On the Computation of Integral Curves in Adaptive Mesh Refinement Vector Fields

Eduard Deines[1], Gunther H. Weber[2], Christoph Garth[1], Brian Van Straalen[2], Sergey Borovikov[3], Daniel F. Martin[2], and Kenneth I. Joy[1]

1 **University of California, Davis, USA**
{edeines,cgarth,kijoy}@ucdavis.edu
2 **Lawrence Berkeley National Laboratory, Berkeley, USA**
{ghweber,bvstraalen,dfmartin}@lbl.gov
3 **University of Alabama, Huntsville, USA**
{snb0003}@uah.edu

Abstract

Integral curves, such as streamlines, streaklines, pathlines, and timelines, are an essential tool in the analysis of vector field structures, offering straightforward and intuitive interpretation of visualization results. While such curves have a long-standing tradition in vector field visualization, their application to Adaptive Mesh Refinement (AMR) simulation results poses unique problems. AMR is a highly effective discretization method for a variety of physical simulation problems and has recently been applied to the study of vector fields in flow and magnetohydrodynamic applications. The cell-centered nature of AMR data and discontinuities in the vector field representation arising from AMR level boundaries complicate the application of numerical integration methods to compute integral curves. In this paper, we propose a novel approach to alleviate these problems and show its application to streamline visualization in an AMR model of the magnetic field of the solar system as well as to a simulation of two incompressible viscous vortex rings merging.

Keywords and phrases integration-based visualization, streamlines, interpolation, adaptive mesh refinement

Digital Object Identifier 10.4230/DFU.xxx.yyy.p

1 Introduction

Simulation and visualization are key components of the modern scientific knowledge discovery pipeline across many different disciplines. In this context, the task of visualization is to provide insight into science problems from raw data generated by measurement or computer simulation. In certain application areas, such as astrophysics, the simulation must accommodate many different spatial scales. For example, in simulations of the solar system, the computational domain may span thousands of astronomical units (AU), whereas some physical structures that need to be resolved occupy significantly shorter length scales. As a consequence, it is not possible to use regular grids, which in many cases are a primary choice due to their conceptual simplicity, to discretize such simulation domains. The uniform resolution required to represent the smallest phenomena in a simulation would result in infeasible overall storage requirements. Unstructured grids, on the other hand, adapt well to differences in length scales, but they also introduce significant overhead by requiring an explicit representation of grid connectivity. Adaptive mesh refinement (AMR) techniques are a hybrid solution that discretizes the simulation domain as set of overlapping, nested, axis-aligned rectilinear

grids [4]. These grids are arranged in levels of increasing resolution, and information for a specific point of the domain is stored at the finest grid overlapping this point. Thus, AMR combines the adaptivity of unstructured grids with the implicit connectivity of regular grids, adding only little overhead in form of a layout description. This arrangement makes it possible to discretize adaptively simulations with comparatively little overhead.

In the visualization process, AMR data presents two challenges not found in other data representations. First, portions of the coarse grid that span the simulation domain are replaced with higher-accuracy information provided by grids at levels of higher resolution. Thus, it is necessary to detect this overlap and always consider data at the finest resolution available. Second, for methods that are built on the existence of a continuous interpolant over the entire data set, special care must be taken to reconcile coarse and fine information resolution at resolution boundaries to yield continuous interpolation. AMR simulations often specify cell-centered values, which further complicates interpolation.

Previous work on addressing these problems for the visualization of AMR data focused mainly on scalar field visualization techniques such as isosurface extraction and direct volume rendering (Section 2.2 provides a brief overview). In this paper, we focus on making it possible to use integration-based visualization techniques on AMR data sets. For this purpose, we investigate existing streamline construction algorithms and apply them to AMR vector fields. The goal of this effort is to identify problems that preclude efficient and accurate visualization of AMR vector fields and then develop a novel algorithm to address these problems. We perform a numerical evaluation of the accuracy of our method, compare it against existing approaches, and document the applicability of our scheme by using it to visualize two AMR data sets from astrophysics and fluid flow simulations. Finally, we discuss limitations of our approach and possible future extensions to overcome them.

2 Background and Related Work

2.1 Adaptive Mesh Refinement (AMR)

Adaptive Mesh Refinement (AMR), introduced by Berger and Oliger [3], combines the topological (structural) simplicity of regular, rectilinear grids with the adaptivity of unstructured grids to local changes in resolution: The computational domain is discretized using one or more coarse grids (also called boxes or patches) in a *root level*. During the simulation, an error estimate is computed for each cell of this root level, and all cells whose error estimate exceeds a given threshold are tagged for refinement. Subsequently, a set of rectilinear grids with an increased resolution is created that overlaps all tagged cells. These tagging and grid creation steps are repeated recursively until all regions are represented with adequate resolution. Simulation results from higher-refined levels are propagated back to the coarser levels, resulting in a consistent representation of the domain. In Berger and Oliger's original approach [3], newly created grids are structured, rectilinear grids that can be rotated with respect to the parent level. Berger and Collela published a modified version [4] of this algorithm where newly created grids are axis aligned with respect to the parent level, which is now widely used for simulations in applied science and engineering [35, 9, 12, 14, 31, 13, 1].

A Berger-Colella AMR hierarchy consists of axis aligned structured grids as building blocks (see Fig. 1 for an example). These grids are arranged in levels, starting at the coarsest root level up to the finest level of resolution. The hierarchy shown in Fig. 1, for example, consists of three levels, including the root level. Within a level, grids have identical resolutions. Going from one level to the next finer level, a coarser grid cell is alway split into an integer number of finer grid cells, where the integer *refinement ratio* specifies the number of fine

grid cells per coarse grid cell along each axis. The AMR hierarchy shown in Fig. 1 has a refinement ratio of two, meaning that each coarser grid cell is covered by two finer grid cells along each axis. Since the depicted data set is defined on a two-dimensional domain, each coarser grid cell is covered by a total of four finer cells. Furthermore, a fine grid always starts at the boundary of a coarse grid cell and a coarser grid cell is always either refined completely or not at all. An AMR level is always contained completely within the next coarser level. While a single grid (e.g., the grid comprising the third level in Fig. 1) at a given level may overlap more than one grid of the next coarser level, it is always enclosed in the union of all grids of that coarser level. Finally, transitions only occur between two consecutive levels. There is always a layer of grid cells of the next coarser level around all grids of a refining level.

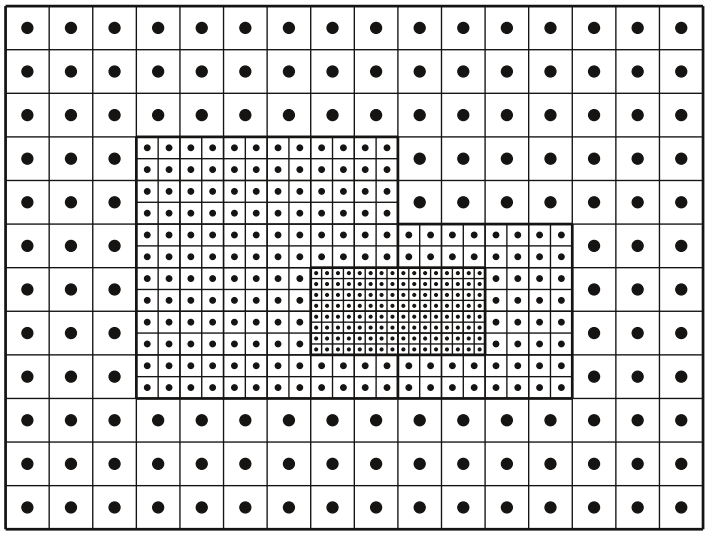


Figure 1 AMR mesh consisting of four grids arranged in three refinement levels.

An important aspect in the parallelization of simulations working on structured, rectilinear grids (AMR and other approaches that utilize multiple grids) is the concept of *ghost cells*, see Fig. 2. Most simulation codes require the computation of derivatives and approximate these derivatives based on the difference between values in cells adjacent to the current cell; however, cells at boundaries do not have neighbors in all directions. To make it possible to handle all cells in a grid in the same way, one introduces ghost cells to the simulation. These ghost cells can either be external to the simulation, i.e., lie outside the computational domain, or they can be located inside the domain. The latter case occurs if the domain is partitioned into multiple blocks. Where blocks are adjacent to each other, ghost cells replicate data from adjacent blocks, reducing the need for communication in parallel computation. In any case, ghost cells are not considered to be part of the actual simulation domain. Note that if one considers ghost cells, finer AMR levels do not necessarily start at coarser cell boundaries.

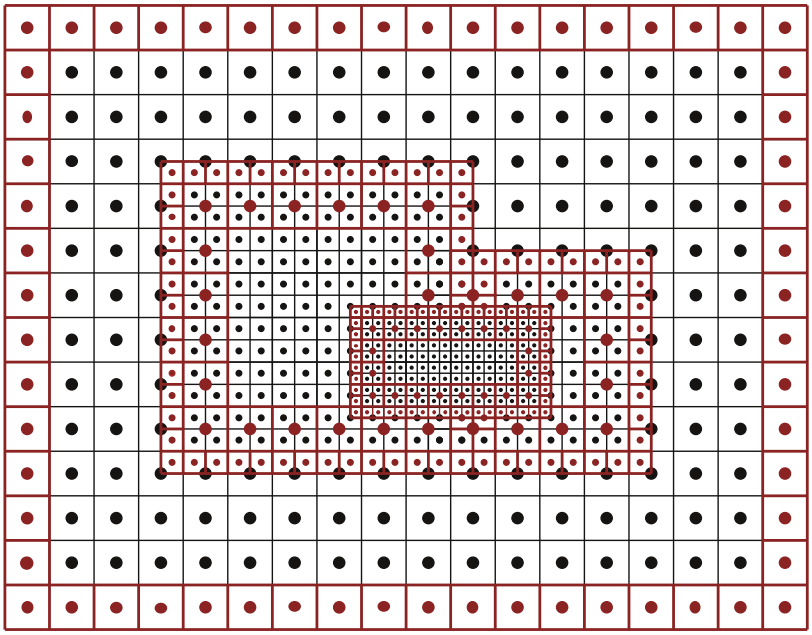


Figure 2 Ghost cells create overlap between resolution levels and allow all cells of a level to have a uniform neighborhood.

2.2 Visualization of AMR Data Sets

Much of the existing research on AMR visualization is focused on the visualization of scalar fields and particularly on the development of effective volume rendering and iso-surface extraction algorithms. Early approaches worked around problems with AMR data in visualization by converting it to a form compatible with existing software implementations, such as regular or unstructured meshes [28]. However, in this conversion the implicit mesh connectivity is lost together with the storage efficiency advantage of AMR over unstructured-grid-based approaches.

Specifically treating the case of iso-surface extraction, Weber et al. developed a method that addresses the discontinuities of interpolation across resolution level boundaries to extract crack-free iso-surfaces from cell-centered AMR data [38, 39]. Similar to our work described below, cell centers of each patch of an AMR hierarchy are interpreted as vertices of a new patch, forming the *dual grid* to the original patch. The use of dual grids leads to gaps between patches comprising different AMR hierarchy levels. Weber et al. used a procedural scheme to fill these gaps with so-called *stitch cells*. Fang et al. [10] presented an alternate iso-surface extraction approach for node-centered AMR data with the main goal of preserving the original patch structures.

Research concerned with volume rendering includes effective sorting schemes for cell-based volume rendering [26], accelerating AMR volume rendering using graphics hardware [37, 20, 16, 29, 18], dealing with interpolation issues [16, 40] and allowing for parallel approaches [23, 20, 41]. Furthermore, Kähler et al. [15] used a set of existing tools to render star formation simulation data produced by the framework developed by Bryan et al. [6]. Finally, Kähler et al. [17] also described a framework for remote visualization of time-varying AMR data.

2.3 Integration-Based Visualization

In the context of vector fields, *integration-based visualization* aims at using the trajectories of idealized, massless particles as basic primitives to construct insightful visualization methods. These particle trajectories, also called *integral curves*, are given as ordinary differential equations over the vector field of interest and are approximated using numerical integration schemes, such as e.g. the class of *Runge-Kutta* schemes. Here, we make use of a scheme by Dormand and Prince termed DoPri5 [32, 33], that is popular for the task of integral curve approximation [11, 34]. It offers 4th-order accuracy, adaptive stepsize control, i.e. the ability to automatically adjust the integration step size based on the complexity of the vector field in the traversed region, and dense output in the form of a polynomial curve describing every integration step.

Beyond direct depiction in the form of *streamlines* (in stationary vector fields) and *pathlines* (in time-varying vector fields), integration-based visualization techniques also include more general techniques. Integral surfaces depict the movement of a family of particles originating on a common curve or surface; this results in an illustrative depiction of vector field structures and avoids visual problems that are often incurred when visualizing many integral curves simultaneously [11]. Topological methods derive specific integral curves, so-called *separatrices*, from an analysis of a vector field's critical points, providing an abstract, skeleton-type overview of a vector field [36]. *Dense methods* focus on depicting all particles traversing a vector field simultaneously to e.g. mimic the effect of flow-induced oil smearing in physical experiment [21].

All these methods rely on the reliable and accurate computation of integral curves of the considered vector field. AMR vector fields present a number of challenges in this context. A number of publications demonstrate streamlines over adaptive meshes [2, 27, 22]; however, they are used as a visual benchmark for the described simulation or grid conversion methods using adaptive techniques, and a description of the used integration algorithm is not provided.

2.4 Problems with Integral Curves in AMR Data

Since the mesh consists of multiple refinement levels, each of which may contain several domains represented as rectilinear grids, the data in the finer levels replaces the coarser level information. Because individual domains across all levels overlap, it is necessary to obtain information from the finest level to represent the vector field data in full fidelity. Identifying the correct domain and level can be a difficult task; below, we present an approach to handle this efficiently. Moreover, due to the typical nature of AMR simulation codes, the vector field is provided on a per-cell basis. Thus, interpolation is required to achieve the requirement of continuity of the vector field for the application of numerical integration. While many interpolation schemes may accommodate this requirement, tri-linear interpolation is typically chosen for efficiency reasons. Here, we examine and compare the typically used cell-averaging interpolation and contrast it with a novel dual-mesh approach that promises better accuracy. Finally, due to the discontinuity of the vector field interpolant when traversing resolution level boundaries, significant errors can be introduced during numerical integration, leading to less accurate or incorrect integral curve approximations. Below, we demonstrate how these boundaries can be explicitly taken into account during integration to avoid this loss of accuracy and correctness.

3 Integral Curves in AMR Vector Fields

In this section we describe several possibilities how we can address the problems of integral curve computation in AMR data sets. Interpolating only in the root grid leads to large differences in resulting streamlines when compared to utilizing properly the data present in all hierarchy levels (see Figure 12). Thus, it is essential to consider the AMR hierarchy. We have examined two different possibilities to handle the AMR refinement grids. To calculate a single integration step, the DOPRI54 algorithm needs to perform six vector field evaluations. In our first method, we identify the corresponding AMR grid in the finest possible level for each evaluation point and evaluate the vector field on this grid. The finest domain can be found efficiently using an interval tree structure. Here, we do not consider the boundaries between the different domains and levels. As a consequence, any possible discontinuity issues mentioned above are captured by the error estimation of the DOPRI54 method, thereby reducing the integration step size and resulting in a larger number of integration steps (many failed steps) during integral curve computation. Additionally, the computational error in the numerical solution after passing the discontinuity is usually larger than the user prescribed tolerance (see [7]). The individual steps of this approach are summarized in Algorithm 1.

Algorithm 1 AMR_Ignore_Boundaries

```
start in domain in finest possible level
while not finished do
  advance integration step {
  for each vector field evaluation do
    find domain in finest possible level
    evaluate field in this domain
  end for
  }
  add step to curve
end while
```

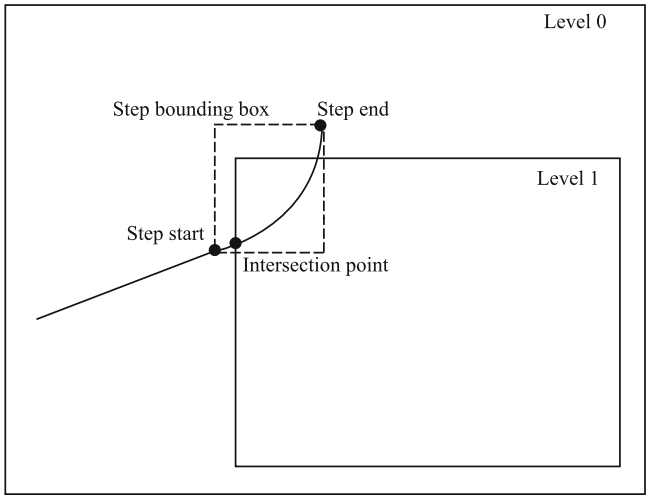


Figure 3 Streamline intersection with level boundary.

The error due to the discontinuity can be avoided by localizing the discontinuity position and stopping and re-starting integration at the discontinuity introducing minimal possible error (see [24, 7]). In our case, discontinuities are located at level boundaries. Thus, we have to calculate an intersection point of the integral curve with the boundaries of the nested refined grids. This is also the transition point where we have to descend into the finer level. We formulate the second approach as follows. We start the integration in the domain in the finest possible level. After each integration step we determine whether a refined region was crossed. For this purpose, we detect whether the bounding box surrounding the current integration step intersects the bounding box containing all the nested grids in the considered domain. If so, we then check for a possible intersection with the boundary of the child domains. Once we have determined an intersection point we can restart the integration in the corresponding nested domain with the intersection position as starting point. This process is illustrated in Figure 3. The Dopri54 method provides a polynomial representation of the integral curve in each integration step. We use Newton's method to calculate the intersection point. The pseudo code of this approach is outlined in Algorithm 3.

Algorithm 2 Integrate_Domain

```
while inside domain do
  advance integration step
  if cross ghost cells then
    if inside children bounding box then
      if found child then
        intersect with child bounding box
        advance to intersection point
        set new domain to child domain
        outside domain
      else
        add step to curve
      end if
    else
      intersect with domain bounding box
      advance to intersection point
      outside domain
    end if
  else
    add step to curve
  end if
end while
```

Algorithm 3 AMR_Explicit_Boundaries

```
while not finished do
  find domain in finest possible level
  Integrate_Domain (see Algorithm 2)
end while
```

The advantage of the second approach, compared to the first, simpler method, is that we can process the curve integration in each AMR domain separately. In the first approach we additionally need at least the fields of the nested child domains and the neighbor domains in each integration step.

As mentioned in the previous section, the simulated data is mostly cell-centered whereas integration requires vertex-centered vector field data. Only using the vector value given in a cell independently of the position (nearest neighbor interpolation) of the evaluation point inside the cell is not sufficient. Thus, we have to calculate the vector values at grid vertices based on the given cell vector values. Computing the vector at the vertex position as average of the vectors inside the cells that surround a node yields a continuous trilinear interpolant. However, it also has limitations due to very strong smoothing and inaccurate calculation of field values at grid boundaries. Since an AMR hierarchy usually consists of a large number of grids, this method can result in a significant error during integral curve computation.

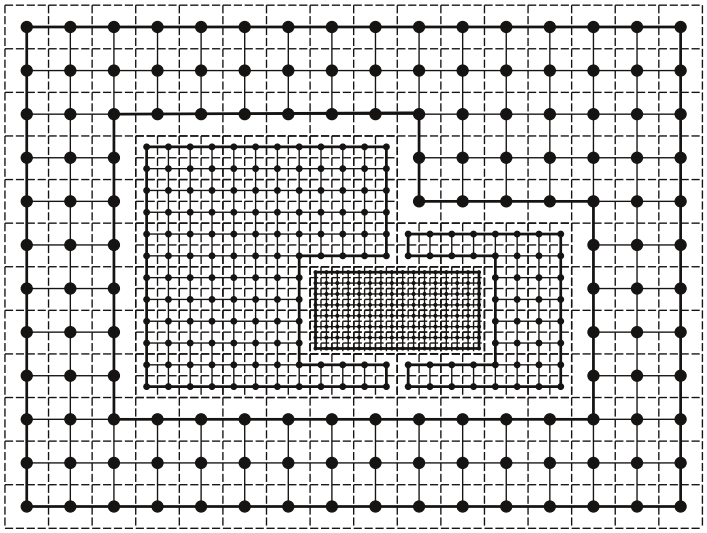


Figure 4 Dual-grid representation without additional ghost cells showing gaps between different AMR grids.

A proven approach in scalar visualization over AMR meshes with empirically better interpolation accuracy for a cell-centerd vector field is a dual-mesh interpolant. The set of all cell centers of the original grid form a grid, called the dual grid. In the case of block-structured AMR the dual grid is also rectilinear. AMR data contains several independent grids. Constructing the dual grid for each of them independently produces gaps between the neighboring domains as shown in Figure 4. A solution to this problem is the computation of an additional ghost layer surrounding the domain grid (see Figure 2). This implies extra data to be stored and processed, but the resulting dual grid of the entire AMR mesh is connected (see Figure 5). Thus, we can use the cell values for integral curve computation. This additional ghost cell layer is often generated during the simulation process.

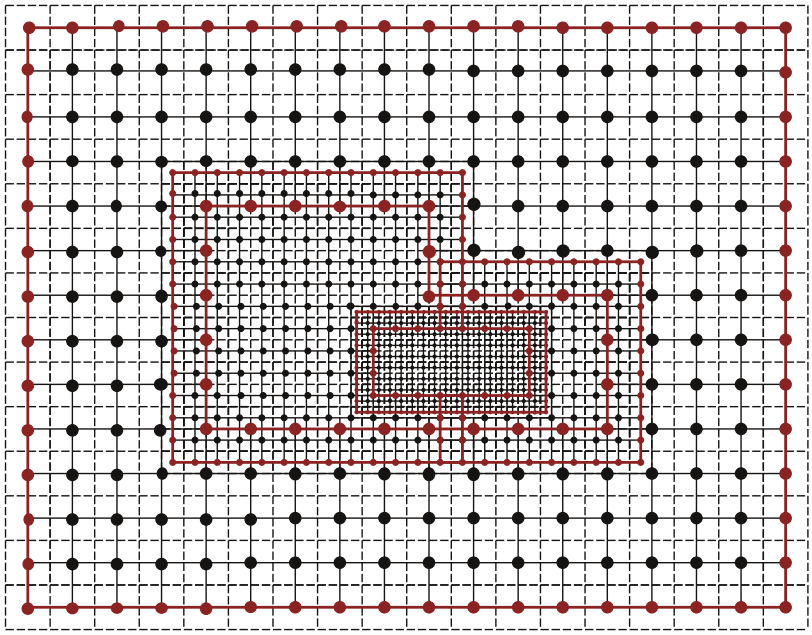


Figure 5 Dual-grid representation with additional ghost cells. The boundaries between different AMR domain grids (red lines) overlap showing no gaps.

4 Numerical Evaluation

To examine and evaluate the different methods for the integral curve computation described in the subsections above we perform the following test (see Algorithm 4). We used an 2D

Algorithm 4 Test_Case

Case: AI
cell data to point data using averaging
execute AMR_Ignore_Boundaries
Case: AE
cell data to point data using averaging
execute AMR_Explicit_Boundaries
Case: DI
build dual-grid
execute AMR_Ignore_Boundaries
Case: DE
build dual-grid
execute AMR_Explicit_Boundaries

AMR mesh with 4 refinement levels and 172 domains. On this mesh we defined a cell-based vector field for which an analytical solution is easy to calculate by:

$$v = \begin{pmatrix} -rsin(\alpha) \\ rcos(\alpha) \end{pmatrix} \tag{1}$$

with $r = \sqrt{x^2 + y^2}$, $\alpha = atan2(y, x)$, and (x,y) are the positions of the cell centers. We calculate integral curves over the interval $[0, 2\pi]$ in the field v and evaluated the relative error

of the final point:

$$E_r = \frac{\|C(2\pi) - S(2\pi)\|}{\|S(2\pi)\|} \tag{2}$$

where C is the computed integral curve and S the exact solution.

In order to make the error calculation more significant and exclude artifacts, we use 100 different initial values and computed the average relative error of these runs. The results are plotted on a log-log scale. Figure 6 depicts the relative error of the test cases AI and DI. For comparison we calculated two additional error plots. The first one shows the error of the integration in the field without using any field interpolation. The integrator uses the field value of the corresponding cell regardless of the evaluation position inside the cell. The second represents the error of the integral curve computation only in the coarsest level of the AMR mesh. In Figure 7 the error plots of all four test cases are compared. Finally, Figure 8 depicts the average number of vector field evaluations needed to calculate the curves for all four test cases.

As we can see from the resulting errors, interpolation is required in order to obtain reasonable results. By using the averaging method problems resulting from discontinuous behavior at level boundaries are clearly seen, introducing a large computational error and an increase in the amount of vector field evaluations. However, this error can be reduced by considering the domain boundaries. The dual grid interpolation shows better results compared to the averaging method. Even if we disregard the domain boundaries, the error is small. The cases DI and DE show equal behavior and the resulting error is smaller compared to the error resulting from the integration in the coarsest level of the AMR. One problem that may occur by disregarding the domain boundaries is that the integrator would not descend into refined levels and important features of the considered vector field can be missed (see for example sketch in Figure 3 and Figure 12). Furthermore, for the method in test case DI, we need to know the entire data set for each integration step, whereas in case DA the integration can be performed for each domain independently.

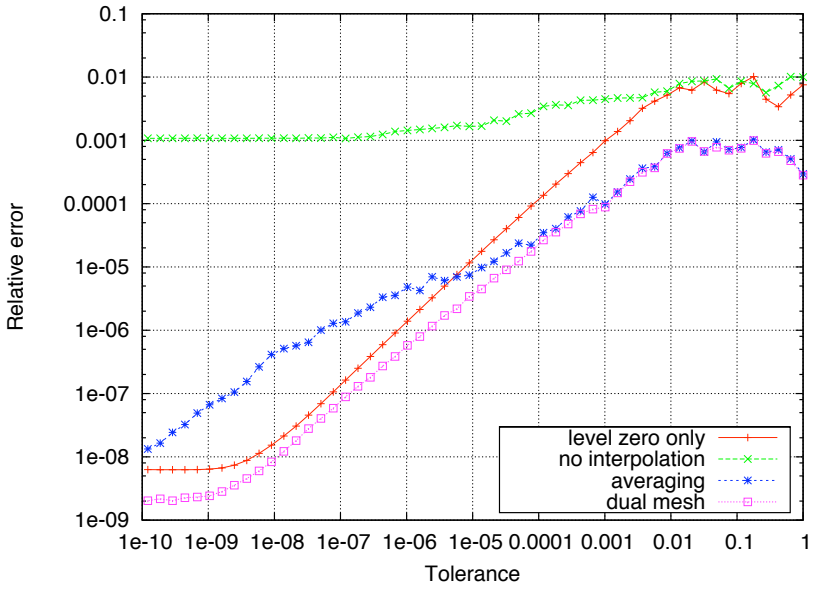


Figure 6 Comparison of relative error for test cases AI and DI.

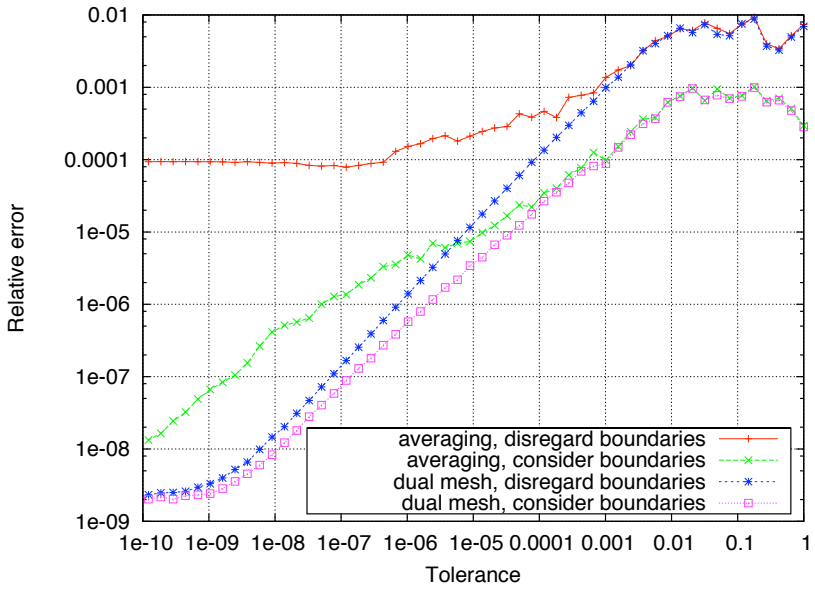


Figure 7 Comparison of relative error for the four test cases.

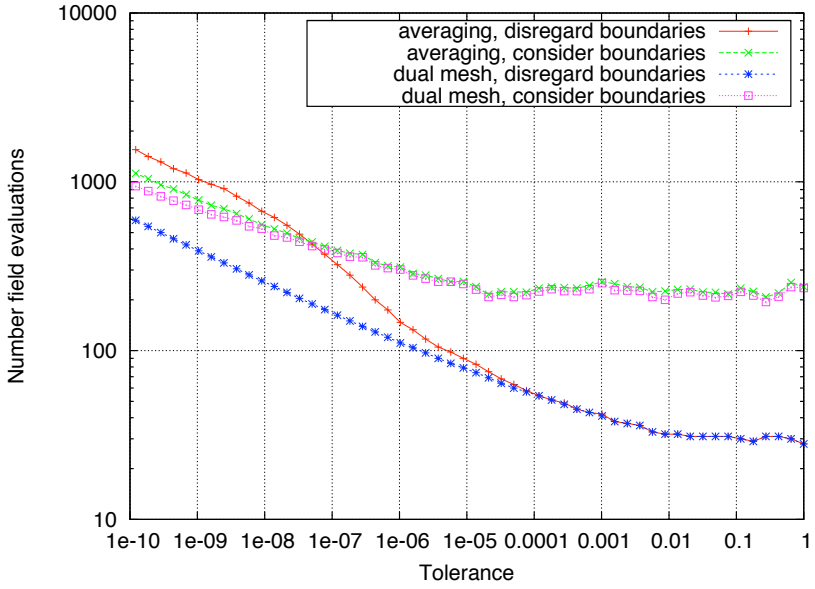


Figure 8 Number of evaluations for the four test cases.

5 Implementation

We have implemented our algorithm using VisIt [8], an open source visualization and analysis tool for large data sets. We chose VisIt as a basis for our work since it already supports a wide variety of AMR file formats. VisIt accommodates AMR as first class data type. It handles coarse cells in AMR data that are also available in a finer resolution level as a special case of "ghost data," i.e., data that is used to make computations more efficient, but which is not considered to be part of the simulation result. This marking is done by adding an array to each box/grid that designates the status of each cell (ghost or non-ghost).

VisIt's marking of refined AMR cells as ghost is facilitated by another data structure, the so called "domain nesting." This data structure specifies which grids/boxes are adjacent to each other and makes it possible to populate boundary ghost cells of one mesh with copied data from an adjacent box. Furthermore, this nesting information also specifies whether a box/grid contains any boxes of a finer level. For example, it is possible to query what boxes/grids of the next finer level intersect the current box/grid and what cells are covered. This information is given as a set of ranges.

VisIt also supports vector field visualization, for example with vector glyph plots and streamlines. However, the current implementation of streamlines in VisIt does not consider an AMR hierarchy. Instead, streamlines that are started in one grid/box of the domain only use information from that box/grid until they exit its *outer* boundaries. In particular, streamlines do not use ghost information to detect refined cells and appropriately descends into the AMR hierarchy.

Algorithm 5 AMR_Integrate_Curve

```
if no outer ghost zones then
  compute additional ghost layer
end if
while not finished do
  find domain in finest possible level
  build dual grid
  Integrate_Domain (see Algorithm 2)
end while
```

Based on the comparison results in section 4 we have implemented the method outlined in Algorithm 5 for the streamline visualization in our application cases. In order to descend into finer regions of the AMR grid, we utilize VisIt's data structures holding the nesting information, the ghost cell array, and the domain nesting described above. The ghost cell array specifies which cells of the coarse grid are refined. The domain nesting structure can be used to obtain the entire region of the finer domains (children domains) to construct the bounding boxes needed for the intersection calculations. If the simulation data do not provide additional ghost cells, we calculate them as follows: If an adjacent grid with equal refinement resolution exists, we use the values in the adjacent cells of the neighbor grid to generate the external ghost layer. If the domain is at the refinement level boundary we interpolate the ghost values using the coarser cell values. Note, that after the calculation of the dual grid we have to adapt the ghost cell array to the new grid, since the number of cells has changed.

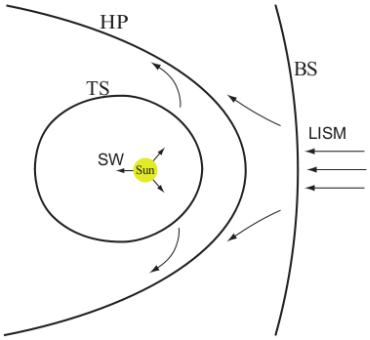
6 Use Case - Application

In this section we will use our integration algorithm in order to visualize streamlines in two different application cases, the simulation of the solar wind interaction with the interstellar medium as well as a simulation of two incompressible viscous vortex cores merging.

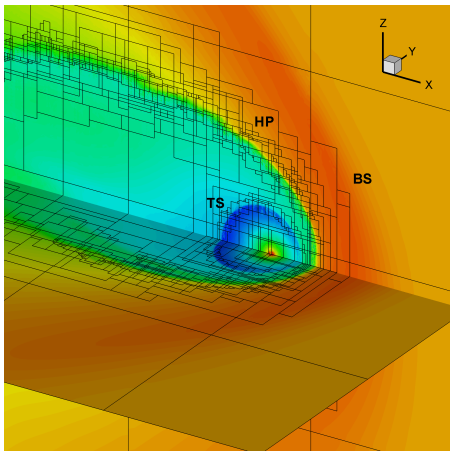
6.1 The Solar Wind Interaction with the Local Interstellar Medium

Simulations of the solar system, particularly the interaction of solar wind with the local interstellar medium, are an important application area of AMR simulations. For example, scientists are particularly interested in plasma quantities in the solar system because they are currently measured by the Voyager 1 and Voyager 2 spacecraft. The global modeling of the heliosphere requires a computational region to be about 1000 AU, whereas length of plasma fluctuations can be 0.01AU, which are too fine to be modeled without AMR. The other application of the AMR is putting a very fine mesh at the spacecraft locations to compare the observations with the theoretical models.

The Voyager program was conducted to investigate distant planets of the solar system. The primary mission ended in 1989 and this was the beginning of the *Voyager Interstellar Mission*, which has the purpose to investigate outer solar system environment and search for the heliopause boundary. The magnetic field plays an important role in investigating our solar system. The solar flares produce strong magnetic field disturbances in the interplanetary magnetic field (IMF). The solar system is surrounded by local interstellar medium (LISM), which is weakly ionized plasma. The interstellar magnetic field (ISMF) has strong influence on heliosphere shape. The Voyager probes have a big set of instruments, however only magnetometers are still operational on both spacecraft. Therefore, modeling and visualization of the magnetic field are extremely important for this problem.



(a)



(b)

Figure 9 (a) The heliosphere. SW: Solar Wind, TS: Termination Shock, HP: Heliopause, BS: Bow Shock, LISM: Local Interstellar Medium. (b) The structure of the AMR mesh.

The heliosphere is the region of space filled by the expanding solar corona, a vast region indeed extending $150 - 180$ astronomic units (AU) in the direction of the Sun's motion through the local interstellar medium (LISM) and several thousand AU in the opposite direction. The solar wind (SW) is a hydrogen plasma, expanding more-or-less radially from the Sun at speeds of $400 - 800$ km/s, with a density that decays as r^{-2} (r denoting radial

heliocentric distance from the Sun). The interstellar plasma, on the contrary, is rather weakly ionized. The Sun moves through the LISM with the velocity of about $25 - 26$ km/s. Collision of the SW and LISM flows creates a tangential discontinuity (the heliopause (HP), see Figure 9a) that separates these flows, a termination shock (TS), which decelerates supersonic SW at the inner side of the HP creating a so-called inner heliosheath, and possibly a LISM bow shock (BS).

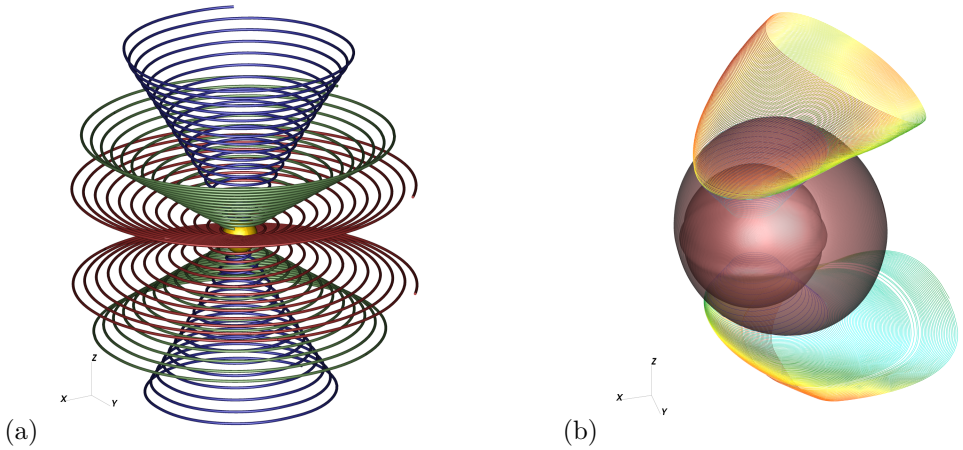


Figure 10 Magnetic field of the Sun. (a) Parker spiral shaped field lines. (b) Field lines passed the termination shock, colored by the field magnitude.

The AMR mesh used for the solar system simulation consists of five refinement levels and 20040 domains. The root grid on the coarsest level contains $100 \times 100 \times 80$ cells. In Figure 9b the structure of the AMR mesh around the heliopause is illustrated. Figure 10 shows the interplanetary magnetic field, which has the shape of the Parker spiral [30]. This distribution of the IMF is due to the expansion of the Sun's dipole magnetic field with SW velocity. As a result of the SW compression at the TS, the distance between two consecutive coils of the spiral becomes 3 times smaller in the inner heliosheath than in the supersonic SW. The streamlines behind the termination shock are shown in Figure 10b.

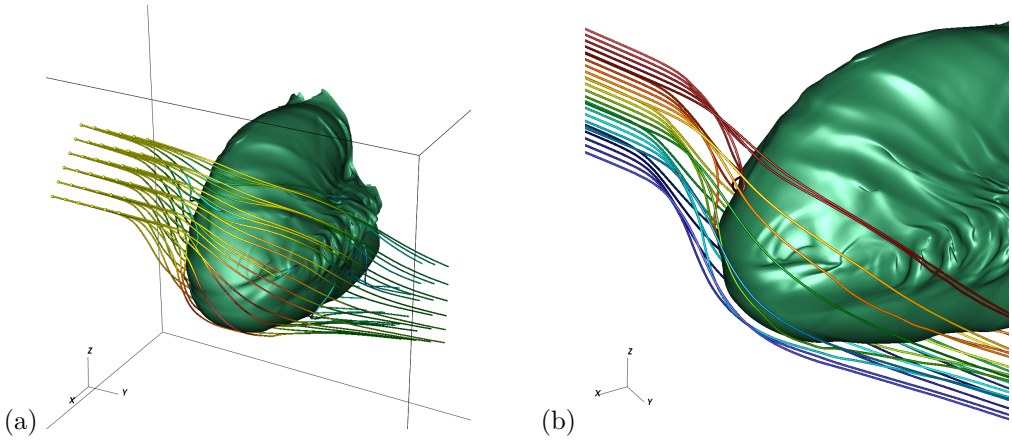


Figure 11 Interstellar magnetic field lines draping the heliopause. (a) Colored by the field magnitude. (b) Single color for each streamline).

Figure 11 shows interstellar magnetic field stream lines wrapping the heliopause, which is shown as the isosurface. The magnetic field squeezes the heliopause in the direction parallel to direction of the interstellar magnetic field at infinity. The wavy structure on the flanks of the heliopause is the combination of the Kevin-Helmholtz and Raleigh-Taylor instabilities (see [5]), which arises from the charge exchange between neutral hydrogen and plasma. The magnetic field acts as the stabilizing force, suppressing the instability, however, if magnetic field becomes perpendicular to the velocity of plasma, this stabilizing force becomes very small. This is the reason why the instability happens in the specific parts of the heliopause.

6.2 Simulation of Two Incompressible Viscous Vortex Cores

A second example comes from a 3D simulation of two incompressible viscous vortex rings merging, taken from [25]. Two co-rotating vortex rings approach each other at an angle, merging as they meet into a single vortex ring. The domain is periodic in the $z-$direction, with no-shear solid wall boundaries in the $x-$ and $y-$directions. This problem has been studied extensively due to its interesting vortex dynamics, both numerically and experimentally [19]. As the vortices merge, a complicated flow field is generated, providing a good testbed for the streamline algorithm. Streamlines provide a very useful visualization tool for interpreting the complex flow patterns produced as the flow evolves. Figure 12 clearly shows that the structure of the flow can not be captured without considering the AMR refined levels.

Figure 13 shows streamlines for the AMR data at the initial time (when there are two distinct vortex rings), after 60 timesteps (when the vortex rings are merging), and after 120 timesteps (after the two vortex rings have merged into a single flow structure). There are two levels of refinement in this simulation, each representing a factor of 4 refinement of the solution. The root grid on the coarsest level consists of 64^3 cells. In each figure, one can see tightly-wound streamlines around the vortex ring cores, along with streamlines farther from the vortex ring cores, which illustrate the far-field flowfield induced by the vortex rings.

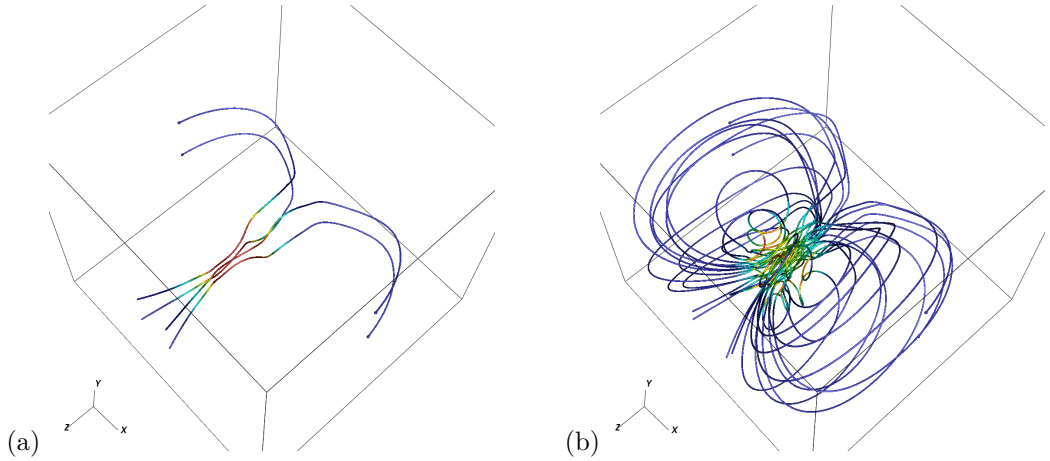


Figure 12 Streamlines in the vortex core data set (a) not considering the refinement levels, (b) using our algorithm. The color represents the velocity field magnitude.

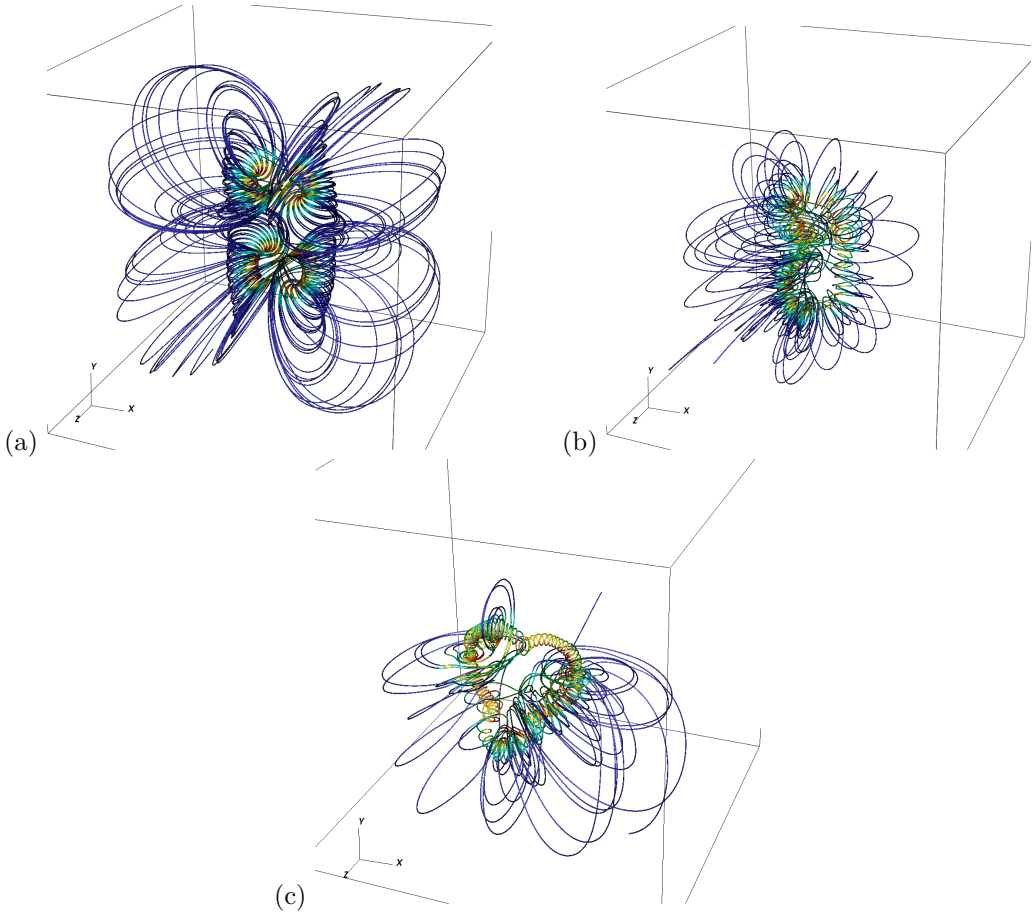


Figure 13 Streamlines visualization of the vortex core simulation for the time steps 0 (a), 60 (b), 120 (c), colored by field magnitude.

7 Conclusions and Future Work

In our paper, we have examined the problem of the computation of integral curves in AMR simulations. We proposed a method for numerical integration which utilizes the data information provided at finest refinement level available and also handles interpolation issues occurring at level boundaries. We applied our algorithm to visualization of streamlines in a model of the solar magnetic field as well as to a simulation of two vortex cores merging. Currently, our method is limited to block-structured rectilinear AMR grids. AMR methods have been expanded to simulate flow fields in complex geometries based on Cartesian grids with embedded boundaries or mapped grids. In future work we want to adapt our algorithm to handle these cases as well, which will make it possible to consider several additional interesting application areas, such as fusion. We further plan to implement a parallel version of our algorithm.

8 Acknowledgments

This work was supported in part by the Director, Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract Nos. DE-

FC02-06ER25780 (University of California, Davis) and DE-AC02-05CH11231 (Lawrence Berkeley National Laboratory) through the Scientific Discovery through Advanced Computing (SciDAC) program's Visualization and Analytics Center for Enabling Technologies (VACET). We thank our colleagues from VACET, the members of the LBNL Applied Numerical Algorithms Group, and the VisIt development team.

References

1 A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, and M. Welcome. A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations. *Journal of Computational Physics*, 142(1):1–46, May 1998.

2 A. Benkenida, J. Bohbot, and J. C. Jouhaud. Patched grid and adaptive mesh refinement strategies for the calculation of the transport of vortices. *International journal for numerical methods in fluids*, 40(7):855–873, 2002.

3 M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53:484–512, March 1984.

4 Marsha Berger and Phillip Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:64–84, May 1989. Lawrence Livermore National Laboratory, Technical Report No. UCRL-97196.

5 S.N. Borovikov, N.V. Pogorelov, G.P. Zank, and I.A. Kryukov. Consequences of the heliopause instability caused by charge exchange. *Astrophysics Journal*, 682:1404–1415, 2008.

6 Greg L. Bryan. Fluids in the universe: Adaptive mesh refinement in cosmology. *Computing in Science and Engineering*, 1(2), 1999.

7 M. Calvo, J.I. Montijano, and L. Randez. On the solution of discontinuous IVP's by adaptive Runge-Kutta codes. *Numerical Algorithms*, (33):163–182, 2003.

8 Hank Childs and Mark Miller. Beyond meat grinders: An analysis framework addressing the scale and complexity of large data sets. In *SpringSim High Performance Computing Symposium (HPC 2006)*, pages 181–186, 2006.

9 P. Colella and L. F. Henderson. The von Neumann paradox for the diffraction of weak shock waves. *Journal of Fluid Mechanics*, 213:71–94, 1990.

10 D. C. Fang, Weber, G. H., H.R. Childs, E.S. Brugger, B. Hamann, and K.I Joy. Extracting geometrically continuous isosurfaces from adaptive mesh refinement data. In *Proceedings of 2004 Hawaii International Conference on Computer Sciences (DVD-ROM conference proceedings)*, pages 216–224, 2004.

11 Christoph Garth, Han Krishnan, Xavier Tricoche, Tom Tricoche, and Kenneth I. Joy. Generation of accurate integral surfaces in time-dependent vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1404–1411, 2008.

12 L.F. Henderson, P. Colella, and E.G. Puckett. On the refraction of shock waves at a slow-fast gas interface. *Journal of Fluid Mechanics*, 224:1–28, 1991.

13 L. H. Howell, R. B. Pember, P. Colella, J. P. Jessee, and W. A. Fiveland. A conservative adaptive-mesh algorithm for unsteady, combined-mode heat transfer using the discrete ordinates method. *Numerical Heat Transfer, Part B: Fundamentals*, 35:407–430, 1999.

14 J. P. Jessee, W. A. Fiveland, L. H. Howell, P. Colella, and R. B. Pember. An adaptive mesh refinement algorithm for the radiative transport equation. *Journal of Computational Physics*, 139(2):380–398, January 1998.

15 Ralf Kähler, Donna Cox, Robert Patterson, Stuart Levy, Hans-Christian Hege, and Tom Abel. Rendering the first star in the universe – a case study. In *IEEE Visualization 2002*, pages 537–540. IEEE Computer Society, 2002.

16 Ralf Kähler and Hans-Christian Hege. Texture-based volume rendering of adaptive mesh refinement data. *The Visual Computer*, 18(8):481–492, 2002. Zuse Institut Technical Report ZR-01-30.

17 Ralf Kähler, Steffen Prohaska, Andrei Hutanu, and Hans-Christian Hege. Visualization of time-dependent remote adaptive mesh refinement data. In *IEEE Visualization 2005*, pages 175–182. IEEE Computer Society, 2005.

18 Ralf Kähler, John Wise, Tom Abel, and Hans-Christian Hege. GPU-assisted raycasting for cosmological adaptive mesh refinement simulations. In *Proceedings of Volume Graphics*. Eurographics Association, 2006.

19 Egon Krause. On vortex loops and filaments: three examples of numerical predictions of flows containing vortices. *Naturwissenschaften*, (90):4–26, 2003.

20 Oliver Kreylos, Gunther H. Weber, E. Wes Bethel, John M. Shalf, Bernd Hamann, and Kenneth I. Joy. Remote interactive direct volume rendering of AMR data. Technical Report LBNL 49954, Lawrence Berkeley National Laboratory, 2002.

21 R.S Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F.H. Post, and D. Weiskopf. The state of the art in flow visualization, part 1: Direct, texture-based, and geometric techniques. *Computer Graphics Forum*, 23(2):203–221, 2004.

22 Z. Li. An adaptive two-dimensional mesh refinement method for the problems in fluid engineering. *Lecture Notes in Computer Science*, 3314:118–123, 2004.

23 Kwan-Liu Ma. Parallel rendering of 3D AMR data on the SGI/Cray T3E. In *Proceedings of Frontiers '99 the Seventh Symposium on the Frontiers of Massively Parallel Computation*, pages 138–145. IEEE Computer Society, 1999.

24 R. Mannshardt. One-step methods of any order for ordinary differential equations with discontinuous right-hand sides. *Numerische Mathematik*, (31):131–152, 1978.

25 D. Martin, P. Colella, and D. T. Graves. A cell-centered adaptive projection method for the incompressible Navier-Stokes equations in three dimensions. *Journal of Computational Physics*, 227:1863–1886, 2008.

26 Nelson L. Max. Sorting for polyhedron compositing. In *Focus on Scientific Visualization*, pages 259–268. Springer-Verlag, 1993.

27 T. May, S. Schneider, M. Schmidt, and V. Luckas. Fast scalar and vector field visualization using a new progressive grid class. In *High Performance Computing Conference (HPC)*, pages 115–120, 2003.

28 Michael L. Norman, John M. Shalf, Stuart Levy, and Greg Daues. Diving deep: Data management and visualization strategies for adaptive mesh refinement simulations. *Computing in Science and Engineering*, 1(4):36–47, 1999.

29 Sanghun Park, Chandrajit Bajaj, and Vinay Siddavanahalli. Case study: Interactive rendering of adaptive mesh refinement data. In *Proceedings of the IEEE Visualization 2002*, pages 521–524. IEEE Computer Society, 2002.

30 E.N. Parker. The stellar wind regions. *Astrophysics Journal*, 134:20–27, 1961.

31 R. B. Pember, L. H. Howell, J. B. Bell, P. Colella, W. Y. Crutchfield, W. A. Fiveland, and J. P. Jessee. An adaptive projection method for unsteady, low mach number combustion. *Combustion Science and Technology*, 140:123–168, 1998.

32 P. J. Prince and J. R. Dormand. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980.

33 P. J. Prince and J. R. Dormand. High order embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 7(1), 1981.

34 D. Stalling. *Fast Texture-Based Algorithms for Vector Field Visualization*. PhD thesis, Freie Universität Berlin, 1998.

35 M. C. Thompson and J. H. Ferziger. An adaptive multigrid technique for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 82(1):94–121, May 1989.

36 X. Tricoche and C. Garth. Topological methods for visualizing vortical flows. In *Mathematical Foundations of Visualization, Computer Graphics, and Massive Data Exploration*. 2006 (to appear).

37 Gunther H. Weber, Hans Hagen, Bernd Hamann, Kenneth I. Joy, Terry J. Ligocki, Kwan-Liu Ma, and John M. Shalf. Visualization of adaptive mesh refinement data. In *Proceedings of the SPIE (Visual Data Exploration and Analysis VIII)*, volume 4302, pages 121–132, 2001.

38 Gunther H. Weber, Oliver Kreylos, Terry J. Ligocki, John M. Shalf, Hans Hagen, Bernd Hamann, and Kenneth I. Joy. Extraction of crack-free isosurfaces from adaptive mesh refinement data. In *Proceedings of the Joint EUROGRAPHICS and IEEE TCVG Symposium on Visualization, Ascona, Switzerland, May 28–31, 2001*, pages 25–34, 335, 2001.

39 Gunther H. Weber, Oliver Kreylos, Terry J. Ligocki, John M. Shalf, Hans Hagen, Bernd Hamann, and Kenneth I. Joy. Extraction of crack-free isosurfaces from adaptive mesh refinement data. In *Hierarchical and Geometrical Methods in Scientific Visualization*, pages 19–40. Springer Verlag, 2003.

40 Gunther H. Weber, Oliver Kreylos, Terry J. Ligocki, John M. Shalf, Hans Hagen, Bernd Hamann, Kenneth I. Joy, and Kwan-Liu Ma. High-quality volume rendering of adaptive mesh refinement data. In *Vision, Modeling, and Visualization 2001*, pages 121–128, 522. Akademische Verlagsgesellschaft Aka GmbH and IOS Press BV, 2001.

41 Gunther H. Weber, Martin Öhler, Oliver Kreylos, John M. Shalf, E. Wes Bethel, Bernd Hamann, and Gerik Scheuermann. Parallel cell projection rendering of adaptive mesh refinement data. In *Proceedings of the IEEE Symposium on Parallel and Large-Data Visualization and Graphics 2003*, pages 51–60. IEEE Computer Society, 2003.

Disclaimer