Visitation Graphs: Interactive Ensemble Visualization with Visitation Maps

Anna-Pia Lohfink
Technische Universität Kaiserlautern, Germany

Christoph Garth
Technische Universität Kaiserlautern, Germany

Abstract

Modern applications in computational science are increasingly focusing on understanding uncertainty in models and parameters in simulations. In this paper, we describe *visitation graphs*, a novel approximation technique for the well-established visualization of steady 2D vector field ensembles using *visitation maps*. Our method allows the efficient and robust computation of arbitrary visitation maps for vector field ensembles. A pre-processing step that can be parallelized to a high degree eschews the needs to store every ensemble member and to re-calculate every time the start position of the visitation map is changed. Tradeoffs between accuracy of generated visitation maps on one side and pre-processing time and storage requirements on the other side can be made. Instead of downsampling ensemble members to a storable size, coarse visitation graphs can be stored, giving more accurate visitation maps while still reducing the amount of data. Thus accurate visitation map creation is possible for ensembles where the traditional visitation map creation is prohibitive. We describe our approach in detail and demonstrate its effectiveness and utility on examples from Computational Fluid Dynamics.

2012 ACM Subject Classification Human-centered computing → Visualization systems and tools

Keywords and phrases Uncertain flow visualization, Ensemble visualization, Visitation maps, In-situ

Digital Object Identifier 10.4230/OASIcs.iPMVM.2020.4

Funding This research was funded by the Deutsche Forschungsgemeinschaft (DFG – German Research Foundation) under contract 252408385 as part of IRTG 2057 Physical Modeling for Virtual Manufacturing.

1 Introduction

The incorporation of uncertainties into computational models is a core challenge in computational science on which quick advances have been made in the recent past. Due to growing computational ability, it has become straightforward to investigate the effects of model and parameter uncertainty through ensemble simulation. Models are realized multiple times and the ensemble of realizations is used as a basis for analysis. However, the amount of data resulting from ensemble simulations with a high number of ensemble members can be very large and of prohibitive size to be able to interactively (in terms of reaction times) visualize and analyze it. An often-used enabling methodology in this context are *in situ* techniques that derive visualization and analysis at data production time and store reduced-size artifacts such as images, videos, or other reduced representations for further inspection.

In this setting we consider the use of *visitation maps* (Figure 1). These are used to elucidate the transport behavior that is described by an ensemble of 2D vector fields. Such visitation maps are easy to use from an analyst's point of view They can be viewed as a generalization of integral curve techniques that form the basis of flow visualization. Consequently, their computation is in practice simple and, given an initial distribution, utilizes Monte Carlo sampling of trajectories across a vector field ensemble. For large (or even medium sized)

2nd International Conference of the DFG International Research Training Group 2057 – Physical Modeling for Virtual Manufacturing (iPMVM 2020).
Editors: Christoph Garth, Jan C. Aurich, Barbara Linke, Ralf Müller, Bahram Ravani, Gunther Weber, and Benjamin Kirsch; Article No. 4; pp. 4:1–4:20

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

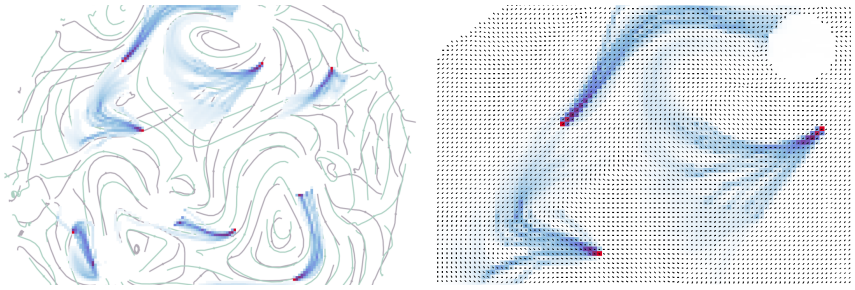

Figure 1 Left: Streamlines of two ensemble members are shown in the background to give an overall impression of the flow. Right: Glyphs calculated from the visitation graph surround the visitation maps.

vector field ensembles, however, this straightforward approach becomes prohibitively costly since running times are not adequate for interactive exploration of uncertain vector fields and all ensemble members are required for sampling.

In this work, we propose *visitation graphs* as a novel intermediate representation of the flow behavior in an ensemble of two-dimensional vector fields and show their usefulness in enabling the interactive exploration of vector field ensembles. We illustrate that visitation maps for arbitrary initial distributions can be derived quickly from the visitation graphs. Furthermore, visitation graphs give a less lossy reduction method than downsampling. Thus, visitation graphs offer the possibility to explore large ensembles interactively, with the flexibility to trade off approximation quality for representation size. In particular, our contributions are:

- After reviewing related work in Section 2, we define visitation graphs (Section 3.1) and discuss how they can be constructed from vector field ensembles (Section 3.2).
- We provide in Section 3.3 an algorithm to quickly approximate a visitation map for an arbitrary initial distribution from a visitation graph and discuss the approximation of visitation maps using shorter visitation maps.
- Details on space requirements and data reduction are given in Section 3.4 and we investigate interactive visualization modes on ensembles enabled by our approach (Section 3.5).
- The behavior of our method under variation of different parameters is examined in Section 4.
- Using a representative set of application datasets, we empirically demonstrate the usefulness of our approach and evaluate its approximation, reduction and runtime properties in Section 5.

Overall, we aim to demonstrate that visitation graphs can function ideally as artefacts of an in situ approach for vector field ensembles that still permit nearly full flexibility during post hoc exploration.

2 Background and Related Work

The present work is focused on the visualization of uncertain vector fields and vector field ensembles, as well as in situ visualization techniques. In the following, we briefly review relevant prior work and background material and discuss differences to our approach.

Visitation Maps. Visitation maps are an established and intuitive visualization to analyze distributions of particle trajectories arising from vector field ensembles or vector valued random fields.

Previous authors have defined visitation maps in an ad hoc manner as the empirical distribution within each cell in the domain obtained as the percentage of generated trajectories of a given length and with a given start point passing through the cell [4, 22, 21]. However, the resulting ad hoc calculation renders the use of visitation maps challenging due to computational effort inherent in their numerical approximation. Given an initial condition (i.e. starting location or initial distribution), the visitation map is typically estimated through direct sampling. For a faithful approximation, a high number of samples is required. For small datasets, parallel computation using for example GPUs can be leveraged to achieve interactive re-computation upon modification of the initial condition. For example, Bürger et al. demonstrated an interactive visualization of ensemble vector fields with visitation maps using GPU-based Monte-Carlo particle tracing [4]. However, for larger datasets, trajectory computation is a difficult problem and has to be handled through non-interactive out-of-core techniques [29] or parallel algorithms [34]. In contrast, our approach performs the sampling calculations in a pre-processing step that can be executed in situ, and stores the result as a compact visitation graph from which visitation maps can be derived quickly in a later state. Thus, we eschew the problem of ensembles being too large to handle interactively. In addition, a tradeoff between pre-processing time, storage requirements, and visitation map accuracy can be made.

A recent variation of visitation maps was proposed by Ferstl et al. as streamline variability plots [12]. In contrast to visitation maps, variability plots are generated by projecting confidence ellipses for obtained streamline clusters in PCA space to domain space. This yields an envelope for most obtained streamlines, together with a calculated mean streamline. For large datasets, the challenge to integrate large numbers of streamlines remains the same.

Representing Vector Fields as Graphs and Webs. Representing information on a given vector field in a graph structure, as it is done in this work, was earlier considered in the forms of Flow Graphs by Nouanesengsy et al.[29] and as Flow Webs by Xu et al. [42]. The Flow Graph contains a node for each block in an underlying grid, and each two neighboring blocks are connected via a weighted edge. The weight of the edge connecting two blocks is determined as the probability that a seeded particle in one of the blocks is transported to the other one by the flow. Afterwards, this flow graph can be used to estimate the workload for each block in parallel streamline calculation.

While Flow Graphs are an efficient approach to estimate workload, they are not suitable for visitation map approximation, as visitation maps calculated from such graphs suffer from the lack of variety in possible directions in the graph. In a Flow Graph each cell is connected only to its four direct neighbors, which results in an extremely coarse approximation of the visitation map.

In Flow Webs, regular axis-aligned sub-regions of the given domain represent nodes in the graph. By sampling streamlines backwards through the regions, links and weights between different sub-regions are determined. While Xu et al. do not consider uncertainty, our approach can be viewed as an adaptation of the Flow Web concept to ensembles to create visitation maps, based on a generalization of the Flow Connection Matrix (adjacency matrix of the Flow Web).

As opposed to the Flow Graphs by Nouanesengsy et al., Flow Graphs by Ma et al. [27] are a tool for streamline and pathline exploration of 3D flow fields. Here, nodes in the graph do not only represent spatial regions but also streamlines in the field. Edges between nodes are assigned various interpretations depending on the kind of nodes they interconnect.

Adaptive transition graphs for vector fields have been employed by Szymczak to calculate Morse connection graphs for piece wise constant vector fields on surfaces [38]. His method of refining transition graphs in an adaptive manner according to strongly connected components can be interpreted as perturbing the vector field near trivial Morse sets to remove recurrent features in the resulting Morse connection graph.

Multiple approaches employing graphs to track and visualize states and state transitions in time-varying vector fields have been proposed, for example by Gu and Wang [14] and by Jänicke and Scheuermann [19]. A recent survey on the utilization of graphs in visualization was given by Wang et al. in [39].

None of the aforementioned graph techniques are aimed at creating visitation maps and providing sufficient information on an underlying ensemble or vector valued random field to obtain adequate visitation map approximations.

Visualization of Uncertain Vector Fields. Uncertainty in vector fields has been recognized as a pressing research problem in scientific visualization for several years and still is a substantial challenge. Recent surveys can be found in [7] and [15]. Heine et al. discuss topology based visualization methods that address uncertainty in [16].

Often, uncertainties arise from multiple results of experiments with varying input parameters collected in ensembles. Examples for uncertainty visualization based on ensembles are: showing ensemble members vanishing over time [6], enabling the user to compare single members to the whole ensemble using glyphs [35] and summarizing ensemble members while highlighting outliers and median in Contour/Curve Box Plots [40, 28]. The topology of ensembles in two and three dimensions was determined by Otto et al. in [31] and [30]. Hummel et al. gave a comparative visual analysis for ensembles of time-varying vector fields using a Lagrangian framework [18]. A two dimensional comparative visual analysis was presented by Jarema et al. in [20].

Uncertainty arising from interpolation and prediction of missing measurements was treated using tubes of varying size [3], glyphs and parallel coordinates for MR spectroscopy data [11, 10], flow radar glyphs for time dependent vector fields with uncertainty given as an interval [17] and using colormapping and line glyphs for uncertain isosurfaces in geosciences [43]. Uncertain predicted multivariate data was visualized by Berger et al. using parallel coordinates and scatter plots [2]. Random fields were treated using fuzzy set theory, volume rendering on trapezoidal possibility distribution [13] and by visualizing isosurfaces in uncertain scalar fields [32, 33]. A FTLE like method was presented by Schneider et al. [36].

Random fields are a stochastic uncertainty model. Following the approach of in situ data reduction by summarizing statistics of certain properties, random fields frequently arise in in situ pre-processing. While in mathematics, the generalization of stochastic processes to higher dimensions is called *random field*, different names have been used in the visualization community up to now. Otto et al speak of *uncertain vector fields* [31], Ferstl et al. use the term *ensemble of vector fields* [12], Sevilla-Lara et. al speak of *distribution fields* in computer vision [37] and Love et al. use the more general term *spatial multivalue data* [26]. In [8], Dutta et al speak of statistical information.

Our method is suited to the uncertainty visualization of both, vector field ensembles and random fields.

Approximation of integral curves. Different approaches for the approximation of a longer particle trajectory by a sequence of (certain) flow maps have been examined. Agranovsky et al. give a two phase approach extracting a basis of known pathlines in situ and calculating arbitrary integral curves post hoc from the extracted results [1]. Two similar phases are

presented in this paper, carrying the idea to approximate longer curves using smaller ones over to visitation maps. To our knowledge, the generalization to uncertainty and thus to visitation maps in this paper is a new contribution.

In Situ Analysis and Visualization. In situ analysis, i.e. data analysis taking place at creation time while the data is still in memory, is an attractive possibility to handle huge datasets that are too large to store. In addition, slow data output is avoided and data can be pre-processed at creation time. As computational power increases and thus simulations of massive ensembles become common, the need for effective in situ processing is ever more pressing.

In this context, in situ capabilities for leading visualization libraries and tools such as Paraview [9] and VisIt [25, 41] were developed, allowing in-situ analysis and visualization of simulations.

While in situ visualization provides crucial insights in the simulation behavior at runtime, interactivity and thus exploration of the data is rarely possible. Pre-processing simulation data in situ and benefiting from reduced data in flexible and scalable post hoc analysis is a possibility to combine the advantages of in situ and post-processing. This new paradigm is illustrated for example by Childs in [5]. Several approaches were already presented that follow this paradigm. For example Lakshminarasimhan et al. proposed ISABELA for in situ sorting and error bounded compression in [23].

A further approach to reduce data in situ is to summarize statistics of properties of interest during the simulation. Afterwards suitable visualization approaches are used for data exploration. This was done recently by Dutta et al. in [8]. A recent survey was given by Li et al. in [24] grouping current research on in situ data reduction in a spectrum between lossless and very lossy techniques.

Following the in situ, post-processing combination in [5], our approach processes steady 2D vector field ensembles in situ to construct a *visitation graph*. With an upper bound for storage requirements that does not depend on the number of ensemble members, this visitation graph is an ideal structure to summarize ensembles. Furthermore, the visitation graph resolution can be chosen freely and does not have to mirror the resolution of ensemble members. This provides a means for data reduction, tailored to the task of creating visitation maps for interactive exploration of ensembles potentially too large to store (cf. Section 3.4 for additional discussion). Based on the visitation graph an interactive post hoc exploration of the ensemble using visitation maps is possible once the simulation and pre-processing are complete.

3 Method

In the following, we introduce the *visitation graph* as the core concept of our work and then proceed to illustrate its use for visitation map computation and visualization.

3.1 Visitation Graphs

We consider two-dimensional vector field ensembles or random vector fields. These are partitioned into cells of their domain of definition $\mathcal{C} = \{c_1, \dots, c_M\}$. A visitation map $\mathcal{V}$ represents for each c_j the distribution of streamline samples of integration length $T > 0$ that "hit" the cell c_j, i.e.

$$\mathcal{V}(c_j) \;:=\; P(S(X,T) \cap c_j \neq \emptyset), \tag{1}$$

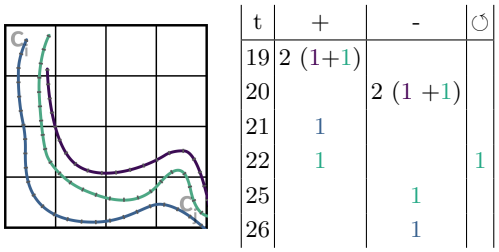

Figure 2 Visitation graph creation: Streamlines starting in cell c_i that pass by c_j are considered and all events are tracked with time point. Table entries are colored according to the triggering streamline.

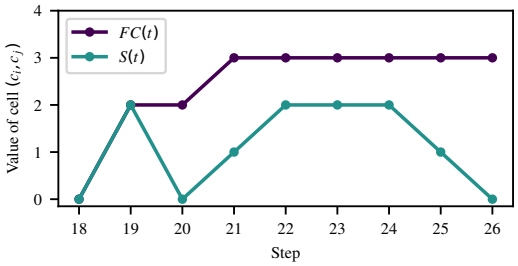

Figure 3 Values of the Flow Connection Matrix and snapshot in the example in Figure 2 (omitting normalization). $FC(t)$: number of streamlines that passed by, $S(t)$: number of streamlines that are currently in the cell.

where $S(X,T)$ is the set of all points of the streamline sample, and its seed point X is a random variable with a fixed initial distribution (in traditional visitation maps X is a uniform distribution in a single cell). In practice, this probability is computed through Monte Carlo sampling as an average over a set of streamline samples.

To speed up visitation map generation, we propose an approximation based on in situ calculated *visitation graphs*.

We define the visitation graph as a directed graph $G(V,E)$ whose nodes V are the cells of the partition, i.e. $V = \mathcal{C}$, and E contains an edge between c_i and c_j if and only if a streamline sample of length T' starting in c_i hits c_j during integration. For each edge (c_i, c_j), streamlines starting in cell c_i are considered. Their (re-)entry and exit in cell c_j are recorded.

Streamline samples are approximated as a polyline with k points, originating in every cell c_i. In practice, we (re-)use the computational grid of the ensemble as the partition $\mathcal{C}$, and approximate streamlines using a second-order Runge-Kutta scheme with fixed step size (hence k steps of step size Δt). Both of these aspects of the technique are easily modified.

Information is collected similar to the Flow Connection Matrix [42]. However, the visitation graph keeps track of more information than just the number of passing streamlines, namely the events and timepoints.

From the visitation graph, visitation maps can be assembled. We explicitly point out that, to approximate a visitation map of length T, it is not required that $k\Delta t := T' = T$ holds; in practice, streamline samples can be much shorter than the desired visitation map length, at the cost of accuracy. This allows to trade off accuracy and speed in visitation map creation for speed in visitation graph creation and storage size of the visitation graph, since shorter streamlines yield less events that need to be stored in the graph. Furthermore, the visitation graph resolution is not necessarily the same as the resolution of ensemble members. This provides a method for data reduction tailored to the task of creating visitation maps for interactive exploration of ensembles potentially too big to store. See Section 3.4 for more details on data reduction using visitation graphs.

3.2 Efficient Computation of Visitation Graphs

For each cell in the visitation graph connected to the considered start cell via an edge, a list of steps and 3-tuples is created recording the occurrence of the events of interest. This is done for each edge (c_i, c_j) by tracking how many streamlines starting in cell c_i enter cell c_j at each step (+), leave cell c_j at each step (-) and re-visited cell c_j at each step (↻). The last entry is obtained by keeping track for each cell whether the considered streamline has already entered c_j or not. An illustration of visitation graph generation can be found in Figure 2.

During visitation graph creation, the number of randomly seeded streamlines started in each cell can be determined either based on the size of the cell (as done in [42]) or on the estimated contribution to the final visitation graph. While the first option is straight forward, the latter is achieved as follows: the number of cells n_h that have been hit by any streamline that was integrated from the considered start cell is determined and a size for streamline bunches is set as parameter s_b. Every time the integration of a bunch of streamlines is completed, it is decided whether one more will be started or not using

$$\frac{n_\mathrm{h}}{s} < \frac{1.0}{s_b}. \tag{2}$$

Where s is the total number of integrated streamlines from the considered start cell. If expression 2 evaluates to false, the next bunch of streamlines is integrated, else the visitation graph creation for the considered cell stops. This stop criterion can be interpreted as follows: it is unlikely that a streamline from the upcoming bunch of streamlines passes by a cell that was not passed before since the number of passed cells per integrated streamline is smaller than one out of a bunch of streamlines.

Handling two ensemble members is independent. Hence, this calculation can be done in situ. For every ensemble member, entries in the visitation graph are updated or added. While the visitation graph can be built completely in situ, it does not have to. In case of ensemble members occupying most of the available memory storage, it might be impossible to create the whole visitation graph simultaneously. In this case, single members or even single cells can be processed in situ and then be stored. In a post processing step all partial results are combined to the final visitation graph. Depending on the available memory every possible storing frequency in between "storing for each cell or member" and "storing once at the end" can be used providing a tradeoff between postprocessing time and memory requirements.

3.3 Efficient Approximation of visitation maps from visitation graphs

Given an initial distribution of cells or a single start cell, visitation maps can be effectively calculated from the visitation graph as follows.

Using the visitation graph consisting of M nodes, two different matrices can be generated for every timepoint $t \leq T'$: The *Flow Connection Matrix* $FC(t)$ as described in [42] and the *snapshot* $S(t)$. The Flow Connection Matrix is an $M \times M$ matrix given by the adjacency matrix of the visitation graph where the weight of edge (c_i, c_j) is given by the probability that a streamline starting in cell c_i passes by cell c_j before timepoint t. The entries are calculated as the sum of the differences of first and third entry of the stored 3-tuples until t, divided by the total number of integrated streamlines starting in c_i. While the first entry provides the information how many streamlines from c_i have entered c_j, the third entry ensures that re-entering streamlines are not counted twice. Note that Xu et al. generated the Flow Connection Matrix only for infinitely many streamline steps, thus our approach constitutes a generalization of their graph structure.

Entry (c_i, c_j) of the $M \times M$ dimensional *snapshot* matrix $S(t)$ specifies the probability that particles starting in c_i are in c_j after integration time t. It is calculated as the sum of the difference of first and second entry of the stored 3-tuples until t, divided by the total number of streamlines starting in c_j. Thus every time a streamline enters c_j, the snapshot entry increases by one and as the streamline leaves c_j, the snapshot entry decreases by one. This is done regardless of re-visiting. Examples for the calculation can be found in Figure 3.

Note that when generating Flow Connection or snapshot matrices from the visitation graph, the total number of streamlines started in the considered initial cell is required. Estimating the number of required streamlines, this number varies between cells. However, it

is not necessary to store this number separately since every streamline is recorded as entering at timepoint 0 in the start cell itself. Thus the total number of streamlines per start cell is easily available in the visitation graph.

Having an integration length T' that is independent of the final visitation map length T, the desired visitation map might have more or less steps than have been integrated while pre-processing. Being able to generate a visitation map based on streamlines shorter than the final map provides a tradeoff between accuracy on one side and shorter pre-processing time and storage savings on the other side. Hence two scenarios for visitation map creation exist: Either $T \leq T'$ or $T > T'$. If $T \leq T'$, the visitation map starting in cell c_i is given by the row representing c_i in the Flow Connection Matrix. That is the exact visitation map value $\mathcal{V}(c_j)$ is given in entry (c_i, c_j) of $FC(T)$. Considering possibly multiple start cells with a start distribution, a M-dimensional start vector v_0 is created holding the start probability for each start cell. This vector is then multiplied with $FC(T)$. Each entry of the resulting vector represents one cell in the grid and holds the visitation map values. The resulting visitation map gives the identical result as a traditional visitation map based on ad hoc created streamlines advancing for time T. If, on the other hand the desired visitation map length exceeds the number of pre-processing steps ($T > T'$), the visitation map is not given in the Flow Connection Matrix but needs to be "assembled" as follows from multiple $FC(t_i)$ where $\sum_i t_i = T$.

The *law of total probability* states that if $\cup B_i = \Omega$ is a countable partition of the entire sample space, then for any event A it holds:

$$P(A) = \sum_i (P(A \mid B_i)P(B_i) \tag{3}$$

For clarity of notation let in the following S, A, and B be arbitrary cells in $\mathcal{C}$. Considering the partition consisting of events

$A_t^S :=$ After time t, the considered streamline starting in cell S ends in cell A

for the fixed integration time in pre-processing T' the law of total probability gives

$$P(B_T^S) = \sum_{A \in \mathcal{C}} P(B_T^S \mid A_{T'}^S)P(A_{T'}^S) \tag{4}$$

Where the event B_T^S is an event considering streamlines after time $T > T'$, thus no calculations for this event have been performed during pre-processing. To estimate $P(B_T^S)$, the conditional probability $P(B_T^S \mid A_{T'}^S)$ is approximated using $P(B_{T-T'}^A)$ in equation 4. So the probability that the considered streamline that starts in S ends in B after time T while ending in A after time $T' < T$ is approximated by the probability that a considered streamline starting in A ends in B after $T - T'$ steps (See Figure 4 for further explanation). This gives

$$P(B_T^S) \approx \sum_{A \in \mathcal{C}} P(B_{T-T'}^A)P(A_{T'}^S). \tag{5}$$

Since $P(B_{T-T'}^A)$ considers an integration time strictly smaller than T, the considered number of steps is potentially smaller than the number of pre-processing steps until time T' such that pre-processed results are available. If however $T - T' > T'$, the same approximation is repeated until the integration time is smaller or equal to T'.

This approximation in general overestimates $P(B_T^S \mid A_{T'}^S)$. Equality is only given if $P(A_{T'}^S) = 1$ since then $P(B_{T-T'}^A)$ is independent of the point in cell A reached by the streamline from S in time T'. Thus every approximation step will induce an error in the

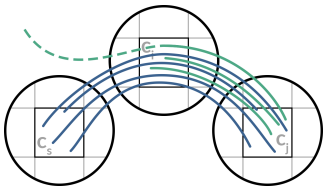

Figure 4 The probability of a streamline path $c_s \to c_i \to c_j$ in time T while in c_i after T' is approximated by the probability of the streamline path $c_i \to c_j$ in $T - T'$. This leads in general to overestimation since every streamline path $c_s \to c_i \to c_j$ is also a path $c_i \to c_j$, but not all streamlines from c_i to c_j pass by (or start in) c_s before entering c_i (f. ex the dashed streamline).

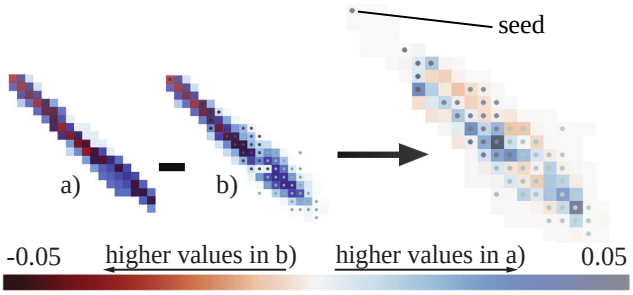

Figure 5 Traditional visitation map a), approximated based on visitation graphs b) and difference between both. $T' = 50$, $T = 160$. Nonzero entries of the start vectors v_i are marked from black to light grey for increasing i. After the first 50 (identical) steps, the approximated visitation map is re-started from all cells containing particles at step 50 (marked in grey). Resulting in an overestimation at the edges and an underestimation in the center. The colormap was chosen to present positive and negative differences and is not intended for visitation map presentation.

resulting visitation map (cf. Figure 5). This theoretical result is implemented as follows: The exact visitation map after time T' is calculated as described above using the start vector v_0. Having reached the maximal calculated step, a new start vector $v_1 = v_0 \cdot S(T')$ is calculated. Using this new start vector holding the positions and probabilities for particles starting in the considered start cell(s) under the considered start distribution after T' steps, the proceeding visitation map is calculated using $FC(T - T')$. The results are combined ensuring to not have doubled results in nonzero entries of v_1. This is repeated m times until $T - mT' \leq T'$.

Illustratively, the visitation map is followed as far as it was pre-computed, then new visitation maps are started at the end incrementally until the desired number of steps is reached.

While a smaller number of pre-processing steps induces errors in calculated visitation maps, pre-processing time becomes shorter and storage requirements of the resulting visitation graph become smaller. This tradeoff is illustrated by experiments in Section 4. Visitation map creation on the other hand becomes more expensive the more snapshots and Flow Connection Matrices are calculated, thus the calculation time for visitation maps from the visitation graph decreases with increasing pre-processing integration time T'. In general, calculation times can be reduced using parallelization: Integrating streamlines from every cell in the visitation graph to determine edges can be heavily parallelized since considering different cells is completely independent. Provided ensemble members reside on different nodes of a cluster, these members can be processed independently on their nodes. The final visitation graph can then be assembled using the parallel reduction approach resulting in logarithmic time savings. In an informal experiment, we observed nearly ideal speedup (7.96x

for 8 CPUs) for streamline integration. The assembly of 10 partial results was executed on 2 CPUs with a speedup of 1.2. In addition, the calculation of Flow Connection and snapshot matrices for visitation map creation can be naturally parallelized for every entry in the resulting matrix. While these matrices are potentially of very high rank, the fact that entries of $FC(t)$ and $S(t)$ can be calculated independently can be used to generate only rows that are required for the computation (that is the rows whose entries in the start vector are nonzero) and save them in a sparse format. Especially for the first steps in visitation map creation where the start vector contains only the chosen start cells, the number of needed rows is very small compared to the total number of rows. If however the resulting matrices become too large, a shorter visitation map or a coarser visitation graph need to be considered.

3.4 Space Requirements and Data Reduction

Visitation graphs are an optimal way to store, and potentially reduce, data in order to explore the data with visitation maps later on. The maximal storage requirements of the visitation graph representation are determined by the ensemble resolution $m \times n$ and T'. As such, they are independent of the number of ensemble members. An upper bound for the number of outgoing edges is $m \cdot n$, which is far from being tight. In reality, the number of outgoing edges is much smaller since it is impossible that every cell in the grid is passed by streamlines from every other cell. Also edges are only possible between cells that can be reached from another by a streamline of set length, thus only cells in a given, field dependent radius around each cell can be connected. At most $T' \times 4$ numbers need to be stored per edge. This upper bound will not be reached in realistic examples and most of the information stored for edges will contain far less than T' tuples (see Section 5). Thus, facing ensembles with a high number of members compared to the number of cells per ensemble, the visitation graph will save space without any additional data reduction. For other ensembles, multiple options leading to data reduction are available:

Visitation graph resolution. The visitation graph resolution is independent of the original resolution of the ensemble. Thus, a coarser resolution can be chosen. While a coarse visitation graph contains only nodes corresponding to a coarser grid, it is based on streamlines that are created based on the high resoluted ensemble. Carrying over the information from the fine grid to the coarse one, coarse visitation graphs are superior to simple downsampling of the grid. See experimental results in Section 4 for a comparison.

Timestep selection. Depending on the application, it is often not necessary to have access to visitation maps for every possible timestep. To exploit this, the temporal resolution of available visitation maps can be restricted. Defining a frequency F at which visitation maps should be available, events don't need to be stored for every timestep but for every frequency cycle. All events occurring between two cycles are summed up in one entry, reducing storage requirements since only every $F - th$ entry is stored. This reduction approach does not affect accuracy of visitation maps at available time steps.

Streamline length. Shorter streamline length at creation time of the visitation graph results in a lower number of events and thus reduces the storage requirements of the visitation graph. While it is still possible to generate longer visitation maps from the resulting visitation graph, the accuracy is reduced after every integration time of T'.

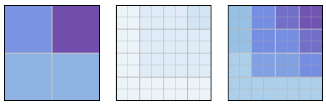

Figure 6 Appearance of the same area considering different grid resolutions. Left: coarse resolution, middle: no color compensation, right: color compensation. In every case the sum of the probabilities over all cells is one and the distribution is equal.

Experiments on data reduction using visitation graphs as well as experiments on storage requirements and their dependency on the number of ensemble members can be found in Section 4. An application of data reduction can be found in Section 5.

3.5 Application to Visualization

As outcome of the previous results, visitation maps from arbitrary start points with arbitrary initial distributions can be generated effectively from visitation graphs. This heavily contrasts with traditional visitation maps generated in ad hoc manner from integrated streamlines. They trigger a complete re-computation every time the start point is changed and considering multiple start cells with a given distribution results in vast computational effort. In addition, traditional streamline computation is always based on the whole ensemble, making this visualization approach unfeasible for ensembles with too many members to store. In our approach, to interactively create visitation maps from arbitrary start points, only the visitation graph needs to be stored. There is no need to store single ensemble members. Providing a tradeoff between storage requirement and accuracy, the visitation graph renders the exploration of ensembles possible that are prohibitive to store due to their size. Provided a visitation graph, two different modes for visitation map creation are available:

Standard mode: A single start point is selected. The visitation map of the desired length is calculated or approximated depending on T'.

Brush mode: Multiple start points following an initial distribution are given. The visitation map is calculated for all start points at once. Different kernel functions are available as possible start distributions, they are scaled such that the integral over all initial cells is one.

While for regular grids the visualization of visitation maps (ad hoc or visitation graph based) is a straight forward mapping of scalars to colors, additional challenges arise when considering grids containing differing cell sizes or the same grid is evaluated in different resolutions.

The probability that a streamline passes by a grid cell depends on the cell's size. Since the sum of the probabilities over all cells in an area remains constant independent of the resolution, small cells obtain smaller probability values in the visitation map than bigger cells. While from the mathematical point of view, this result is correct and intuitive, the resulting visualization might be misleading. Areas with smaller cells might appear less likely to be visited than coarser areas with the same probability. To handle this effect when plotting the visitation map, a compensation for the cell size can be used. The probability value assigned to each cell is multiplied by the size of the largest cell in the grid divided by the size of considered cell. This yields an intuitive visualization, see Figure 6. However the plotted values no longer represent the actual probabilities, so both visualization options are accessible in our implementation.

To give an orientation in the explored vector field ensemble, visitation maps can be combined with any other suitable static visualization as background. In the case of not having stored single ensemble members, this is hardly possible with common techniques.

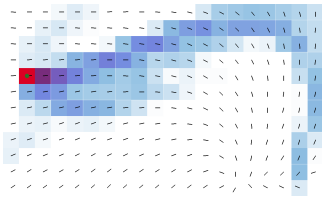

Figure 7 Glyphs indicating the average direction of connected cells in the visitation graph.

To be still able to give an impression of the underlying field, we developed visualization techniques based on the visitation graph, intended to support the visitation map visualization:

Glyph visualization: glyphs pointing towards the average direction of connected cells in the Flow Connection Matrix weighted with their probability can be drawn. That is, for every nonzero entry (c_i, c_j) in the Flow Connection Matrix, the midpoint of cell c_j contributes with weight $FG(t)(c_i, c_j)$ to the direction the glyph in cell c_i will point to. See Figure 7 for an example.

Degree visualization: In- and out- degree of each cell in the visitation graph is plotted using different colors. Like this, areas with large variance and areas that are passed by many different streamlines are visible, giving an orientation which areas might be interesting in to explore. See Figure 14 for an example.

While exploration of vector field ensembles is the main purpose of our approach, it can be employed to visualize random fields as well.

Given a probability space $(\Omega, \mathcal{F}, P)$, a random field of dimension $d \in \mathbb{N}$ is a family of random variables $\{Y(x, \cdot)\}_{x \in \mathbb{R}^d}$. A trajectory in a random field can be defined analogously to the certain case by an ordinary differential equation with a random field on the right hand side, which makes the equation a random ordinary differential equation (RODE).

$$\frac{dX_i(t,\omega)}{dt} = Y_i(X(t,\omega), t, \omega) \qquad \text{with } t \geq t_0, \omega \in \Omega, i = 1 \dots d. \tag{6}$$

RODEs can be solved using standard numerical approaches such as Runge-Kutta methods or multi-step methods. Thus, instead of integrating streamlines in every ensemble member, multiple streamlines are integrated using samples of the underlying random field.

4 Experiments

To evaluate the behavior of the system, different tests have been conducted on a standard workstation PC. The implementation was done in Python. Unless otherwise noted, a 50×50 testing ensemble containing 50 members created from normal distributions with constant $\mu = (1, 1)$ and varying correlation between -0.5 and 0.5 was considered for all tests in this section. The cell size was uniform and T' was chosen as 60 while $T = 100$.

Figure 8 illustrates the tradeoffs resulting in the generation of the visitation graph: a smaller number of streamline steps in pre-processing results in lower accuracy of the resulting visitation maps while time for visitation graph creation and storage requirements drop. Conversely, the longer the pre-processing integration time, the more accurate is the result while pre-processing runtime and storage requirements increase. The runtime for visitation map generation depends on T': With fewer timesteps stored in the graph, more calculations have to be executed to obtain the visitation map approximation (cf. Section 3.3). Thus, time needed for visitation map creation decreases with increasing accuracy. All runtimes are measured on a purely serial execution of the code. As ground truth, the visitation

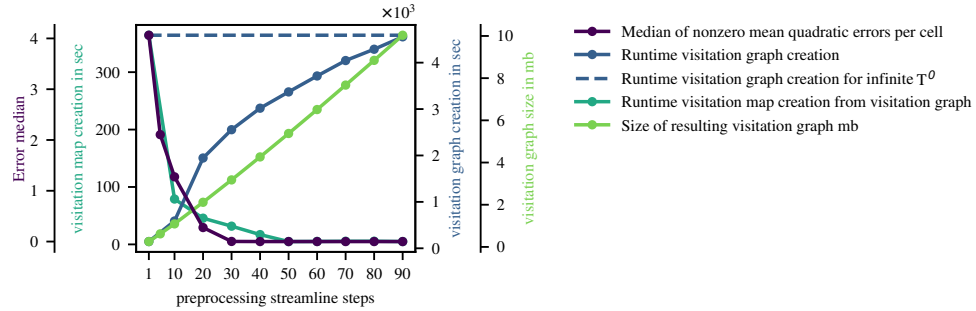

Figure 8 Visitation graph tradeoffs: With increasing T', accuracy, storage requirements and pre-processing time increase while the required time to create visitation maps decreases. The median of nonzero errors is shown since low errors are far more common than high errors.

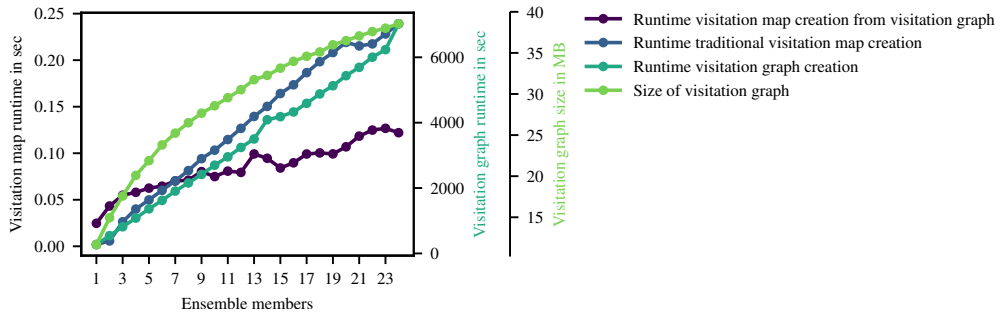

Figure 9 The more members are considered, the more beneficial is our method. The difference in runtime increases and already for very few ensemble members it outperforms the traditional ad-hoc creation. Storage requirements for visitation graphs converge towards a fix maximal size. Visitation graph creation grows linearly with the number of processed ensembles.

graph with infinite T' and corresponding visitation maps are considered. That is every generated streamline is pursued until it either ends in a critical point or leaves the grid. For less streamline steps, the ground truth visitation graph is pruned at the desired step to avoid differences resulting from randomized streamline seeding. The mean squared error is calculated per cell as the difference between traditional visitation map starting in this cell and the visitation map generated from the visitation graph. As the number of pre-processing steps reaches the number of steps taken in the visitation map, the error drops to zero. No assembly is required and the exact visitation map is calculated.

In addition to the advantage of in situ pre-processing obviating the need to store every ensemble member to be able to calculate visitation maps, Figure 9 illustrates that already for a small number of ensemble members, the visitation map creation from the visitation graph is faster than the ad hoc creation. The larger the ensemble, the more beneficial is our method. This observation is strengthened when considering multiple start cells (cf. Figure 10). To obtain accurate results for traditional visitation maps with multiple start cells under a delocalized start distribution, even more realizations are necessary to achieve adequate sampling.

To examine visitation graphs as an alternative to downsampling, we executed a further experiment: Our approach using coarse visitation graphs to obtain visitation maps was compared with the traditional approach to first sample down the data, then store it and create visitation maps in the traditional way. All approaches result in visitation maps of the same output resolution. A real-world and an analytical example were tested. The resolution was reduced from 168×168 to 80×80 and 336×168 to 160×80 respectively. We compared:

Our Method: A visitation graph of output resolution is calculated on high resoluted input data. Using the visitation graph visitation maps are created.

Traditional: High resoluted input data is downsampled to output resolution. Visitation maps are calculated in the traditional way.

Ground Truth (Test on analytical data): Exact streamlines in the analytical field are used to create exact visitation maps of the given output resolution.

Ground Truth (Test on real-world data): Visitation maps are calculated in the traditional way on high resoluted input data, the final visitation map is sampled in the given output resolution.

Figure 11 holds the results for the real-world example. Because of the exact ground truth, errors of our method in the analytical example were higher. While they reside in the same range as the errors of the traditional approach, the maximal deviation only exceeds the one of the traditional approach after $T = 160$. For $T = 100$ again the error is much smaller than the one generated by traditional approximation. For $T \in [110, 160]$ our method gives a similar mean deviation and a smaller maximal deviation. So again our method outperforms the traditional approach not even for $T = T'$ but also for additionally approximated visitation maps for $T > T'$ up to a limit.

5 Results and Applications

In this section, results for real-world applications are given. While our contribution is a new calculation method for visitation maps, using visitation maps for visualization is well-established. Hence, this section aims more for giving details on applications of our method than on illustrating usefulness of visitation maps in general. The following datasets were used:

Industrial Stirring: Mixing in a stirring apparatus was simulated for 20 slightly differing viscosities resulting in a 2D flow field ensemble of size 168×168. The device consists of two counter-rotating pairs of mixing rods that stir a medium in a cylindrical tank.

Convection: Flow around a hot pole was simulated for 30 slightly perturbed initial velocity conditions at the bottom of the domain. The resulting 2D flow field ensemble is of size 128×256. Material at rest is heated around the pole, begins to rise, and forms a plume.

Cavity Flow: Laminar, in-compressible flow in a two-dimensional square domain where one border is moving with $1\frac{m}{s}$ was simulated. The Reynolds number of the simulated liquid is increased by one between each member generation, resulting in 1990 ensemble members with Reynolds numbers between 10 and 2000. The resolution is 1000×1000.

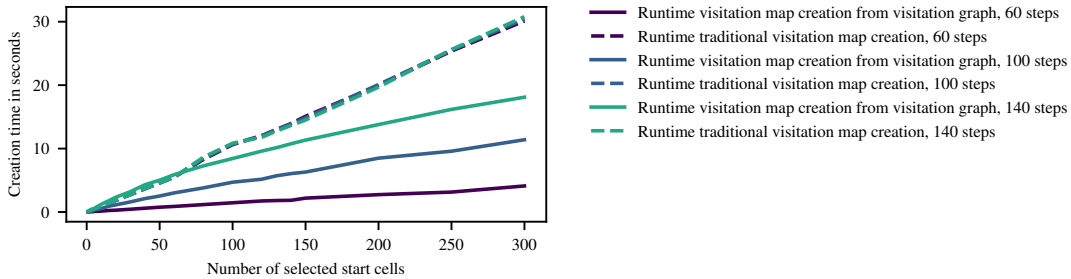

Figure 10 The creation time of visitation maps starting in multiple start cells. Visitation maps are faster generated from visitation graphs. The difference becomes larger with increasing number of start cells. In pre-processing, streamlines of 60 steps were calculated. Thus to calculate visitation maps with 100 and 140 steps, two resp. three calculation steps are needed. Intervals requiring the same number of steps result in similar generation times.

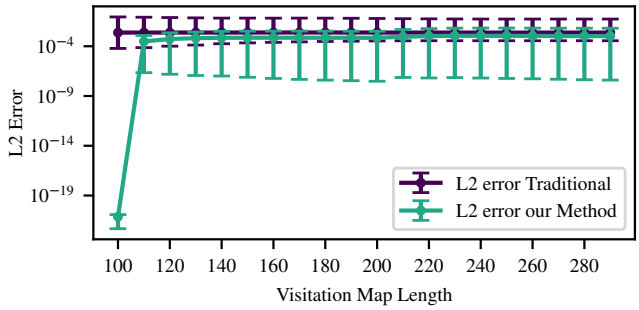

Figure 11 Visitation graphs are an optimal way to reduce data resolution and generate visitation maps afterwards: $T' = 100$, thus for a visitation map length of 100, our method equals the ground truth. Also for higher $T > T'$, our method outperforms the traditional approach. The increase in the error of our method at a visitation map length of 220 is to be expected every time T is increase d by T'. The error of the traditional method is higher and varies more.

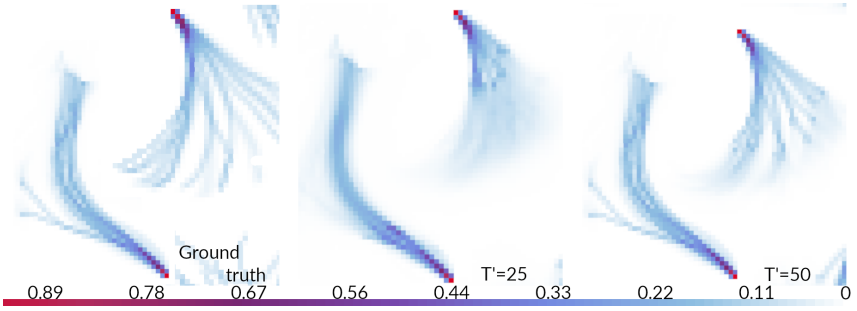

Figure 12 Results for $T' = 25$ (center) and $T' = 50$ (right) are compared with the ground truth (left). The approximation start after 25 and 50 steps respectively is clearly visible.

In all ensembles a fixed timepoint was investigated using visitation maps based on a visitation graph.

Industrial Stirring simulation. This example illustrates the approximation of visitation maps using visitation graphs based on shorter streamlines and different choices for T'.
Two pre-processing integration times were used. Serial runtime for $T' = 50$ with about 40 minutes per member is significantly higher than runtime for $T' = 25$ (about 20 minutes). The number of realizations was estimated for the latter execution. Based on a bunch size of 10, cells with very small vectors, especially the ones where no data exists, were start point for much less streamlines than in the non-automated calculation with 100 realizations for every cell. An exploratory visualization of the stirring simulation is given in Figure 1. Combining multiple start points with glyph visualization or streamline visualization of multiple members gives an overall impression. Additional initial points can be selected interactively. Like this, interesting regions, for example the flow behavior around the mixing rods can be examined easily by starting visitation maps close to them. With a pre-processing streamline length of 50, visitation maps of length 100 were generated. In Figure 12, visitation maps for $T' = 25$ and $T' = 50$ are compared. The error induced after the first approximation step is clear to see. Still, the tradeoff between pre-processing time and accuracy is profitable for exploration. The early approximated visitation maps are as useful as the ones based on $T' = 50$. In addition, the storage requirement of the visitation graph drops from 380MB to 118MB. Compared to the original ensemble size is 4.5MB, this is an increase. Yet, this example does

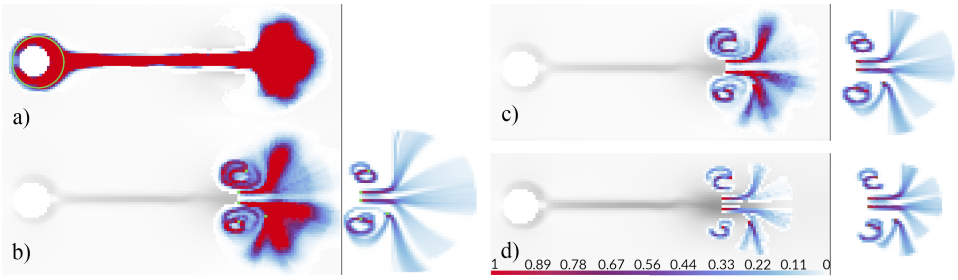

Figure 13 Convection simulation: Seeding multiple start cells around the pole (green circle) gives a first impression (a). Areas of interest are then explored using multiple single start cells (b-d). Visitation map length from b) to d) is 300,200,100. On the right, traditional visitation maps generated from 10.000 streamlines are given. For $T = T'$ differences result from the different number of streamlines. For $T > T$, the required approximation steps blur the result.

not aim on reduction of the data size. With 20 members of size 168×168, the number of ensemble members is not large compared to the number of cells. The average numbers of edges are 163.4 and 72.0 for $T' = 25$, 50 respectively, confirming that in real world examples the upper bound of $m \times n = 28,224$ is vastly overestimating the number of outgoing edges. The average number of stored 4-tuples per edge is 10.3 and 7.3 respectively which is again much smaller than the upper bound of T' in both cases.

Convection simulation. In this example we illustrate interactive exploration possibilities using visitation maps that are based on visitation graphs. A visitation graph with $T' = 100$ was generated for a single timestep of the convection simulation (Figure 13). Having detected the area of turbulent behavior using seeds around the pole, it can be explored by interactively adding multiple start points. Figures 13 a) and b) to d) show this process using different visitation map lengths; for orientation the average velocity over all ensemble members is given in the background. Processing one member took 145 minutes. Again, the upper bound of $m \times n = 32,768$ outgoing edges per cell is much higher than the real number which is 18.8 on average. 14.8 timesteps are stored per edge on average which is again much less than the upper bound $T' = 100$. This results in raw data of 72.8MB compared to original data size of 5.9MB. While our approach is not able to reduce the data in this case with a relatively low number of ensemble members, the speedup in visitation map creation is still given.

Cavity Flow. This example illustrates the data reduction abilities of visitation graphs. On the cavity flow dataset with a high number of members, a data-reducing visitation graph of resolution 100×100 was created with a timestep-storing frequency of 10 and T'=20. While creating visitation maps based on the original resolution might be impossible, the resulting visitation graph is able to retain information from the original resolution and still reduce the data. With 8 cores and a speedup factor close to 8, the calculation time of the visitation graph is about 140sec per member for $T' = 20$ and 100 streamlines per cell. Instead of the 30.2GB large ensemble, the visitation graph with 43.7MB can be stored and used to create visitation maps. On average, 131.67 outgoing edges were stored per cell and on each edge on average 1.15 timesteps (this number is reduced due to the chosen storing frequency). In Figure 14, an example for incremental exploration of the field with degree visualization background is given. Visitation maps generated from visitation graphs and in the traditional way are compared and a visitation map for the fine resolution is given.

6 Discussion and Conclusion

Vector field ensembles can be explored in an intuitive way using visitation maps. Traditional ad hoc calculation of visitation maps requires every ensemble member to be stored and sampled. Thus it can not be applied to ensembles that have a large number of members. In addition, every change of the seed point (or initial distribution) requires a full re-computation of the visitation map. And an initial distribution that is not strongly localized, results in even more time expensive computation. Visitation graphs provide solutions to these problems by isolating the sampling in in situ pre-processing while allowing the computation of visitation maps in an interactive fashion also for large ensembles and delocalized initial distributions. Costly calculation time is shifted to in situ pre-processing.

While the speed up of visitation map generation is present already for small ensembles, our method provides substantial advantages for ensembles with many members. For these ensembles, storage savings can be accomplished and visitation map generation speed up is quite significant. Multiple additional storage saving techniques are available for visitation graphs, allowing in-situ data reduction that is superior to other approaches with respect to visitation map creation.

The visitation graph stores information about all events of all generated streamline samples. Depending on spatial resolution, storage requirements can still be significant. A possible aggregation of the data in addition to a set storing frequency is part of our future work. Furthermore, parallelization (potentially on GPUs) can provide dramatically better runtimes for visitation graph generation and will be further examined in future work. Further potential for runtime reduction lies in re-using of generated streamlines during visitation graph generation.

Representing vector field ensembles as their corresponding visitation graphs offers a plethora of opportunities. Graph algorithms and clustering could be applied (e.g. in similarity to [38]), to exploit all strengths of this representation. Considering multiple time-steps of time dependent fields, results could be compared by comparing the visitation graphs, and missing time-steps could be interpolated on the graph level. Further investigations in these very promising areas are planned. Concerning visualization, different techniques could be applied to present the calculated visitation map such as isocontours or using transparency.

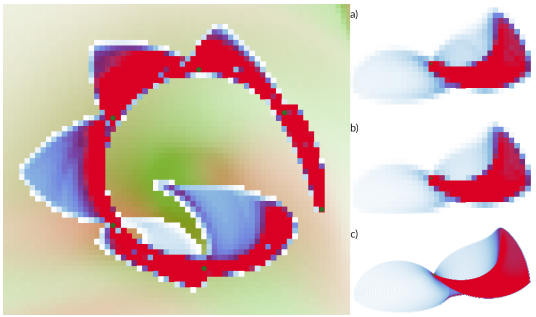

Figure 14 The cavity flow simulation. Left: Interactive exploration of the huge ensemble is possible. In the background, in- and out-degree of the cells is plotted in red and green respectively, indicating interesting areas. Seeding multiple start cells, the swirl in the data can be easily discovered and isolated. Right: Visitation maps from the same point, created with different approaches: a) our approach (0.06 sec) b) traditional based on the original ensemble (ground truth) (129,21 sec), c) traditional with original resolution (19,322.96 sec).

Finally, generalization of visitation graphs to 3D random fields will be part of our future work. While the basic technique of visitation graphs can be transferred quite naturally to 3D, the increasing number of grid cells will be the main challenge. The data reduction techniques presented in this paper seem to be promising to deal with this challenge. In addition, a link between visitation graphs and graphs used for 3D space partitioning is an exciting research topic we are going to examine. Furthermore the generalization of visitation maps to 3D requires an evaluation of existing 3D scalar field visualization techniques for this special purpose.

References

1 A. Agranovsky, D. Camp, C. Garth, E. W. Bethel, K. I. Joy, and H. Childs. Improved post hoc flow analysis via lagrangian representations. In *2014 IEEE 4th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 67–75, November 2014. doi:10.1109/LDAV.2014.7013206.

2 W. Berger, H. Piringer, P. Filzmoser, and E. Gröller. Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction. *Computer Graphics Forum*, 30(3):911–920, 2011. doi:10.1111/j.1467-8659.2011.01940.x.

3 Ryan A. Boller, Scott A. Braun, Jadrian Miles, and David H. Laidlaw. Application of uncertainty visualization methods to meteorological trajectories. *Earth Science Informatics*, 3(1):119–126, June 2010. doi:10.1007/s12145-010-0052-5.

4 Kai Bürger, Roland Fraedrich, Dorit Merhof, and Rüdiger Westermann. Instant visitation maps for interactive visualization of uncertain particle trajectories. In *IS&T/SPIE Electronic Imaging*, pages 82940P–82940P. International Society for Optics and Photonics, 2012.

5 Hank Childs. Data exploration at the exascale. *Supercomputing frontiers and innovations*, 2(3):5–13, 2015.

6 Jonathan Cox, Donald House, and Michael Lindell. Visualizing uncertainty in predicted hurricane tracks. *International Journal for Uncertainty Quantification*, 3(2):143–156, 2013.

7 John Dill, Rae Earnshaw, David Kasik, John Vince, and Pak Chung Wong. *Expanding the frontiers of visual analytics and visualization*. Springer, 2012.

8 Soumya Dutta, Chun-Ming Chen, Gregory Heinlein, Han-Wei Shen, and Jen-Ping Chen. In situ distribution guided analysis and visualization of transonic jet engine simulations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):811–820, 2017.

9 Nathan Fabian, Kenneth Moreland, David Thompson, Andrew C Bauer, Pat Marion, Berk Gevecik, Michel Rasquin, and Kenneth E Jansen. The paraview coprocessing library: A scalable, general purpose in situ visualization library. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pages 89–96. IEEE, 2011.

10 D. Feng, L. Kwock, Y. Lee, and R. Taylor. Matching visual saliency to confidence in plots of uncertain data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):980–989, November 2010. doi:10.1109/TVCG.2010.176.

11 David Feng, Lester Kwock, Yueh Lee, and Russel M. Taylor. Linked exploratory visualizations for uncertain mr spectroscopy data. *Proc.SPIE*, 7530:7530–7530–12, 2010. doi:10.1117/12.839818.

12 Florian Ferstl, Kai Bürger, and Rüdiger Westermann. Streamline variability plots for characterizing the uncertainty in vector field ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):767–776, 2016.

13 N. Fout and K. L. Ma. Fuzzy volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2335–2344, December 2012. doi:10.1109/TVCG.2012.227.

14 Yi Gu and Chaoli Wang. Transgraph: Hierarchical exploration of transition relationships in time-varying volumetric data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2015–2024, 2011.

15 Charles D Hansen, Min Chen, Christopher R Johnson, Arie E Kaufman, and Hans Hagen. *Scientific visualization: uncertainty, multifield, biomedical, and scalable visualization*. Springer, 2014.

16 C. Heine, H. Leitte, M. Hlawitschka, F. Iuricich, L. De Floriani, G. Scheuermann, H. Hagen, and C. Garth. A survey of topology-based methods in visualization. *Computer Graphics Forum*, 35(3):643–667, 2016. doi:10.1111/cgf.12933.

17 Marcel Hlawatsch, Philipp Leube, Wolfgang Nowak, and Daniel Weiskopf. Flow radar glyphs - static visualization of unsteady flow with uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1949–1958, 2011.

18 M. Hummel, H. Obermaier, C. Garth, and K. I. Joy. Comparative visual analysis of lagrangian transport in cfd ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2743–2752, December 2013. doi:10.1109/TVCG.2013.141.

19 Heike Jänicke and Gerik Scheuermann. Visual analysis of flow features using information theory. *IEEE Computer Graphics and Applications*, 30(1):40–49, 2010.

20 M. Jarema, J. Kehrer, and R. Westermann. Comparative visual analysis of transport variability in flow ensembles. *Journal of WSCG.*, 24(1):25–34, 2016. URL: http://hdl.handle.net/11025/21642.

21 D. K. Jones. Tractography gone wild: Probabilistic fibre tracking using the wild bootstrap with diffusion tensor mri. *IEEE Transactions on Medical Imaging*, 27(9):1268–1274, September 2008. doi:10.1109/TMI.2008.922191.

22 Derek K. Jones and Carlo Pierpaoli. Confidence mapping in diffusion tensor magnetic resonance imaging tractography using a bootstrap approach. *Magnetic Resonance in Medicine*, 53(5):1143–1149, 2005. doi:10.1002/mrm.20466.

23 Sriram Lakshminarasimhan, Neil Shah, Stephane Ethier, Scott Klasky, Rob Latham, Rob Ross, and Nagiza F. Samatova. Compressing the incompressible with isabela: In-situ reduction of spatio-temporal data. In Emmanuel Jeannot, Raymond Namyst, and Jean Roman, editors, *Euro-Par 2011 Parallel Processing*, pages 366–379, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

24 S. Li, N. Marsaglia, C. Garth, J. Woodring, J. Clyne, and H. Childs. Data reduction techniques for simulation, visualization and data analysis. *Computer Graphics Forum*, 37(6):422–447, 2018. doi:10.1111/cgf.13336.

25 Jay F Lofstead, Scott Klasky, Karsten Schwan, Norbert Podhorszki, and Chen Jin. Flexible io and integration for scientific codes through the adaptable io system (adios). In *Proceedings of the 6th international workshop on Challenges of large applications in distributed environments*, pages 15–24. ACM, 2008.

26 A. L. Love, A. Pang, and D. L. Kao. Visualizing spatial multivalue data. *IEEE Computer Graphics and Applications*, 25(3):69–79, May 2005. doi:10.1109/MCG.2005.71.

27 Jun Ma, Chaoli Wang, and Ching-Kuang Shene. Flowgraph: A compound hierarchical graph for flow field exploration. In *2013 IEEE Pacific Visualization Symposium (PacificVis)*, pages 233–240. IEEE, 2013.

28 Mahsa Mirzargar, Ross T Whitaker, and Robert M Kirby. Curve boxplot: Generalization of boxplot for ensembles of curves. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2654–2663, 2014.

29 Boonthanome Nouanesengsy, Teng-Yok Lee, and Han-Wei Shen. Load-balanced parallel streamline generation on large scale vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1785–1794, 2011.

30 M. Otto, T. Germer, and H. Theisel. Uncertain topology of 3d vector fields. In *2011 IEEE Pacific Visualization Symposium*, pages 67–74, March 2011. doi:10.1109/PACIFICVIS.2011.5742374.

31 Mathias Otto, Tobias Germer, Hans-Christian Hege, and Holger Theisel. Uncertain 2d vector field topology. *Computer Graphics Forum*, 29(2):347–356, 2010. doi:10.1111/j.1467-8659.2009.01604.x.

32 Tobias Pfaffelmoser, Matthias Reitinger, and Rüdiger Westermann. Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields. *Computer Graphics Forum*, 30(3):951–960, 2011. doi:10.1111/j.1467-8659.2011.01944.x.

33 K. Pothkow and H. C. Hege. Positional uncertainty of isocontours: Condition analysis and probabilistic measures. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1393–1406, October 2011. doi:10.1109/TVCG.2010.247.

34 Dave Pugmire, Hank Childs, Christoph Garth, Sean Ahern, and Gunther H. Weber. Scalable computation of streamlines on very large datasets. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC '09, pages 16:1–16:12, New York, NY, USA, 2009. ACM. doi:10.1145/1654059.1654076.

35 Jibonananda Sanyal, Song Zhang, Jamie Dyer, Andrew Mercer, Philip Amburn, and Robert Moorhead. Noodles: A tool for visualization of numerical weather model ensemble uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1421–1430, 2010.

36 Dominic Schneider, Jan Fuhrmann, Wieland Reich, and Gerik Scheuermann. *A Variance Based FTLE-Like Method for Unsteady Uncertain Vector Fields*, pages 255–268. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi:10.1007/978-3-642-23175-9_17.

37 L. Sevilla-Lara and E. Learned-Miller. Distribution fields for tracking. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1910–1917, June 2012. doi:10.1109/CVPR.2012.6247891.

38 Andrzej Szymczak. Morse connection graphs for piecewise constant vector fields on surfaces. *Computer Aided Geometric Design*, 30(6):529–541, 2013.

39 Chaoli Wang and Jun Tao. Graphs in scientific visualization: A survey. *Computer Graphics Forum*, 36(1):263–287, 2017. doi:10.1111/cgf.12800.

40 Ross T Whitaker, Mahsa Mirzargar, and Robert M Kirby. Contour boxplots: A method for characterizing uncertainty in feature sets from simulation ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2713–2722, 2013.

41 Brad Whitlock, Jean M. Favre, and Jeremy S. Meredith. Parallel in situ coupling of simulation with a fully featured visualization system. In *Proceedings of the 11th Eurographics Conference on Parallel Graphics and Visualization*, EGPGV '11, pages 101–109, Aire-la-Ville, Switzerland, Switzerland, 2011. Eurographics Association. doi:10.2312/EGPGV/EGPGV11/101-109.

42 Lijie Xu and Han-Wei Shen. Flow web: A graph based user interface for 3d flow field exploration. In *IS&T/SPIE Electronic Imaging*, pages 75300F–75300F. International Society for Optics and Photonics, 2010.

43 Björn Zehner, Norihiro Watanabe, and Olaf Kolditz. Visualization of gridded scalar data with uncertainty in geosciences. *Computers & Geosciences*, 36(10):1268–1275, 2010. doi:10.1016/j.cageo.2010.02.010.