as a volume rending. The resulting scalar field is then rendered by common volume rendering techniques (e.g. Fig. 3(d)).

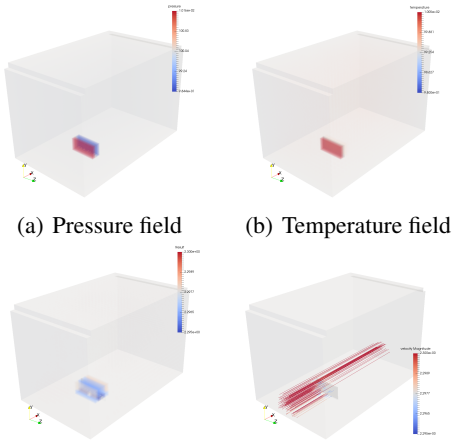
5.3 Glyph-Based Mappings

To preserve the information of local energy transport on the scale of a cell, we employ glyphs for the multivariate data.

Transport Glyph. This visualization has the advantage of keeping all the information calculated intact, including directions of transport, as well as energy ratios. The simplest way of displaying all the graph's information is directly showing a subgraph for each node in a set of selected destination cells.

For every edge from a source cell to the destination cell, a glyph is constructed. For the energy remaining in the cell, a sphere is drawn, otherwise a cone is shown, pointing towards the destination cell. The radius of the sphere, or the cone base respectively, encodes the amount of inflow f_e from the source cell, while the color shows the conversion ratio. The visualization enables a detailed examination of energy flow and conversion for a smaller region. The glyphs may be placed randomly or manually.

Importance-Weighted Glyph Placement. To avoid the disadvantages Transport Glyph (occlusion) and Direct Conversion Rendering (aggregated information) have, both variants can be joined into a combined visualization. Based of the conversion ratio (Sec. 5.1) as an importance weight, cells are selected as candidates for displaying glyphs. Using a user-defined threshold for the conversion ratio $C_{\max}$, a cell is added to a set of glyph candidates. Since displaying a glyph for all candidates would still lead to massive occlusion, candidates are then randomly picked to represent a small neighborhood. The size of this neighborhood can also be interactively chosen by the user, offering an overview vs. detail trade-off. This allows for several valid placement distributions. To select one, we run the placing algorithm several times, and keep the distribution with the most glyphs, as this is the one offering the most information. Note that the glyph distribution can be controlled by any scalar value as weighting, and may be used in conjunction with completely different feature detection methods.



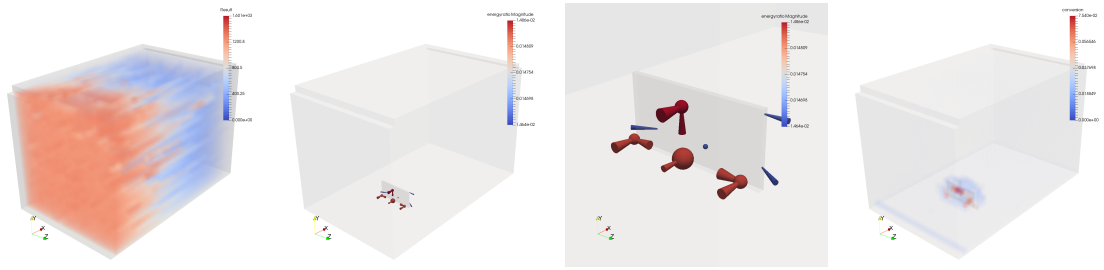
(a) Pressure field (b) Temperature field

(c) Velocity magnitude and streamlines

Figure 2: Volume renderings of the input fields. The images show the state variables for the beam in cross flow data set from the destination time step for comparison to our approach shown in figure 3.

6 Results

We employ several flow simulation data sets to discuss the results achieved with our approach and demonstrate its utility. These data sets were produced by different computational fluid solvers. For the beam in cross flow data set, most simulation parameters were known, particularly an exact accounting for all the boundary conditions of all variables. In summary, the data consists of an unstructured, time-varying grid containing 14469 cells. The topology is constant, and 7 boundary patches are defined. The state of the fluid is recorded in the velocity, pressure and temperature. The boundary conditions are set to appropriate Dirichlet values for the inlet, "slip" for the side walls, "outflow" for top and back walls, and "no-slip" for the floor. See below for exact values. For the 5jets data set (obtained from K.L. Ma, UC Davis), only the state variables were available. The data consists of a regular grid, converted to the corresponding unstructured mesh consisting of 262144 cells. The state variables provided are internal energy, density and velocity. We choose the time interval between time steps to be sufficiently small for our assumptions to apply. The boundary condition had to be reconstructed by considering the data close to the boundary, and estimating sensible boundary condition types and



(a) Total Energy Flow (b) Sparse Visualization (c) Zoomed in on structure (d) Conversion

Figure 3: All representations of the energy transport result generated by the minimum cost flow algorithm. (a) The Total Energy Flow clearly shows the decrease of energy along the x-axis. (b),(c) The Transport Glyph gives a detailed look of a region of interest (the beam) and the surrounding energy flow including the composition of the energy. (d) Finally, the Conversion shows the energy exchange between kinetic and internal energy for the entire domain.

values. For the nozzle data set, all necessary parameters were known and implemented accordingly. Details can be extracted from the publicly available data set (see Sec. 4). The calculations for the minimum cost flow algorithm and the subsequent visualization were performed on an Intel i7-2600K without multithreading. In our implementation, we used Google's implementation of a minimum cost flow solver to determine the energy flux from our constructed graph.

In the following, we give a short summary of the data sets, and present our examination results for our visualization and the implemented mappings in sections 6.1, 6.2 and 6.3 respectively. We then discuss its computational performance (section 6.5), and finally present our expert study (section 6.4).

6.1 Beam in Cross Flow

For this dataset, many of the defining parameters for the energy transport problem are known, and can be directly implemented. The foam-extend-3.1[2] package for flow simulations was used to generate this simulation data set. It is a fork of the well-known OpenFOAM package[3]. The constants employed are $p_0 = 100, T_0 = 100, R = 1, \gamma = 1.4$. The images in Fig. 2 show volume renderings of the pressure, density and velocity magnitude fields, respectively. We produced a result for all the representations explained in Sec. 5, which can be seen in Fig. 3.The image in Fig. 2(a) shows the Total Energy Flow Amount moved into a destination cell. One can clearly see the diminishing energy transport from the inlet (on the left) towards the outlet (on the right). This immediately implies that energy, being only injected on the inlet boundary condition, dissipates from the domain, at approximately constant rate. This suggests that the energy leaves the domain through the top boundary (being an outflow), and the lower boundary (being a "no-slip" wall). However, this visualization type does not show a detailed transport of energy, nor any energy composition information for the region of interest in the data set, the beam structure in the center.

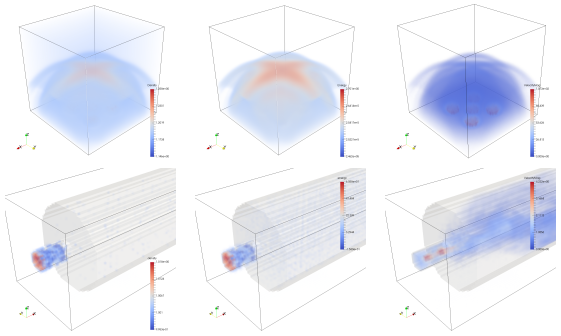
For this, the Transport Glyph can be employed. The images 3(b) and 3(c) show glyphs depicting the individual subgraphs for manually selected cells around the structure. Since in this simple case the region of interest is known, a manual selection is possible. The large sphere in the center implies that for the lower middle part of the structure, not too much energy is moved, and the small cone pointing into the sphere shows the small amount of energy entering the cell. The other cones and spheres can be interpreted in a similar fashion, and it can be easily seen that the energy flow is similar to the mass flow. However, considering Fig. 2(c), differs significantly, as the energy has other modes of transportation than convection. The visualization gives a detailed view of the transport for individual cells, and is applicable when the region of interest is small. Finally, conversion per cell is rendered in the last figure showing the Energy Conversion. Employing the volume rendering technique allows for a complete overview of the whole domain.

6.2 5jets

For the 5jets data set, the producing parameters are not known. With our technique, only knowledge of the boundary conditions is necessary. Information about the boundary condition can be derived by assuming a given boundary condition, and testing if parameters can be found that match the values on cell boundaries. After empirically determining the boundary conditions to be outflow on all sides and the top, and a slip condition for the bottom, as well as 5 cir-

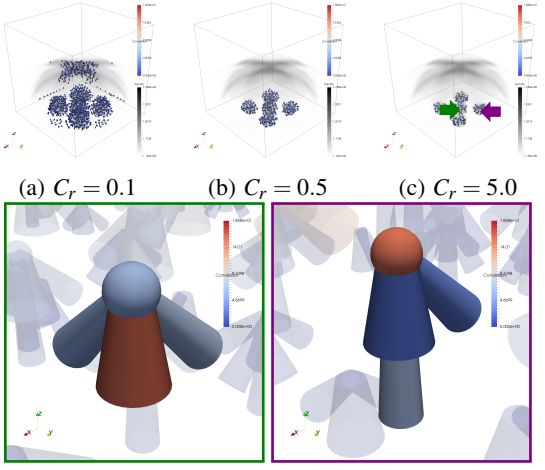
[2] http://www.extend-project.de/

[3] http://openfoam.org/



(d) Density (e) Internal energy (f) Momentum

Figure 4: Volume renderings of the input fields using common visualization techniques. The images show the state variables for the 5jets (top row) and the nozzle (lower row) data set from the destination time step for comparison to our approach shown in Fig. 5 and Fig. 6.



(a) $C_r = 0.1$ (b) $C_r = 0.5$ (c) $C_r = 5.0$

(d) Detailed glyphs

Figure 5: Energy Conversion glyphs for the 5jets data set for several conversion thresholds $C_{\max}$. (a) - (c) An overview of the energy conversion can be created by setting a low threshold. (d) Zooming in, one can adjust the size and density of the glyphs to reveal the details of the energy flow. While the conversion for the left cell is heavily influenced by the energy flow from below, for the cell on the right it is rather low, with highest conversion taking place for the energy remaining in the cell.

cular patches being the inlets for the jets, we modeled our setup accordingly. Since the inlet velocity was found to vary over time, we also employed further adjustments to the empirically determined inlet velocity. With our approach still being very stable against these inaccuracies, these assumptions allow for an analysis. In Fig. 4 we show how the destination time step's state variable fields look like in an internal energy, density and velocity magnitude volume rendering.

Fig. 5 shows the application of the Importance-Weighted Glyph Placement. A low-opacity volume rendering of the density for the destination time step has been added for context. The utility of the (interactively adjustable) conversion threshold parameter $C_{\max}$ becomes apparent when used to explore the conversion distribution as a whole, and find areas with a fairly high (or low) conversion (top row, Fig. 5(a)-Fig. 5(c)). Zooming in on a high conversion area identified this way, energy conversion can be examined per cell. The glyph on the left of Fig. 5(d) receives a high amount of energy from the cells in the area below ($\rightarrow$ large diameter on cones), only the fraction transferred from the cell directly below is heavily converted between energy forms ($\rightarrow$ saturated red). The glyph on the right has little conversion and a low inflow of total energy from other cells. On the other hand, the energy remaining in the cell ($\rightarrow$ encoded in sphere), is subject to a fair amount of conversion.

6.3 Nozzle

The nozzle data set, created with OpenLB[4], is a good example of a more turbulent simulation. Parameters defining the simulation can be extracted from the examples bundled with the solver. In Fig. 4 we show the destination time step's state variable fields, similar to the 5jets data set. For context, the bounding geometry of the nozzle and tube are shown. Fig. 6 shows a distribution acquired using Importance-Weighted Glyph Placement. The density of the glyph coverage as well as the glyph size can be adjusted interactively to deal with potential clutter. Both color and opacity of the glyphs depict the Conversion value. This gives a good overview of the conversion, and also allows for a closer examination of the conversion-heavy regions in the center of the nozzle's jet. The Conversion (encoded in color) is particularly large right after the nozzle exit, as well as at the "tip" of the jet. Smaller cones and spheres at the nozzle imply a small transfer of energy, indicating that most of the kinetic energy is converted further along the jet.

[4]http://optilb.org/openlb/

6.4 Informal Study

For our study, we interviewed a flow scientist involved in the development of compressible flow solvers, working as Ph.D. student at TU Delft. On a daily basis, he uses various visualization tools to analyze the results. For this task, he develops flow solver algorithms within the OpenFOAM framework customized to scalable fluid-structure and fluid-fluid boundary interaction. He confirmed that the local-

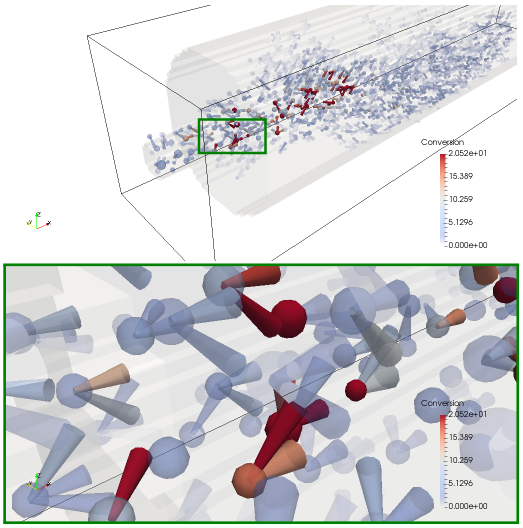


Figure 6: Our energy transport glyphs derived from time step 49200 → 49800. The outflow from the inner regions of the nozzle's jet toward the outer layer can be immediately derived. Also the energy 'dissipating' right after exiting the nozzle is rather low (small cone diameters in glyph) the conversion is very high (red color).

ity assumption for energy transport made in section 4 is valid. For the visualization type showing volume renderings of the total energy, we showed him the total energy flow for the beam in cross flow, as seen in Fig. 3(a). He appreciated that outflow through the top boundary condition, accounting for the loss of energy over the domain, can be extracted from the visualization. However, the domain scientist identified this visualization inadequately visualizes the energy flow behavior around the prominent region of interest, the structure in the fluid flow. This is addressed in the Glyph Coverage image (Fig. 3(c)), which we presented as a solution. The scientist really liked the clear visualization and the easy interpretation of the glyph, and had no problem extracting information about energy flow, when asked about the information gained. He also was intrigued about the volume visualization of the conversion ratio. In summary, the expert was clearly interested in the visualization, and appreciated the fact that the information is easily gained from both visualization types. He was especially intrigued by the Sparse Rendering / Importance-Weighted Glyph Placement variants for allowing both an option for overview while enabling a detailed examination for more interesting features. Overall, he expects the visualization in general to be very valuable, and will yield a significant improvement for the analysis of flow in general.

Data elements	**Beam**	**5jets**	**nozzle**
Number of nodes	28938	573440	211248
Number of edges	319665	6382336	5116252
Process Graph			
Normalization (4.3)	0.7 s	3.85 s	1.41 s
Centers (4.4)	1.02 s	18.25 s	14.43 s
Build graph (4.4)	0.42 s	8.49 s	6.55 s
Run solver (4.4)	10.31 s	624.10 s	557.30 s
Generate Mappings			
Total Energy (5.2)	0.05 s	0.92 s	-
Energy Conversion (5.2)	0.06 s	1.38 s	1.51 s
Weighted Glyphs (5.3)	-	2.13 s	2.10 s

Table 1: Timings for the different steps performed for preparing the problem and running the minimum cost flow algorithm. Once the fluxes have been determined, the various mappings can be generated and manipulated interactively. The relevant section is given in parentheses. Unmentioned timings were negligible ($\ll 10ms$).

6.5 Timings

There are several steps to compute during the execution of the algorithm. The timings for calculating the input energy distributions (i.e. the kinetic and internal energy from state variables) can be neglected ($\ll$ 1s). As the mesh may be unstructured with arbitrary cells, and varying over time (as is the case for the beam in cross flow), cell centers need to be recomputed for each time step, taking the arithmetic mean of the vertices defining a cell. The normalization of the total energy to exactly match contained energy between source and destination time step, as explained in section 4, is then performed. Next, the nodes and edges for the minimum cost flow algorithm are allocated and established. When the graph is established the algorithm can be executed to solve the problem. Finally, one or more visualization types can be chosen to be performed on the algorithm's result. Timings were taken for all data sets and the individual steps performed, the results are given in Tab. 1. The timings are clearly dominated by the minimum cost flow algorithm, which has a high complexity in the number of nodes and edges. Once calculated, the various mappings can be produced interactively. The prototype implementation has various ways for improving performance, in particular using additional physical constraints like fluid parameters to reduce the number of edges needed for the solution of the minimum cost flow problem.

7 Conclusion

We presented a technique to visualize the transport of and conversion between internal and kinetic

energy from arbitrary compressible flow simulations. For this purpose, we modeled our energy transportation problem as a graph, and solved it via a minimum cost flow solver, inherently respecting energy conservation. Here, we explicitly considered of various simulation parameters like boundary conditions and energy transport mechanisms. Based on the resulting flux, we then derived a local measure for the conversion between energy forms using the distribution of internal and kinetic energy. We finally employed different mapping approaches, and evaluated our technique by means of different simulation data sets and feedback by a domain scientist. In particular, we introduced glyphs to specifically depict convergence. By controlling glyph placement via the conversion ratio, we minimized occlusion problems by focusing on regions that are generally of interest. For future work, we aim to further study and evaluate our approach with additional types of simulations, as well as data from measurements obtained via experiments. Another promising improvement to enhance the quality of the total energy flux estimation, is deriving edge weights from given physical information, e.g. using the (easy to calculate) momentum flux to favor certain edges. For the Importance-Weighted Glyph Placement, glyph density and size could be automatically adjusted in a view-dependent fashion.

REFERENCES

Bonneel, N., van de Panne, M., Paris, S., and Heidrich, W. (2011). Displacement interpolation using lagrangian mass transport. *ACM Trans. Graph.*, 30(6):158:1–158:12.

Borgo, R., Kehrer, J., Chung, D. H. S., Maguire, E., Laramee, R. S., Hauser, H., Ward, M., and Chen, M. (2013). Glyph-based visualization: Foundations, design guidelines, techniques and applications. Eurographics Association.

Boring, E. and Pang, A. (1996). Directional flow visualization of vector fields. In *Visualization '96. Proceedings.*, pages 389–392.

Goedbloed, J. and Poedts, S. (2004). *Principles of Magnetohydrodynamics: With Applications to Laboratory and Astrophysical Plasmas*. Cambridge University Press.

Goldberg, A. V. and Tarjan, R. E. (1986). A new approach to the maximum flow problem. In *Proceedings of ACM Symposium on Theory of Computing*, STOC '86, pages 136–146, New York, NY, USA. ACM.

Goldberg, A. V. and Tarjan, R. E. (1990). Finding minimum-cost circulations by successive approximation. *Math. Oper. Res.*, 15(3):430–466.

Hasert, M. (2014). *Multi-scale Lattice Boltzmann simulations on distributed octrees; 1. Aufl.* PhD thesis, München.

Hlawatsch, M., Leube, P., Nowak, W., and Weiskopf, D. (2011). Flow radar glyphs-static visualization of unsteady flow with uncertainty. *IEEE TVCG*, 17(12):1949–1958.

Kraus, M. and Ertl, T. (2001). Interactive data exploration with customized glyphs. In *WSCG*, pages 20–23.

Kroes, T., Post, F. H., and Botha, C. P. (2011). Interactive direct volume rendering with physically-based lighting. techreport 2011-11, TU Delft. 1–10.

Landau, L. D. and Lifshitz, E. M. (1987). *Fluid Mechanics, Second Edition*. Butterworth-Heinemann, 2 edition.

Laramee, R. S., Schneider, J., and Hauser, H. (2004). Texture-based flow visualization on isosurfaces from computational fluid dynamics. In *IEEE TVCG / VisSym*, pages 85–90.

Leal, L. G. (2007). *Advanced transport phenomena: fluid mechanics and convective transport processes*. Cambridge University Press.

McLoughlin, T., Jones, M. W., Laramee, R. S., Malki, R., Masters, I., and Hansen, C. D. (2013). Similarity measures for enhancing interactive streamline seeding. *IEEE TVCG*, 19(8):1342–1353.

McLoughlin, T., Laramee, R. S., Peikert, R., Post, F. H., and Chen, M. (2010). Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1829.

Meyers, J. and Meneveau, C. (2012). Flow visualization using momentum and energy transport tubes and applications to turbulent flow in wind farms. J. Fluid Mech. (2013), vol. 715, pp. 335-358.

Ropinski, T., Oeltze, S., and Preim, B. (2011). Survey of Glyph-based Visualization Techniques for Spatial Multivariate Medical Data. *Computers & Graphics*.

Rubner, Y., Tomasi, C., and Guibas, L. (2000). The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121.

Sadlo, F., Üffinger, M., Pagot, C., Osmari, D., Comba, J., Ertl, T., Munz, C.-D., and Weiskopf, D. (2011). Visualization of cell-based higher-order fields. *Computing in Science and Engineering*, 13(3):84–91.

Sankey, H. R. (1905). *The energy chart : practical applications to reciprocating steam-engines*. Albert Frost and Sons.

Schroeder, W. J., Volpe, C. R., and Lorensen, W. E. (1991). The stream polygon-a technique for 3d vector field visualization. In *IEEE Conference on Visualization*, pages 126–132, 417.

Schultz, T. and Kindlmann, G. (2010). A maximum enhancing higher-order tensor glyph. *Computer Graphics Forum*, 29(3):1143–1152.

Tong, X., Zhang, H., Jacobsen, C., Shen, H.-W., and McCormick, P. (2016). Crystal Glyph: Visualization of Directional Distributions Based on the Cube Map. In *EuroVis 2016 - Short Papers*.

Üffinger, M., Schweitzer, M. A., Sadlo, F., and Ertl, T. (2011). Direct visualization of particle-partition of unity data. In *Proceedings of VMV*, pages 255–262.

Wittenbrink, C. M., Pang, A. T., and Lodha, S. K. (1996). Glyphs for visualizing uncertainty in vector fields. *IEEE TVCG*, 2(3):266–279.