Interactive view-driven evenly spaced streamline placement

Zhanping Liu and Robert J. Moorhead II*

High Performance Computing Collaboratory, Mississippi State University, MS 39762-9627, USA

ABSTRACT

This paper presents an *Interactive View-Driven Evenly Spaced Streamline* placement algorithm (IVDESS) for 3D explorative visualization of large complex planar or curved surface flows. IVDESS rapidly performs accurate streamline integration in 3D physical space, i.e., the flow field, while achieving high quality streamline density control in 2D view space, i.e., the output image. The correspondence between the two spaces is established by using a projection-unprojection pair constituted through geometric surface rendering. An inter-frame physical-space seeding strategy based on streamline reuse+lengthening is adopted, on top of intra-frame view-space seeding, to not only enable coherent flow navigation but also speedup placement generation. IVDESS employs a view-sensitive streamline representation that is well suited for streamline reuse, lengthening, and rendering. In addition, it avoids temporal incoherence caused by streamline splitting and jaggy lines caused by unprojection errors. Our algorithm can run at interactive frame rates (9FPS for placement generation) to allow for 3D exploration of surface flows with smooth evolution of high-density (1%) evenly spaced streamlines in a large window (990 × 700 pixels) on an ordinary PC without either pre-processing or GPU support.

Keywords: Flow visualization, evenly spaced streamlines, streamline placement, seeding strategy, view-dependent, image space, physical space, temporal coherence, perspective projection, off-screen rendering

1. INTRODUCTION

Flow visualization is widely used in oceanographic-atmospheric modeling, computational fluid dynamics simulation, and electro-magnetic field analysis. There have been a variety of flow visualization methods, among which texture-based ones[1-4] are gaining significant attention for the dense depiction of a flow. Although texture-based techniques are powerful in visualizing 2D flows, they may be ineffective in handling curved surface and volume flows as the dense representation tends to be a disadvantage in 3D due to view occlusion, depth ambiguity, direction vagueness, and aliasing artifacts. Complicated transfer functions are usually needed to reconstruct or extract flow lines in a volume by rendering a synthesized 3D *scalar* texture of the flow. As a geometry-based method, streamlines remain one of the most important approaches to flow visualization for the straightforward direction cueing and the low computational expense. A layout of evenly spaced streamlines[5-8] may provide an aesthetic and informative pattern, without either incompleteness or cluttering, to facilitate mental reconstruction of the flow. The sparseness makes evenly spaced streamlines, particularly coupled with curve illumination[9], a promising solution for visualizing planar flows, curved surface flows, and volume flows. In addition, evenly spaced streamlines can clearly show flow directions readily obtained through streamline integration without any "reconstruction".

The application of evenly spaced streamlines to 3D exploration of volume flows, curved surface flows, and even planar flows in a perspective-view setting poses at least three challenges. The first problem is how to create a placement of streamlines such that they are indeed evenly spaced in 2D *view space* (the output image) because streamlines evenly spaced in 3D *physical space* (the flow field) may not *visually* retain the uniformity when projected to view space through a foreshortening transform. Thus view-dependency issues need to be considered to address incompleteness, cluttering, occlusion, and non-uniformity in the resulting image. The second challenge is how to maintain a smooth transition between two consecutive frames of evenly spaced streamlines for coherent flow exploration. This concern is critical for dynamic level-of-detail flow investigation as features may be missing in a static low-density placement or indiscernible in a static high-density layout due to the foreshortening effect. The third issue is how to find a fast universal mechanism,

* Further author information: (Send correspondence to Zhanping Liu)
Zhanping Liu: E-mail: zhanping@hpc.msstate.edu, Telephone: (662) 325-2526
Robert J. Moorhead II: E-mail: rjm@hpc.msstate.edu, Telephone: (662) 325-2850
Color figures, images, and accompanying movies are available at http://www.zhanpingliu.org/Research/FlowVis/IVDESS

Visualization and Data Analysis 2008, edited by Katy Börner, Matti T. Gröhn,
Jinah Park, Jonathan C. Roberts, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 6809
6809A, © 2008 SPIE-IS&T · 0277-786X/08/$18

regardless of the data size or the grid type, to enable interactive placement of visually evenly-spaced streamlines for high-performance visualization of large flows defined on complex grids.

To tackle the aforementioned problems, we present an interactive view-driven evenly spaced streamline placement algorithm for coherent explorative flow visualization. A projection-unprojection pair is exploited through geometric surface rendering to construct the transformation between physical space and view space. A fourth-order Runge-Kutta integrator with adaptive step size and error control is used in combination with line-field clipping to integrate streamlines in physical space. Segment sampling, line-view clipping, inter-line distance checking, and intra-line loop detection are performed in view space to control streamline density. On top of intra-frame view-space seeding, inter-frame physical-space seeding based on streamline reuse+lengthening is adopted to maintain smooth inter-frame transition and accelerate placement generation. A view-sensitive streamline representation in support of streamline reuse+lengthening is employed to avoid both the temporal incoherence problem caused by streamline splitting and jaggy lines caused by unprojection errors. Our algorithm is applicable to interactive 3D exploration of not only a planar flow but also the flow on either a model surface or a streamsurface/isosurface.

The remainder of this paper is organized as follows. Section 2 gives an introduction to related streamline placement algorithms. Section 3 describes our view-driven evenly spaced streamline placement algorithm. Results are provided in section 4 to demonstrate the view-dependent streamline placement quality, the temporal coherence performance, and the placement generation speed of our algorithm. We conclude the paper with a brief summary and outlook on future work.

2. RELATED WORK

Existing algorithms for creating evenly spaced streamlines may be roughly categorized into image-guided and sample-based. Image-guided algorithms[5, 10] take a streamline placement as a binary-valued image, of which low-pass filtered versions are compared against a reference uniform gray-scale image to direct the iterative refinement of an initial layout toward an optimal result. Turk and Banks[5] proposed the first algorithm of this kind that works by inserting, merging, re-positioning, lengthening, shortening, or deleting streamlines in a trial-and-error manner at a large computational cost.

Sample-based algorithms[6-8, 11-15] employ inter-sample distance control to approximate inter-line distance control as streamlines are advected to leave samples in a placement. Given a threshold separating distance, which governs the density of a placement of evenly spaced streamlines, it is mandatory that the interval between any two successive samples of a streamline be less than the threshold to make inter-sample distance control acceptable. A brute-force solution is to perform distance checking on the newest sample of the current streamline against each sample of the existing streamlines in the placement to determine whether this streamline is further advected upon sample acceptance or immediately terminated upon any sample refusal. Sample-based algorithms mainly differ from one another in the seeding strategy that drives the placement process until a dynamic queue of candidate seeds is empty and in the data structure that is used to store streamline samples and perform inter-sample distance control.

Jobard and Lefer presented a cell-based inter-line oriented inter-sample distance controller[6] to limit distance checking on the newest sample of the current streamline to only the samples that have been accepted by the sample's nine local cells within a virtual Cartesian grid. This grid is "superimposed" on the flow field, with the cell size equal to the threshold separating distance. Liu et al. proposed the use of an additional cell-based but intra-line oriented inter-sample distance controller in the ADVESS algorithm[8] to detect streamline loops by performing similar inter-sample distance checking via a temporarily maintained streamline-based sparsely populated Cartesian grid[16], in contrast with the initial placement-based densely populated Cartesian grid accessed by the inter-line distance controller.

Jobard and Lefer adopted a non-priority neighborhood seeding strategy[6], by which each sample of an accepted streamline introduces into a FIFO queue two neighboring candidate seeds in the direction orthogonal to the flow. Verma et al. proposed a flow-guided priority seeding strategy[13] to emphasize flow topology in a placement. Topological templates are employed to place seeds schematically around critical points before additional seeds are scattered using Poisson-disk distribution. Liu et al. presented a double-queue priority seeding strategy in the ADVESS algorithm[8] that enhances the initial neighborhood seeding scheme[6] with priority assigned to both flow topology and long streamlines. A primary queue serves as the major seed provider to store not only topologically distributed seeds but also streamline seed-introduced candidates that are sorted by the streamline length. A secondary FIFO queue stores candidates introduced by non-seed streamline samples and it is used only when the primary queue is temporarily empty. Mebarki et al. proposed a farthest point seeding strategy[7] based on the use of a queue of unvisited Delaunay triangles, of which the largest one provides the circumcircle center as the new seed. For each advected streamline, the unvisited incident

triangles with the circumcircle diameters greater than the threshold distance are added to the queue and sorted by the diameter.

There are several streamline placement algorithms related to surface/volume flows or flow exploration. Jobard and Lefer presented a method for interactive multi-resolution 2D exploration of a planar flow through the use of a multi-density placement representation constructed by incrementally generating and hierarchically storing a highest-density placement of evenly spaced streamlines[11]. However, a pre-processing stage is required to refine a lowest-density layout toward a highest-density placement to build the hierarchical representation. In addition, this static mechanism supports only a fixed number of levels of detail. Mattausch et al.[14] adopted a similar pre-processing scheme to offset the large computational expense of producing a multi-density representation of evenly spaced illuminated streamlines for explorative volume flow visualization. The negligence of view-dependency issues incurs occlusion and cluttering and therefore streamlines are not evenly spaced in view space as they are in physical space. These problems occur with Ye et al.'s 3D flow-guided approach[17] too that seeks to create a placement of intelligently distributed rather than evenly spaced streamlines. Mao et al.[10] extended the original image-guided algorithm[5] for evenly spaced streamline placement on curved surfaces, but view-dependency factors were not addressed. The basic idea is to generate a non-uniform layout in 2D computational space such that a uniform placement in 3D physical space is obtained when the former is mapped onto a curved surface. Thus streamlines are evenly spaced in physical space, but not in view space.

Li and Shen proposed a framework for image-based streamline placement and rendering[15]. Their contributions lie in how to place streamlines for surface and volume flows such that occlusion and cluttering may be alleviated to some degree to reveal important flow features when streamlines are projected to view space. In fact, streamlines are not evenly spaced in the output image where occlusion and cluttering still noticeably remain. A streamline reuse approach to inter-frame transition is only briefly mentioned in the framework, whereas some critical issues such as temporal incoherence caused by streamline splitting and jaggy lines caused by unprojection errors are simply ignored. Our work is related to but significantly different from Li and Shen's work. The essential difference is that our view-driven algorithm is capable of placing streamlines that are indeed evenly spaced in the output image. Furthermore, a tangible and effective seeding strategy, based on greedy streamline reuse+lengthening and view-sensitive streamline representation, is adopted in our algorithm to provide a complete solution for coherent flow exploration with evenly spaced streamlines.

3. INTERACTIVE VIEW-DRIVEN EVENLY SPACED STREAMLINE PLACEMENT

In this section, we present an *Interactive View-Driven Evenly Spaced Streamline* placement algorithm (IVDESS), which is built on our previous ADVESS algorithm[8] — a 2D streamline placement engine with robust loop detection[16]. First we describe the components of the IVDESS pipeline that are involved in generating a single placement or frame. Then we propose a temporally coherent seeding strategy to maintain smooth inter-frame transition for explorative purposes. Accordingly, we present a view-sensitive streamline representation for streamline reuse, lengthening, and rendering without introducing temporal incoherence or propagating unprojection errors.

3.1 IVDESS Pipeline

Fig. 1 shows the pipeline of IVDESS. Streamline integration, streamline reuse+lengthening, line-field clipping, and streamline storage (representation) are performed in physical space, as opposed to segment sampling, line-view clipping, intra-line distance control (loop detection), inter-line distance control (density control), and neighborhood candidate seed selection in view space. IVDESS adopts a hybrid-space multi-level seeding mechanism in which physical-space seeding takes priority over view-space seeding. The latter is used to create a separate frame of view-dependent evenly spaced streamlines while the former is used to establish inter-frame coherence for flow exploration. We defer the discussion of physical-space seeding and streamline representation until sections 3.2 and 3.3, respectively.

3.1.1 Off-Screen Rendering and Space Transformation

The transformation between physical space and view space is achieved in IVDESS through off-screen rendering to decouple streamline placement from image output. This choice not only avoids multi-pass rendering but also brings flexibility to geometric surface (a quad for planar flows) rendering. This is the case with 3D exploration of a planar flow or a streamsurface, of which the geometry is not necessarily shown in the output image. What is really required in the view-driven streamline placement process is the view-dependent depth of the surface such that given a 2D sample in view space and within the surface's projection, the corresponding 3D point in physical space can be obtained via *unprojection*, i.e., the backward transform. As the forward transform, *projection* can be used to map a 3D point in

physical space to the corresponding 2D sample in view space. In fact, this projection-unprojection pair is readily available as two functions in the OpenGL GLU library. Since *pixel buffer* is widely supported by OpenGL window system interfaces like WGL, it is adopted in IVDESS to implement off-screen rendering. An alternative solution is to use Frame Buffer Objects (FBOs) on some latest GPUs, though currently IVDESS is designed to be GPU-independent.

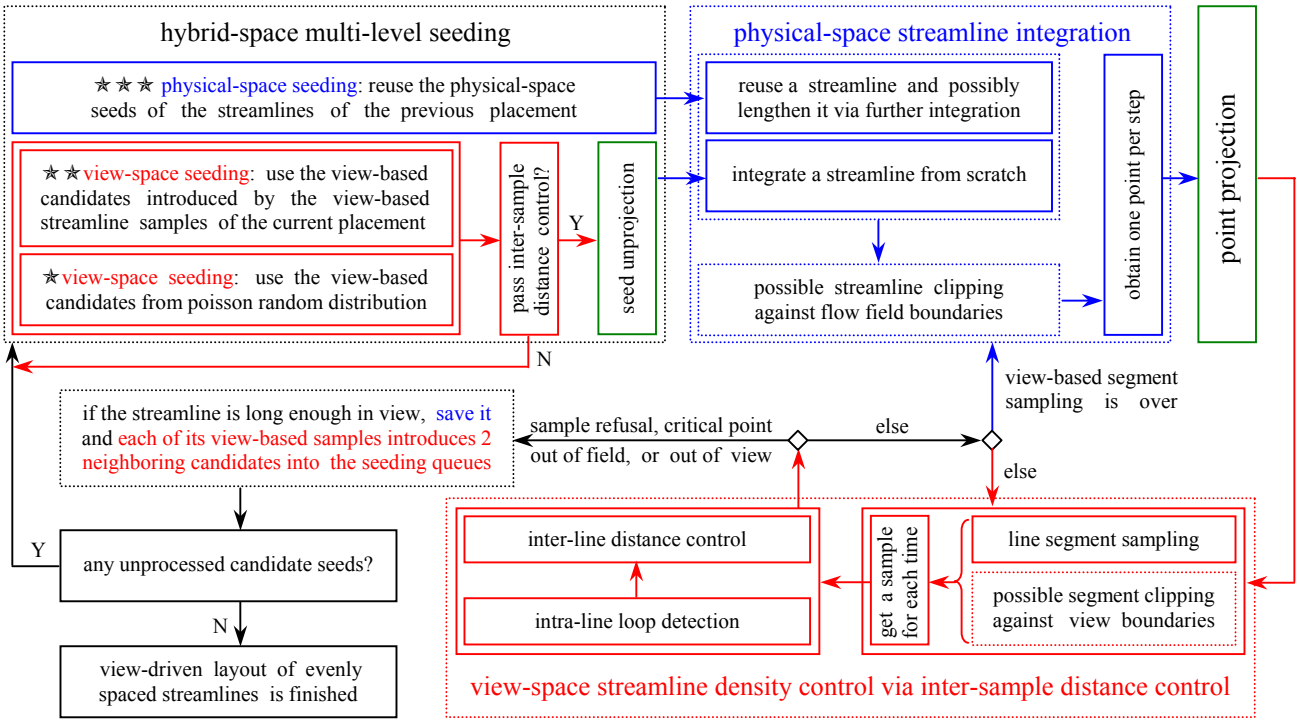


Figure 1. The IVDESS pipeline built on our 2D ADVESS engine[8] for view-driven evenly spaced streamline placement.

Specifically, a secondary rendering context other than the intrinsic primary one is bound to pixel buffer for off-screen rendering. The two rendering contexts are configured and, upon user interaction with the scene, dynamically updated in exactly the same way. They serve as the active context alternately to fulfill their respective rendering tasks, the secondary for depth generation and the primary for image output. Once off-screen rendering is completed for a frame, the depth of each pixel is available for access. Given a view-space sample, the depth value may be:

(1) depth ≤ 0.0: the sample is not within the view
(2) depth ≥ 1.0: the sample is within the view but not within the projection of the surface, i.e., *noninformative*
(3) 0.0 < depth < 1.0: the sample is within both the view and the projection of the surface, i.e., *informative*

of which only the third case is considered for useful sample unprojection. A view-space sample generated in the streamline placement process is seldom at an integer pixel location and therefore the depth needs to be interpolated within a quad of four neighbouring pixels so as to unproject the sample. Thus the field boundaries and the view boundaries must be handled with care to guarantee valid unprojection.

Since projection is a non-linear transform, unprojection as the inverse transform is more computationally expensive than projection. In addition, unprojection incurs numerical errors, which poses a problem for inter-frame streamline reuse+lengthening (section 3.3). Unprojection is conducted for candidates popped from seeding queues and samples obtained via line-view clipping. Projection is performed on points directly generated by streamline integration and those produced indirectly by line-field clipping such that a streamline can be sampled segment by segment in view space.

3.1.2 Streamline Integration and Segment Sampling

Any fixed step size integrator, even a fourth-order Runge Kutta (RK4) scheme, is a tradeoff between computational speed and numerical accuracy. In addition, it involves the constraint that a given threshold distance imposes on the

selection of a valid step size. The drawbacks to employing a fixed step size integrator for streamline placement are exacerbated in view-dependent scenarios. Numerical errors dormant in a 2D placement may emerge in 3D exploration as noticeable jaggy line artifacts. This is particularly the case with a CAVE-like environment in which the user may be very close to streamlines as the flow is investigated. In fact, such an integrator even with a very small step size still cannot obviate the need for additional segment sampling because a segment may be greatly "lengthened" in view space.

IVDESS adopts an RK4 integrator with adaptive step size and error control for fast but accurate streamline integration. The error of each RK4 integration is estimated using embedded RK formulae. Given an error tolerance range $[\varepsilon_{min}, \varepsilon_{max}]$, the step size is doubled when $\varepsilon < \varepsilon_{min}$, whereas it keeps being halved for repeated RK4 integrations as long as $\varepsilon > \varepsilon_{max}$. For physical-space streamline placement[8], cubic Hermite polynomial interpolation may be used to exploit readily available normalized flow vectors to evenly sample a streamline rapidly. However, this method does not apply to IVDESS due to perspective view projection. Given the head and tail of a segment in view space, i.e., two *raw segment samples* — the projection of two physical-space *raw points* of the streamline, the segment is uniformly sampled in a brute-force manner by the threshold distance to generate *intermediate segment samples* for inter-sample distance control.

3.1.3 Distance Control and Line Clipping

As a streamline is integrated, each view-space segment sample is first sent to the intra-line distance controller[16] to detect any closed or open but tightly spiraling loop and, unless rejected, forwarded to the inter-line distance controller to maintain placement density. If accepted by both, the sample is immediately registered in the intra-line distance controller. However, samples are registered on a per-streamline basis in the inter-line distance controller only if the streamline passes the view-space length check upon the termination of the advection. The interval between two successive view-space samples of a streamline varies due to the involvement of raw segment samples in distance control, but the influence on placement uniformity is negligible if an appropriate ratio between two distance parameters is used[8].

It is necessary to perform line-field clipping during flow advection in physical space and line-view clipping during segment sampling in view space. A lack of line-field clipping may introduce cavities and discontinuities in the placement as subsequent streamlines approach from opposite directions. Line-view clipping cannot be substituted by rendering segments across the view boundaries because the negligence of both sample generation and registration in the inter-line distance controller allows other streamlines to encroach inward near the view boundaries to cause cluttering.

3.1.4 Temporally Incoherent Seeding Strategy

The flow-guided seeding strategy[13] employs several topological templates as seeding patterns to exploit spatial coherence that exists more or less in the direction orthogonal to the flow around critical points. Thus in each template, seeds are usually, or as much as possible for sources and sinks, distributed in such directions in an evenly spaced manner to facilitate streamline placement. This strategy is effective in creating 2D evenly spaced streamlines[8] and producing 3D *unevenly* spaced but intelligently placed streamlines[15, 17]. However, its advantage considerably decreases for view-dependent scenarios because that kind of spatial coherence is not only reduced in view space through projection but also further degraded through inaccurate unprojection (section 3.3). Furthermore, static global topological analysis is not appropriate for view-driven and hence local dynamic streamline placement, particularly when large flows are explored.

To generate a single frame of view-dependent evenly spaced streamlines, IVDESS adopts a view-space seeding mechanism that integrates Poisson-disk seed distribution, neighborhood candidate selection, and double-queue seed scheduling. Poisson-disk distribution is used at the beginning to push some random seeds to the tail of the secondary queue to init the placement process while guaranteeing view coverage. Candidates introduced by the seed of each accepted streamline are stored in the primary queue and sorted by the view-space streamline length, whereas others by non-seed samples are simply stored in the secondary FIFO queue. The primary queue takes priority over the secondary queue in providing candidate seeds to exploit spatial coherence, if any. As this intra-frame scheme does not consider the transition between two consecutive placements of flow exploration, it is a *temporally incoherent seeding strategy*.

An IVDESS placement provides a multi-resolution flow representation, as indicated by the correspondence between the physical-space layout and the view-space output (Fig. 2). In fact, a higher resolution is used to represent a nearer flow area (e.g., the saddle at the lower part) than used to represent a farther flow area (e.g., the focus at the upper part) in a visually evenly-spaced streamline placement, which coincides with the human visual system. The sketch depiction of the distance serves as the exploration context while the coarse view of a region of interest (e.g., the focus) helps the user find important features. Due to the view-driven property, the multi-resolution flow representation is dynamic in that a flow area can be investigated at an increasingly high resolution in a very natural fashion as the user approaches it.

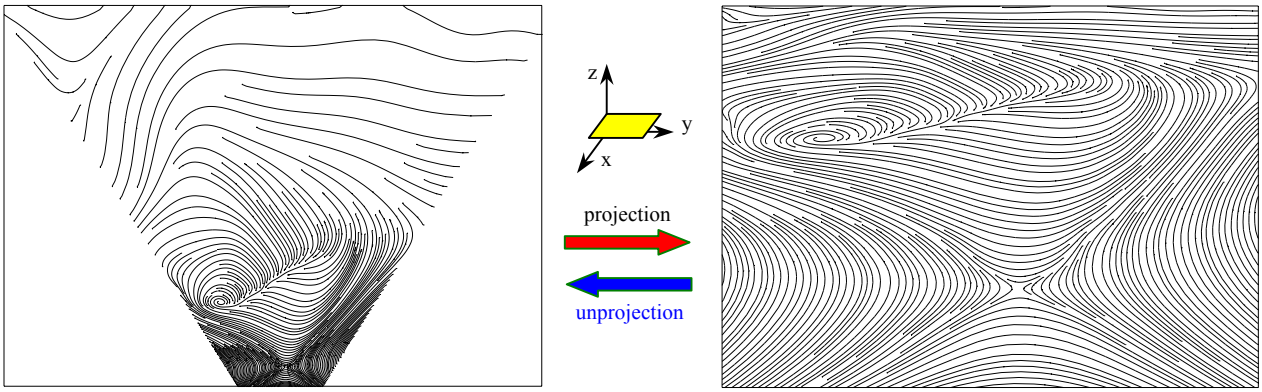


Figure 2. The correspondence between (a) the non-uniform streamline placement of a planar flow (within the foreshortening frustum) in 3D physical space and (b) the resulting visually uniform streamline placement in 2D view space, i.e., an IVDESS placement.

3.2 Temporally Coherent Seeding Strategy

Temporally incoherent intra-frame view-space seeding ignores explorative issues. As a result, visual distractions may occur as the user navigates through the scene. To achieve smooth inter-frame transition for immersive flow investigation, IVDESS employs physical-space seeding prior to view-space seeding to reuse, through segment reprojection and resampling, and lengthen the streamlines of the previous frame under normal density control in the current frame. The order in which streamlines are reused plays a crucial role in maintaining *temporal coherence*. Thus streamlines need to be sorted by some criteria such as the view-space length and optionally the line-view clipping status.

3.2.1 Efficient Greedy Non-Split Streamline Reuse+Lengthening

To establish temporal coherence, each streamline of the previous frame is processed beginning with the seed in both directions, which facilitates streamline lengthening. A streamline is potentially reused in either direction as long as *the first in-view segment sample* (Fig. 3), either an intermediate segment sample obtained via line-view clipping or a raw segment sample — the projection of the in-view seed, is accepted by inter-sample distance control. This *greedy streamline reuse+lengthening* scheme makes it possible for a streamline even with the seed out of the view to be reused. It is based on the observation that a streamline may be visually long in the previous frame despite the seed approaching the view boundaries outward or having been out of the view. The disappearance of such streamlines in the current frame may bring noticeable change to the view. For greedier streamline reuse, not only the accepted part of a streamline but also the rejected part and even the out-of-view part are saved if the streamline passes the view-space length check. In case the latter two invisible parts are accepted in subsequent frames, repetitive streamline integration can be avoided.

This greedy streamline reuse+lengthening method effectively prevents streamline splitting by saving the out-of-view part of a streamline to "invisibly" concatenate two in-view parts, i.e., child-streamlines (Fig. 4). Streamline splitting may cause discontinuities in a placement and affect inter-frame coherence when the out-of-view split part re-enters the view, as is the case with back-and-forth flow investigation. An abrupt discontinuity tends to emerge as the two child-streamlines approach each other on the split side. A similar discontinuity may occur too on the other side in case the two child-streamlines are further advected toward each other to form a common closed loop. Furthermore, the probability for either child-streamline to be rejected by inter-sample distance control may be higher than that for the initial streamline when the streamlines of the previous frame are sorted by the length. In fact, the disappearance of a streamline over the view boundaries greatly affects temporal coherence. Another problem with streamline splitting for back-and-forth flow exploration is the significant increase in the number of streamlines and hence in the cost of both storing and sorting streamlines. *Non-split* streamline reuse+lengthening completely avoids discontinuities caused by intra-streamline rejection between child-streamlines. Part of a streamline can safely leave and re-enter the view without introducing incoherence over the view boundaries. In addition, closed streamlines as important flow features are allowed to form.

To obtain the first in-view segment sample of a streamline in either direction, the physical-space seed is first accessed from the storage buffer and projected to get a raw segment sample in view space. If this sample is in the view, it serves as the first in-view segment sample (R_0 in a, b, and c of Fig. 3). Otherwise, subsequent raw points in the current direction are projected one by one to generate raw segment samples until the first raw point is found, if any, of which the

projection sample (R_0 in d, e, and f of Fig. 3) is in the view. This sample is then used in combination with the previous out-of-view raw segment sample to compute the first in-view segment sample by means of line-view clipping. In either case, only when the first in-view segment sample passes inter-sample distance control, is the streamline further processed in the current direction. Then subsequent raw points are extracted for view projection and segment sampling under normal distance control. Any sample refusal terminates projection and sampling, whereas a successful process through the streamline tail leads to further flow advection, i.e., streamline lengthening, in physical space. In either case, the segment-wise view-space length of the accepted part of the streamline is accumulated. If the seed is in the view, the total length in both directions determines whether or not the streamline can be present in the current frame. Otherwise, the view-space length check needs to be performed on a per-direction basis. In this scenario, a successfully reused streamline with both sides saved may have only one side shown in the placement.

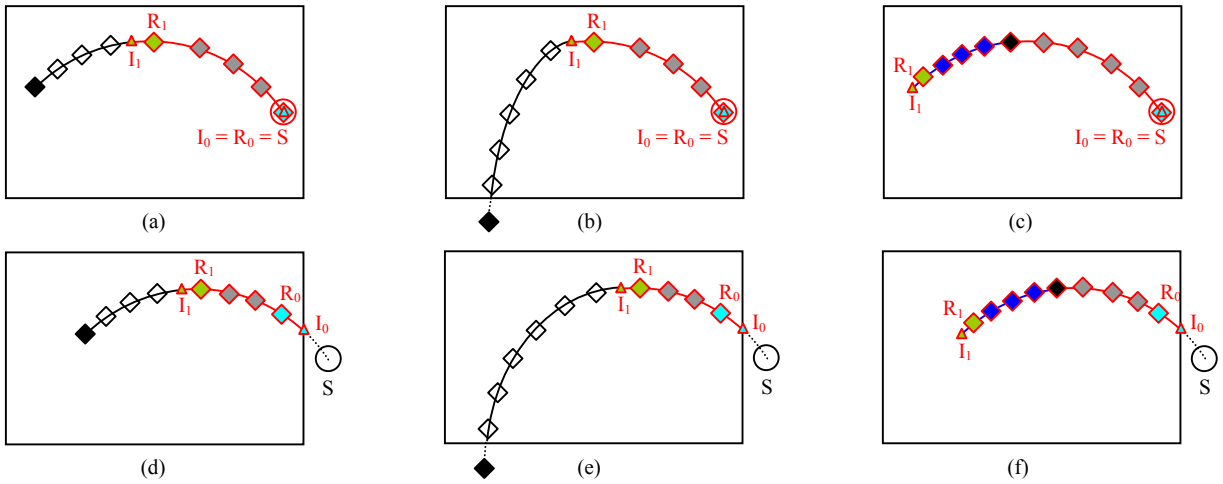


Figure 3. Greedy streamline reuse+lengthening in the one-direction case. Regardless of whether the seed S is in the view (in a, b, and c) or not (in d, e, and f), a streamline is considered for reuse as long as the first in-view segment sample (I_0), either an intermediate segment sample obtained via line-view clipping (in d, e, and f) or a raw segment sample (just the seed itself in a, b, and c), passes inter-sample distance control before subsequent physical-space raw points are projected to view space to produce raw segment samples (diamonds in grey and green in a, b, d, and e; diamonds in grey and black in c and f), in combination with segment sampling, under normal inter-sample distance control toward the end (diamonds in black) of the streamline. Any sample refusal terminates raw point projection and segment sampling, whereas a successful process through the end invokes further streamline integration, coupled with segment sampling between raw segment samples (diamonds in blue and green in c and f). Once the streamline survives the view-space length check, not only the accepted original part (red lines) and the lengthened part (blue lines) but also the rejected part (black lines) including the out-of-view part (dotted black lines) are saved. I_0, R_0, R_1, and I_1 can be used to represent the viewable part of an accepted (successfully reused) streamline. Note that some intermediate segment samples are not shown here for illustrative purposes.

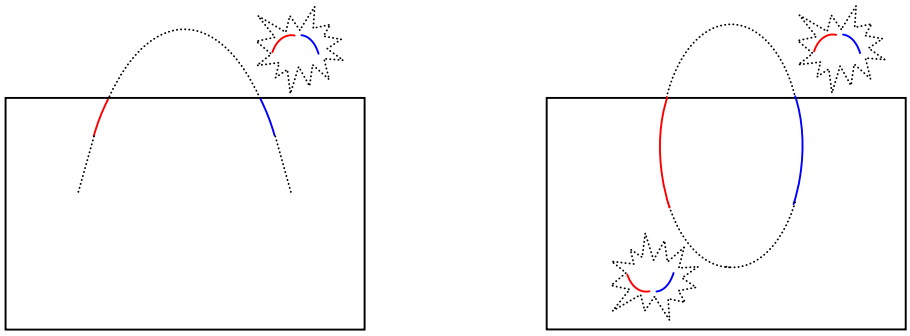


Figure 4. The problems with streamline splitting. Two separated but accepted in-view child-streamlines (solid) resulting from streamline splitting by the out-of-view part (dotted in the upper part) give rise to a discontinuity on the split side when they are integrated to approach each other to make the out-of-view part re-enter the view. A similar discontinuity emerges on the other side (dotted in the lower part of the right figure) when the two child-streamlines are integrated to form a common closed loop.

To reuse streamlines greedily without sacrificing computational efficiency, a streamline is not considered for reuse in either direction if the first in-view segment sample fails inter-sample distance checking. On the other hand, a streamline even with the seed out of the view is possibly reused as long as the first in-view segment sample passes inter-sample distance checking. This requirement is based on the observation that the point projection and in-view judgment process that is used to get the first in-view segment sample is far computationally cheaper than inter-sample distance checking.

3.2.2 Temporally Coherent Seeding Strategy

View-space seeding is a temporally incoherent seeding strategy (section 3.1.4), but the addition of physical-space seeding, i.e., streamline reuse+lengthening, on top of it leads to a *temporally coherent seeding strategy*. This hybrid–space multi-level seeding mechanism is well suited for explorative visualization of massive flows with smooth evolution of view-driven evenly spaced streamlines. Streamline reuse+lengthening is first employed for each frame to maintain temporal coherence before neighborhood candidate selection and Poisson-disk seed distribution are used to advect new streamlines to fill the view. In addition to the two seed queues adopted for double-queue scheduling, two dynamically updated streamline queues are used in IVDESS, i.e., *preQue* for storing the streamlines of the previous frame and *curQue* for storing both the streamlines successfully reused and those newly advected in the current frame. These two streamline queues work in a ping-pong fashion to accomplish streamline reuse that exploits spatial coherence of flow lines to maintain temporal coherence of user exploration. In fact, inter-sample distance checking on the first in-view segment sample of each streamline and view-space length checking on the accepted part can prevent the number of streamlines from excessively increasing. The number varies within only a small range to ensure the high efficiency of IVDESS while the greedy scheme works well to achieve effective streamline reuse. Fig. 5 shows the pseudo code for IVDESS' hybrid-space multi-level seeding component based on the temporally coherent seeding strategy.

```
///given a streamline (seed & view-space length), to fill seeding queues
void  FillSeedQueues(STREAMLINE strm, LEN l, VIEW-SPACE SEED s)
{  s introduces two neighboring candidates c1 and c2 with weight l;
   Insert c1 and c2 to primary queue and sort them by weight l;
   for(each non-seed view-space segment sample of strm)
   Push two neighboring candidates to secondary queue tail;
}

///pop a seed from the head of a specified seeding queue for advection
void  PopToAdvect(SEED QUEUE* q)
{  Pop a view-space seed s from q head; if(fail distance check) return;
   Physical-space seed S = unprojection(s);
    if((l = view-space length of streamline strm advected from S ) > L)
    {  Insert strm to curQue and sort it by l; FillSeedQueues(strm, l, s);  }
}

void  HybridSpaceMultiLevelSeeding(FRAME_INDEX f)
{  Swap curQue and preQue;
   if(f == 0)  PoissonDiskDistributionToSecondarySeedQueueTail();
   else while(preQue is not empty)     ///streamline reuse+lengthening
   {  streamline strm = ReuseLengthenStreamline( PopHead(preQue) );
      if(strm's view-space length l > L)
      {  Insert strm to curQue and sort it by l;
         FillSeedQueues(strm, l, projection(strm's physical-space seed));
      }  else  delete strm;
      PoissonDiskDistributionToSecondarySeedQueueTail();
   }
   do                                          ///double-queue seed scheduling
   {  while(primary queue is not empty) PopToAdvect(primary queue);
      if(secondary queue is not empty) PopToAdvect(secondary queue);
   } while(secondary queue is not empty);
}
```

Figure 5. Pseudo code for IVDESS' hybrid-space multi-level seeding component based on the temporally coherent seeding strategy.

3.3 View-Sensitive Streamline Representation

A streamline successfully reused in an IVDESS frame may include an out-of-view part and/or an in-view but rejected part, though neither should be rendered to the output image. To facilitate streamline reuse+lengthening in both directions, the physical-space raw points of a streamline need to be sequentially stored in a buffer for fast access, e.g., to obtain the first in-view segment sample in either direction.

In IVDESS, the negative side of a streamline is stored prior to the positive side to coincide with the flow orientation for easy maintenance of sample time-stamps[8] used in loop detection[16]. A header is attached to the beginning of the streamline buffer to save the number of raw points, the seed's buffer-index, and the view-space streamline length. Two *view-sensitive descriptors* (VSD), one for each direction, are used after the header but before the array of raw points in the buffer to describe the accepted viewable part(s) of the streamline. A VSD stores the physical-space counterpart of the first accepted in-view segment sample — either the seed itself or the unprojection of the sample, the first accepted in-view raw point's buffer-index, the last accepted in-view raw point's buffer-index, and the physical-space counterpart of the last accepted in-view segment sample, which refer to I_0, R_0, R_1, and I_1 in Fig. 3, respectively. There are two additional elements saved in a VSD. One is the instantaneous step size at the streamline end used to maintain a smoothly adaptive step size for streamline lengthening. The other is the closing point employed to form a closed streamline, if any.

Although a streamline may have more than two separated but accepted parts in the view, currently only the two closest parts to the seed are described in IVDESS by two VSDs. The VSDs provide a general description of the accepted viewable parts of a streamline and therefore there may be some redundancy for specific cases, e.g., the redundancy between an in-view seed and the counterpart of the first accepted in-view segment sample (a, b, and c in Fig. 3). Some fields such as the closing point need to be properly padded such that the streamline can be rendered as it is. The extreme scenario is a newly advected streamline accepted in the current frame, for which two VSDs need to be totally made up. This general description allows for universal fast access to streamlines. During greedy streamline reuse+lengthening, two VSDs are dynamically updated to keep track of the change of the accepted parts. For the negative part of a streamline, the reuse+lengthening order is opposite to the storage order. Thus a temporary buffer is used for subsequent rearrangement purposes and the VSD needs to be rearranged too to comply with the storage order or the rendering order.

It is worth mentioning that the *main body* of a streamline buffer stores only the physical-space raw points, excluding the unprojection of any intermediate segment sample. Only temporarily used in the current frame for rendering purposes, the counterpart of a line-view-clipped sample — an unprojection point, is instead stored in a VSD so as to avoid being used in the subsequent frames because unprojection errors can be propagated to a noticeable degree as the user approaches the streamline. Otherwise, a jaggy line segment might result from flow advection when the streamline is lengthened from such an unprojection point — actually not the intended point in physical space (Fig. 6).

View-sensitive streamline representation enables the use of the temporally coherent seeding strategy based on greedy non-split streamline reuse+lengthening for view-driven evenly spaced streamline placement. On one hand, it allows streamlines to be thoroughly reused to enhance inter-frame coherence. On the other hand, it ensures streamlines are properly rendered in the current frame without introducing jaggy lines in the subsequent frames.

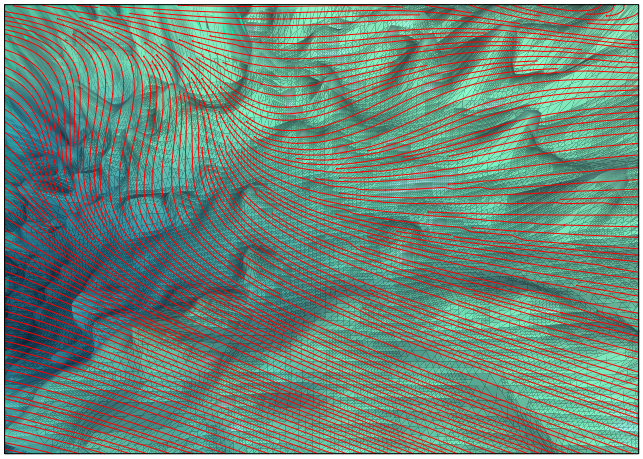
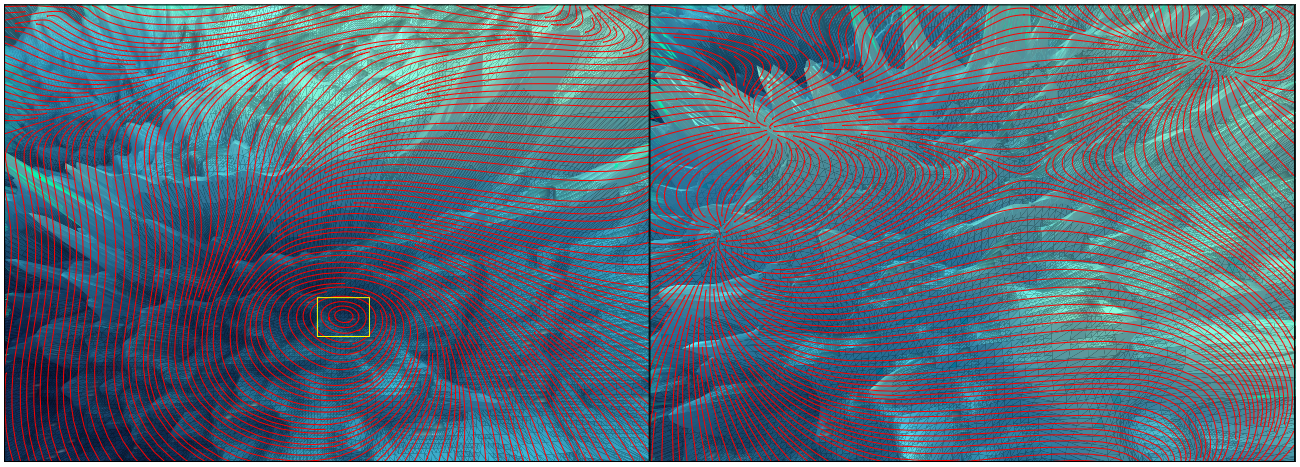


Figure 6. Physical-space points that are obtained by unprojecting line-view-clipped intermediate segment samples need to be rendered to the current frame to show streamlines properly. However, jaggy lines may emerge if unintended unprojection points are used in the subsequent frames (e.g., here one such frame of an ocean flow over the bathymetry) to lengthen streamlines.

4. RESULTS AND DISCUSSIONS

The design of IVDESS allows the algorithm to be applicable to interactive 3D exploration of not only a planar flow but also the flow on either a model surface or an extracted streamsurface/isosurface. In fact, physical-space streamline integration is almost transparent to view-space inter-sample distance control. The former just needs to provide streamline points, one for each time to the interface — view projection, unless the latter rejects any view-space sample. Thus IVDESS is pretty amenable to flows on complex grids. Streamline integration modules for various kinds of grids can be easily connected with the view-dependent streamline density controller of IVDESS. The computational expense of an IVDESS placement is mainly determined by other components of the pipeline than streamline integration, particularly when we consider the high performance of an adaptive step size integrator used in IVDESS. Given a threshold distance in pixels, the size of a 2D view, and the negligence of such issues as data loading, memory access, and cache coherence, the overall cost is roughly independent of the flow size as the number of streamlines in any placement falls in a small range. Thus IVDESS can handle huge flows at an interactive and nearly constant frame rate without any pre-processing.

Currently we have implemented IVDESS for 3D exploration of planar flows in a perspective-view setting and tested it on a PC (Celeron M 1.60GHz, 512MB RAM, Windows XP) without GPU support, which is a rather ordinary platform. To test the view-driven streamline placement quality, the temporal coherence performance, as well as the placement generation speed and the placement generation+rendering speed of IVDESS, we chose a 468 × 337 2D flow field of the Northeast Pacific Ocean, in which there are over one hundred critical points making very complex flow patterns. Perspective projection was configured as: near clipping plane = 1.0, far clipping plane = 10000.0, field-of-view angle = 90.0°, aspect ratio = 1.0, and view size = 990 × 700 (in pixels). The error tolerance range used to govern the adaptive step size (initialized to 0.0625) was [0.000001, 0.00001] in cells. The separating distance threshold and the view-space streamline length threshold were set to 10 and 30 in pixels, respectively. In addition, the ocean bathymetry was rendered to the output image to improve the perception of the 3D scene. Fig. 7 shows two of 100 frames of flow exploration generated using IVDESS with the *t*emporally *c*oherent *s*eeding *s*trategy (IVDESS-TCSS, or IVDESS by default). The user interaction log was then accessed to programmably repeat exactly the same exploration to produce another sequence of 100 frames using IVDESS with the *t*emporally *i*ncoherent *s*eeding *s*trategy (IVDESS-TISS). The statistics for IVDESS-TCSS and IVDESS-TISS in creating their respective 100 frames are shown in Figures 8-11.



(a) (b)

Figure 7. (a) Streamlines are visually evenly-spaced in an IVDESS-TCSS placement without cluttering or distracting discontinuities and there are three closed streamlines (marked by the rectangle) effectively recognized by IVDESS' loop detector. (b) Streamlines are placed considerably well around critical points, even without topology-based seed distribution[13], in an IVDESS-TCSS placement.

As shown in Fig. 7a, streamlines are evenly spaced in the output image of an IVDESS-TCSS placement without cluttering or distracting discontinuities. In particular, there are three closed streamlines successfully detected and properly formed by the intra-line distance controller[16] (loop detector) of IVDESS. The IVDESS-TCSS layout in Fig. 7b demonstrates the capability of our seeding strategy, even without topology-based seed distribution[13], in placing evenly spaced streamlines around critical points. Fig. 8 shows the number of IVDESS-TCSS-reused streamlines, the number of IVDESS-TCSS-advected streamlines, the number of IVDESS-TCSS streamlines, and the number of IVDESS-TISS streamlines in each frame. For more than half of the IVDESS-TCSS frames, there are far more reused streamlines than advected ones per frame. Even for the other frames, the number of reused streamlines is only a little bit less than that of advected ones per frame. This demonstrates the effectiveness of the greedy non-split streamline reuse+lengthening scheme adopted by TCSS, which is also reflected by the high streamline reuse percentages of IVDESS-TCSS in Fig. 9. As shown in Fig. 8, IVDESS-TCSS and IVDESS-TISS placed a very approximate number of streamlines in their respective frames. This indicates the high performance of TCSS in preventing the number of streamlines from excessively increasing and hence the high efficiency in reusing streamlines to maintain temporal coherence. Fig. 10 compares IVDESS-TCSS with IVDESS-TISS in frame generation time and in frame generation+rendering time (note that a large portion of the rendering time was spent on the entire illuminated ocean bathymetry). For nearly every frame and for either case, less time was consumed by IVDESS-TCSS than by IVDESS-TISS. This shows that TCSS is able to achieve not only temporal coherence but also speedup in placement generation as demonstrated in Fig. 11 with respect to frame rate. Furthermore, Fig. 10 shows that the variation in the time required to generate a frame (as well as the variation in the time required to generate and render a frame) by IVDESS-TCSS was considerably less than that by IVDESS-

TISS. The nearly constant frame generation rate of IVDESS-TCSS adds great strength to the conclusion that IVDESS-TCSS, i.e., IVDESS, is capable of interactive coherent flow exploration.

5. CONCLUSIONS AND FUTURE WORK

We have presented IVDESS, an interactive view-driven evenly spaced streamline placement algorithm for physically non-uniform, but visually uniform, representation of planar or curved surface (model surface, streamsurface, isosurface) flows in a perspective-view setting. A projection-unprojection pair is employed through off-screen surface rendering to link physical-space streamline integration with view-space density control to distribute streamlines that are evenly spaced in the output image — a view-dependent placement. To maintain smooth inter-frame transition, physical-space seeding based on streamline reuse+lengthening is used prior to view-space seeding to constitute a temporally coherent seeding strategy. A view-sensitive streamline representation is adopted for streamline reuse, lengthening, and rendering without introducing temporal incoherence or causing jaggy lines. Our tests demonstrate the practical applicability of IVDESS to coherent level-of-detail 3D exploration of large complex flows at interactive frame rates without either pre-processing or GPU support.

As for future work, we plan to enhance the current version of IVDESS with streamline integration on curvilinear and unstructured grids. We will also conduct research on adaptive depth selection issues in an effort to extend IVDESS for explorative visualization of volume flows.

ACKNOWLEDGMENTS

This work was supported by the DoD HPCVI program. The authors would like to thank Dr. David L. Kao for his inspiring discussions and insightful suggestions. Thanks go to the anonymous reviewers for their valuable comments.

REFERENCES

1. J. J. van Wijk, "Spot noise: texture synthesis for data visualization," *Proc. SIGGRAPH'91*, pp. 309-318, 1991.
2. B. Cabral and L. Leedom, "Imaging vector fields using line integral convolution," *Proc. SIGGRAPH'93*, pp. 263-270, 1993.
3. B. Jobard, G. Erlebacher, and M. Y. Hussaini, "Lagrangian-Eulerian advection of noise and dye textures for unsteady flow visualization," *IEEE Trans. Visualization and Computer Graphics*, vol. 8, no. 3, pp. 211-222, 2002.
4. J. J. van Wijk, "Image based flow visualization," *Proc. SIGGRAPH'02*, pp. 745-754, 2002.
5. G. Turk and D. Banks, "Image-guided streamline placement," *Proc. SIGGRAPH'96*, pp. 453-460, 1996.
6. B. Jobard and W. Lefer, "Creating evenly spaced streamlines of arbitrary density," *Proc. Eighth Eurographics Workshop on Visualization in Scientific Computing*, pp. 45-55, 1997.
7. A. Mebarki, P. Alliez, and O. Devillers, "Farthest point seeding for efficient placement of streamlines," *Proc. Visualization'05*, pp. 479-486, 2005.
8. Z. Liu, R. J. Moorhead II, and Joe Groner, "An advanced evenly spaced streamline placement algorithm," *IEEE Trans. Visualization and Computer Graphics*, Vol. 12, No. 5, pp. 965-972, 2006.
9. D. C. Banks, "Illumination in diverse codimensions," *Proc. SIGGRAPH'95*, pp. 327-334, 1995.
10. X. Mao, Y. Hatanaka, H. Higashida, and A. Imamiya, "Image-guided streamline placement on curvilinear grid surfaces," *Proc. Visualization'98*, pp. 135-142, 1998.
11. B. Jobard and W. Lefer, "Multiresolution flow visualization," *Proc. WSCG'01,* pp. 33-37, 2001.
12. B. Jobard and W. Lefer, "Unsteady flow visualization by animating evenly spaced streamlines," *Proc. Eurographics'00,* pp. 21-31, 2000.
13. V. Verma, D. Kao, and A. Pang, "A flow-guided streamlines seeding strategy," *Proc. Visualization'00*, pp. 163-170, 2000.
14. O. Mattausch, T. Theubl, H. Hauser, and E. Groller, "Strategies for interactive exploration of 3D flow using evenly spaced illuminated streamlines," *Proc. Nineteenth Spring Conf. on Computer Graphics*, pp. 213-222, 2003.
15. L. Li and H.-W. Shen, "Image-based streamline generation and rendering," *IEEE Trans. Visualization and Computer Graphics*, Vol. 13, No. 3, pp. 630-640, 2007.
16. Z. Liu and R. J. Moorhead II, "Robust loop detection for interactively placing evenly spaced streamlines," *IEEE Computing in Science and Engineering*, Vol. 9, No. 4, pp. 86-91, 2007.
17. X. Ye, D. Kao, and A. Pang, "Strategy for seeding 3D streamlines," *Proc. Visualization'05*, pp. 471-478, 2005.

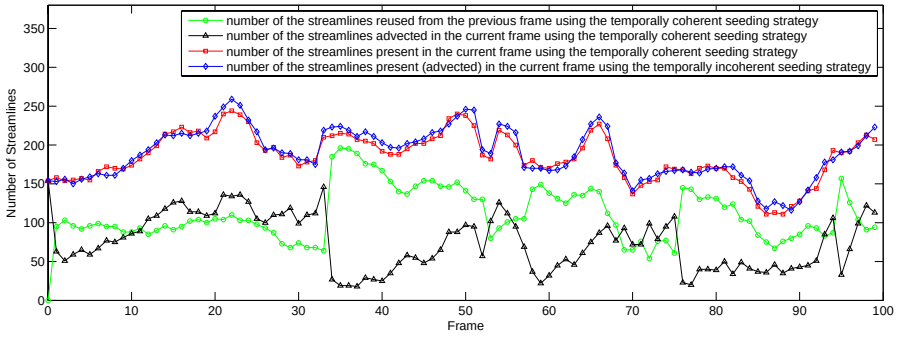


Figure 8. The number of IVDESS-TCSS-reused streamlines, the number of IVDESS-TCSS-advected streamlines, the number of IVDESS-TCSS streamlines, and the number of IVDESS-TISS streamlines in each frame.

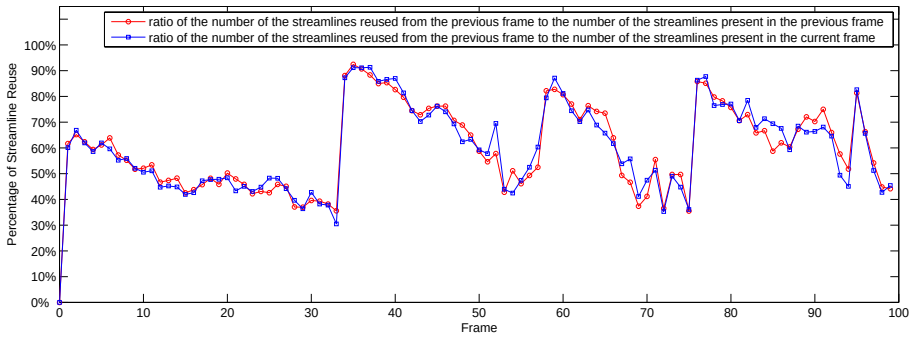


Figure 9. The percentages of IVDESS-TCSS-reused streamlines relative to the previous frame and the current frame, respectively.

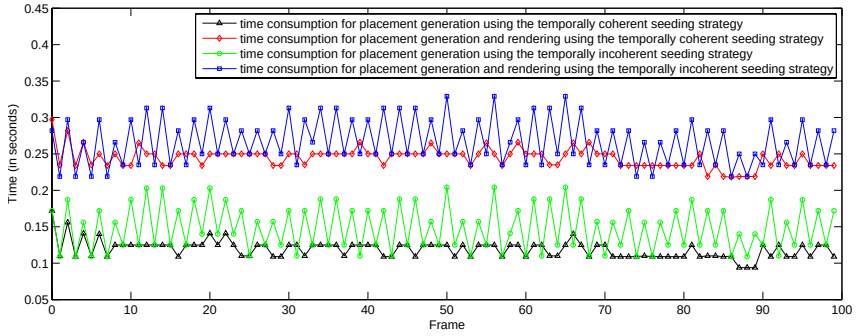


Figure 10. IVDESS-TCSS compared with IVDESS-TISS in frame generation time and in frame generation+rendering time.

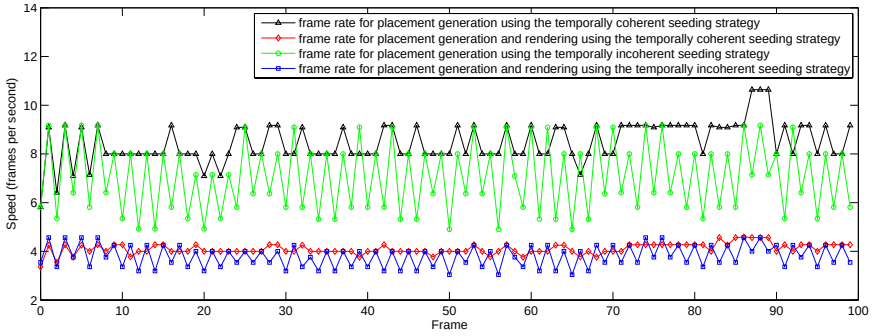


Figure 11. IVDESS-TCSS compared with IVDESS-TISS in frame generation rate and in frame generation+rendering rate.