Interactive View-Dependent Extraction of Large Isosurfaces

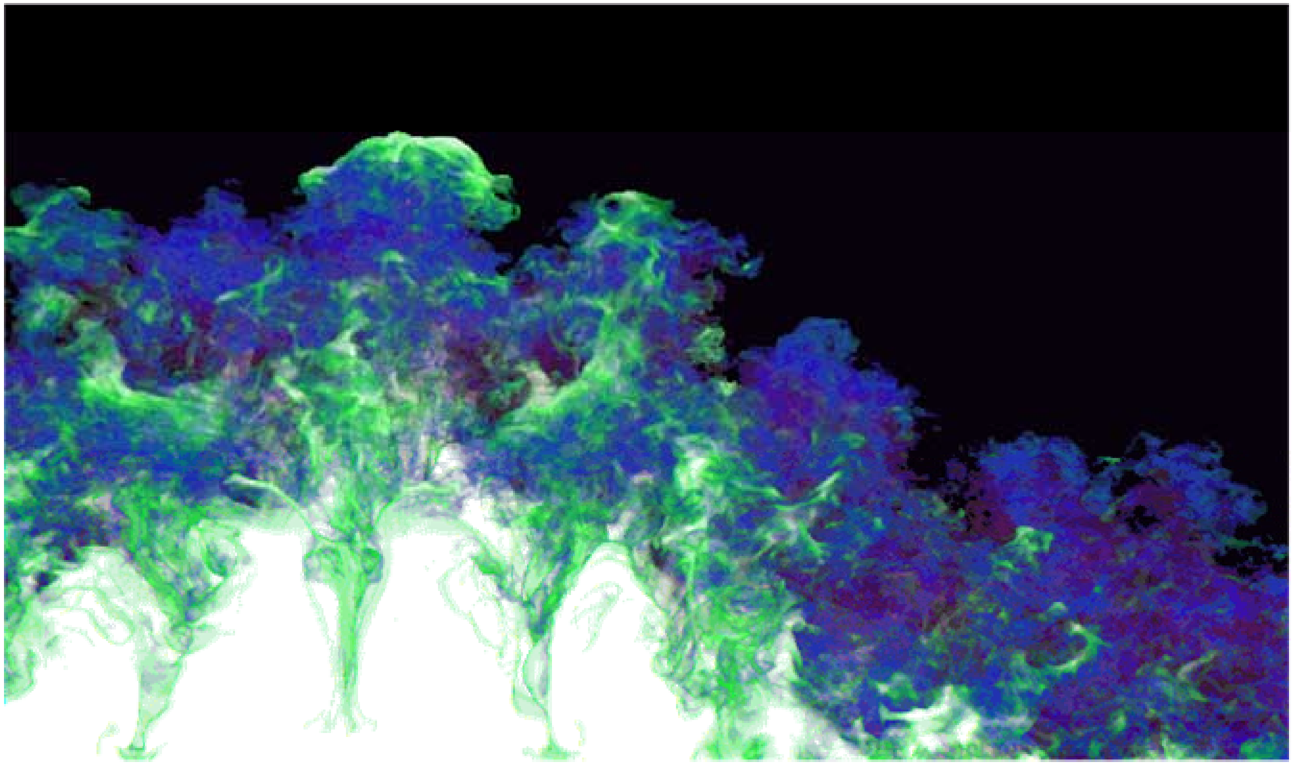
Benjamin Gregorski[*†]
Mark Duchaineau[*], Peter Lindstrom[*], Valerio Pascucci[*]
Kenneth I. Joy[†]
[*]Lawrence Livermore National Lab
[†] Center for Image Processing and Integrated Computing, UC Davis

Outline

- The Problem
- Related Work
- Our Approach
- Longest edge bisection
- System Overview
- Implementation Details
  - Precomputed data, memory layout
- Changing isovalues
- Conclusions and Future Work


ASCI White
ASCI Blue-Pacific
ASCI Blue Mountain
ASCI Red
NPACI Blue Horizon
ASCI Cplant

Simulation of Richtmyer-Meshkov instability



Winner of 1999 Gordon Bell Award for Performance Dec. 6, 1999 mirin12

Motivation

- Rapidly increasing dataset size
  - Simulations with 2k x 2k x 2k grids and 273 timesteps
- Still need interactive visualization
  - Point, isosurface, and volume rendering
- Process only what is necessary
  - Reuse computations

Related Work

- Recursive Tetrahedral Meshes
  - Bottom up merging for simplification Zhou et al (Vis 1997)
  - Top down refinement with topology preservation Gerstner and Pajarola (Vis 2000)
- Error based refinement
  - ROAM (Duchaineau et al. 1997)
    - Dual queue data structure
  - Exploits frame-to-frame coherence
- Out-of-core isosurfacing
  - Volumes/isosurfaces do not fit in main memory
  - External Memory Interval Tree Silva et al (Vis 1997, 1998)

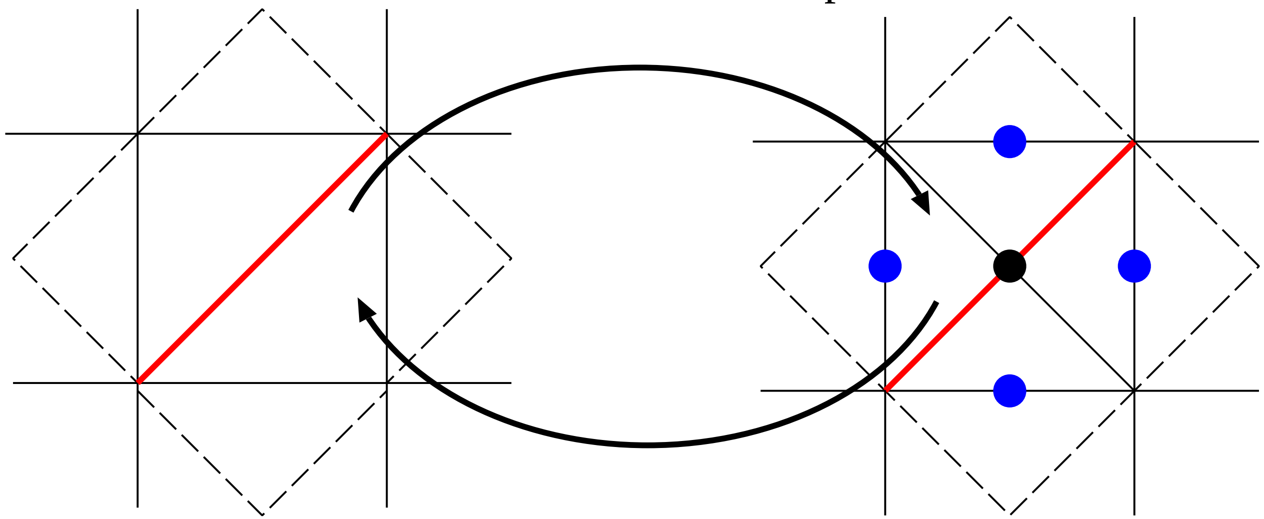
Our Approach

- Recursive Tetrahedral Meshes
  - Combine top down and bottom up refinement
  - Adaptively refinable mesh
- Error based refinement
  - Dual split/merge queue algorithm of ROAM
  - Exploit frame-to-frame coherence
    - Minimal updates
    - Cached triangles
- Out-of-core isosurfacing
  - Precomputed errors, gradients, and min/max values
  - Hierarchical ordering follows mesh refinement
  - Memory map data file at runtime

Phase 0 Split/Merge

Splitting a Phase 0 Diamond

Gives 4 Phase 1 tets and child diamonds
Child split vertices are shown in blue



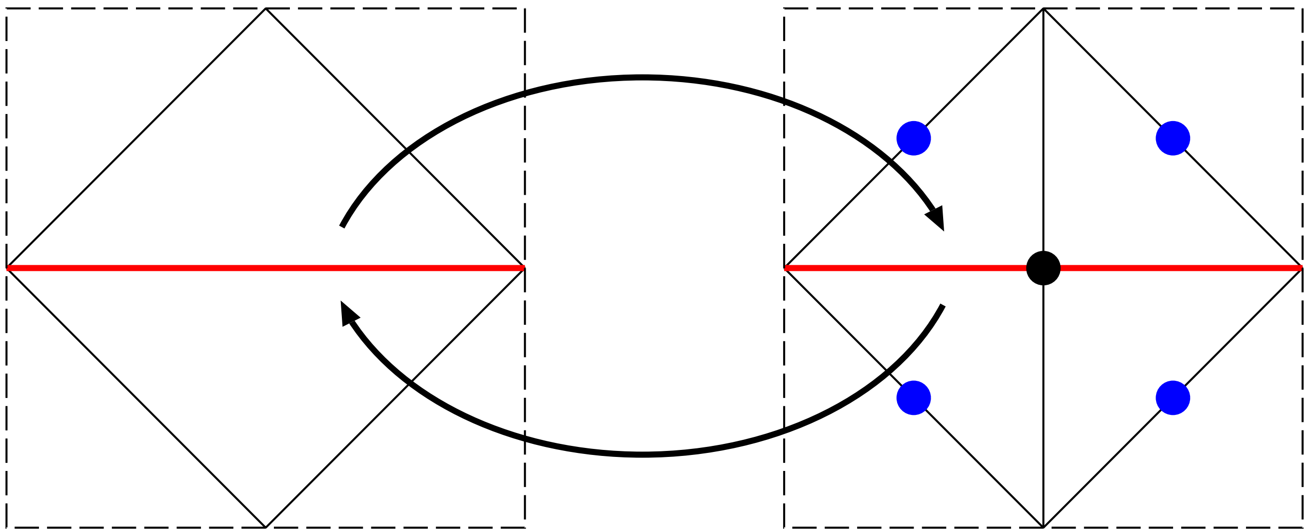
Removes 4 Phase 1 tets from child diamonds

Merging a Phase 0 diamond

Phase 1 Split/Merge

Splitting a Phase 1 Diamond

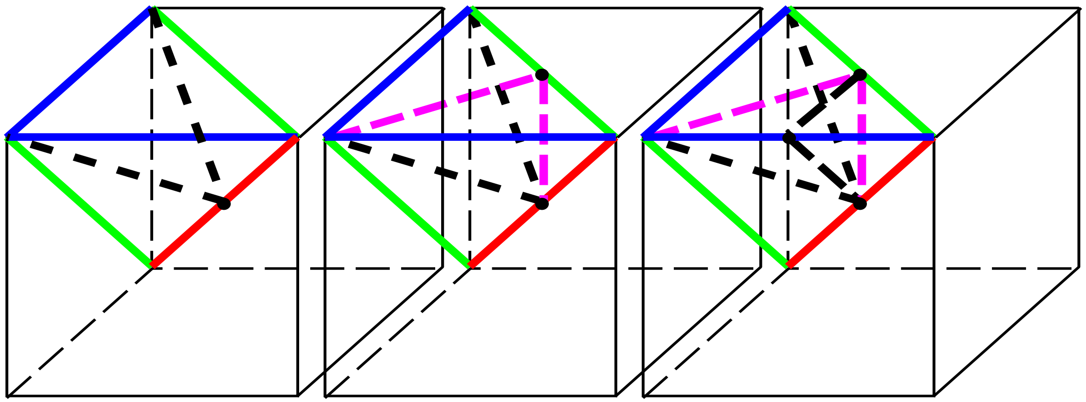
Gives 4 Phase 0 tets and child diamonds
Child split vertices are shown in blue



Removes 4 Phase 0 tets from child diamonds

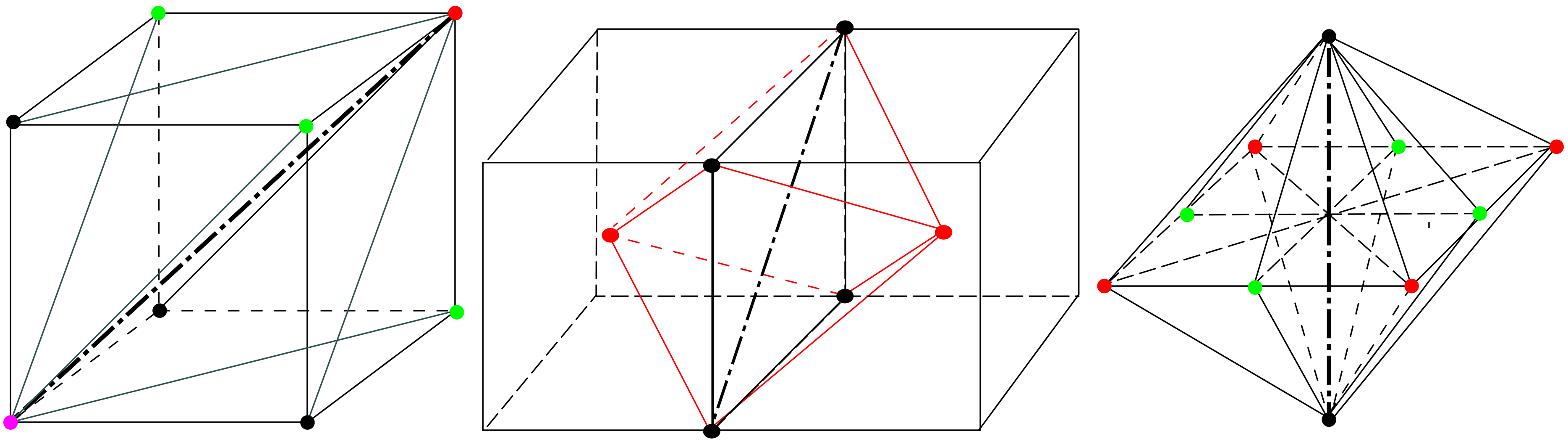
Merging a Phase 1 diamond

3D Refinement



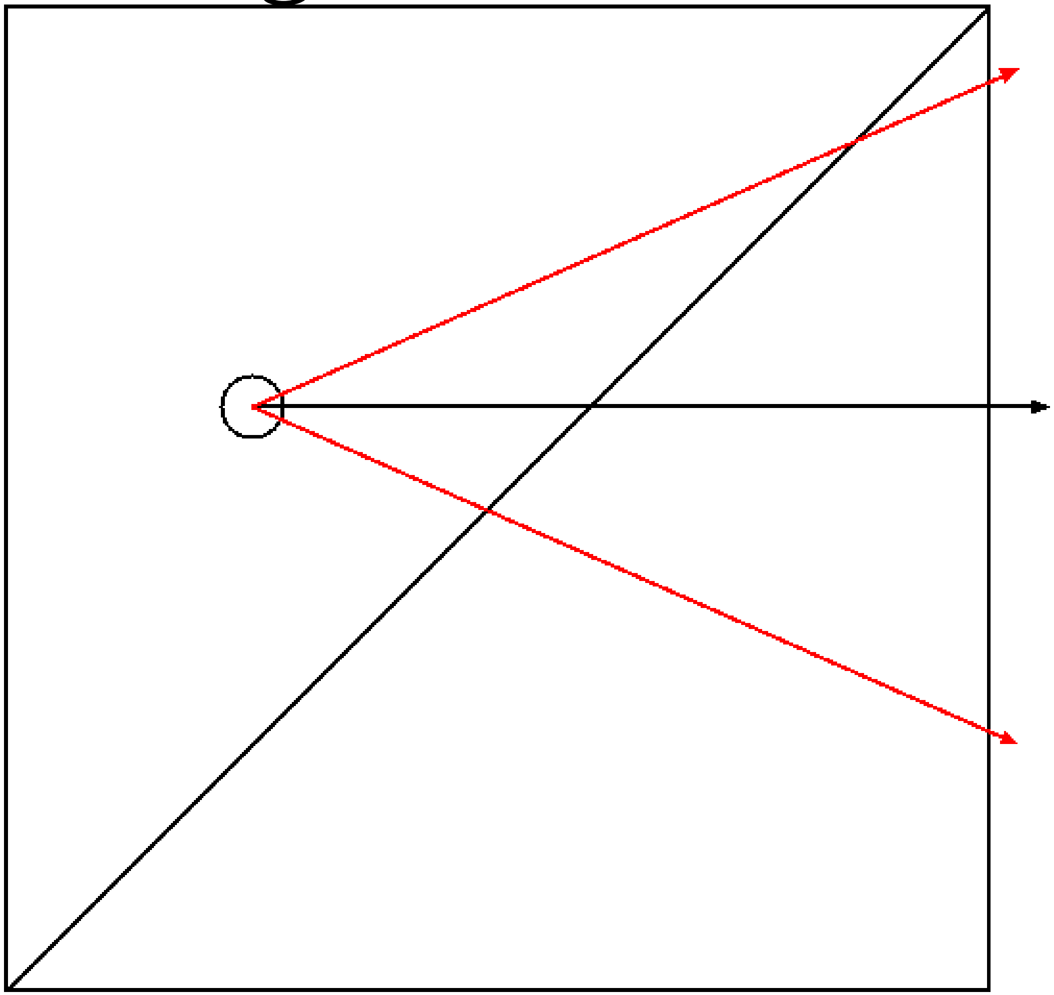
- Three phases of refinement
- Split cells, then faces, then edges of an octree
- Three refinements equivalent to one octree subdivision

Summary of 3D Diamond Shapes

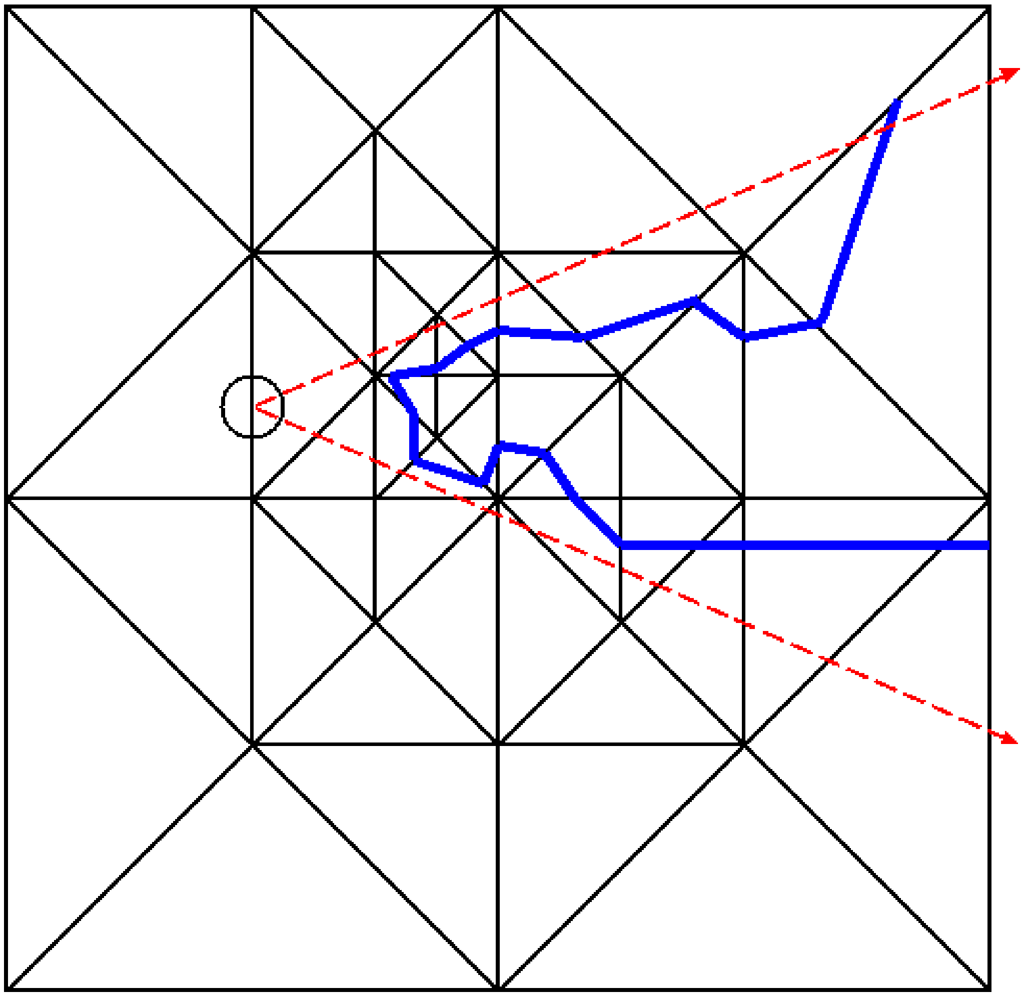


- Phase 0: 6 tetrahedra, 3 parents, 6 children
- Phase 1: 4 tetrahedra, 2 parents, 4 children
- Phase 2: 8 tetrahedra, 4 parents, 8 children

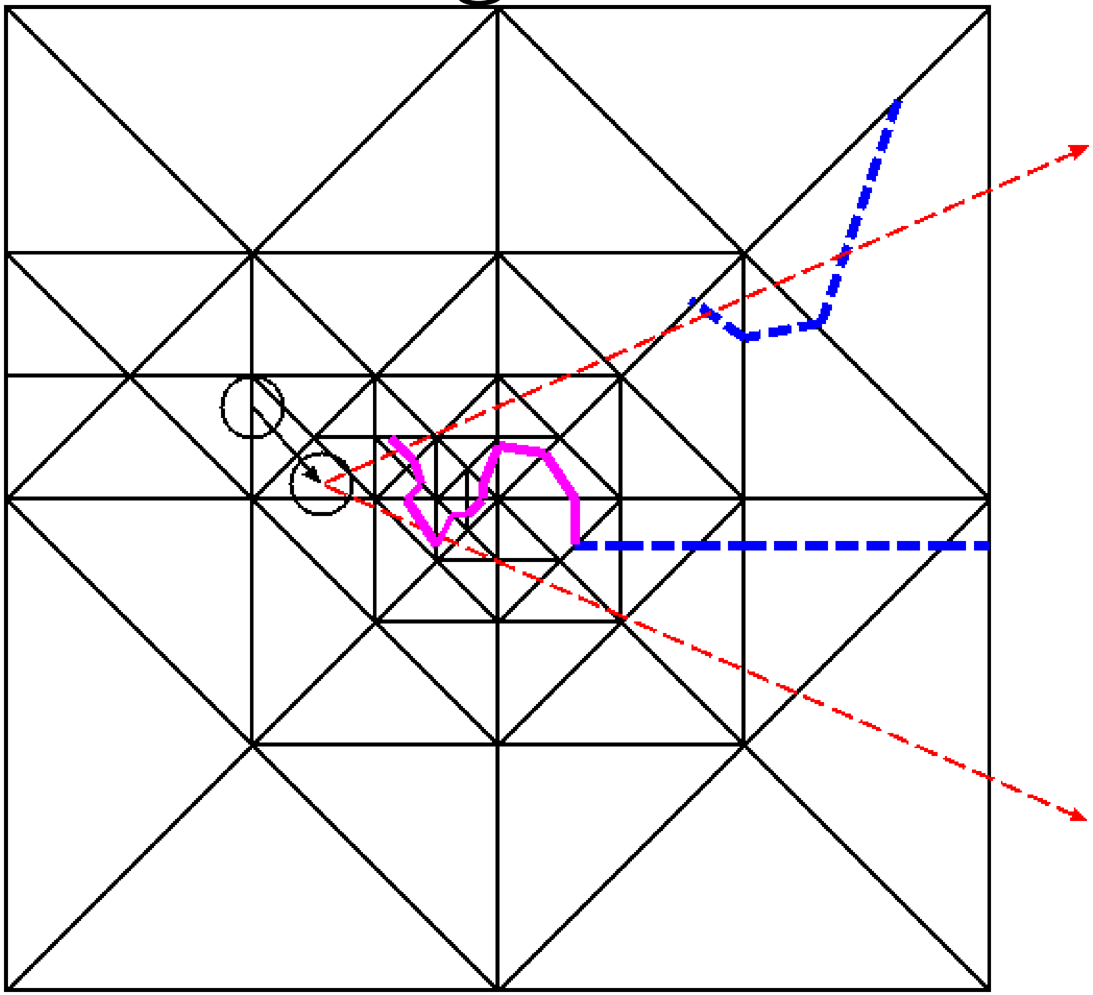
Initial Configuration



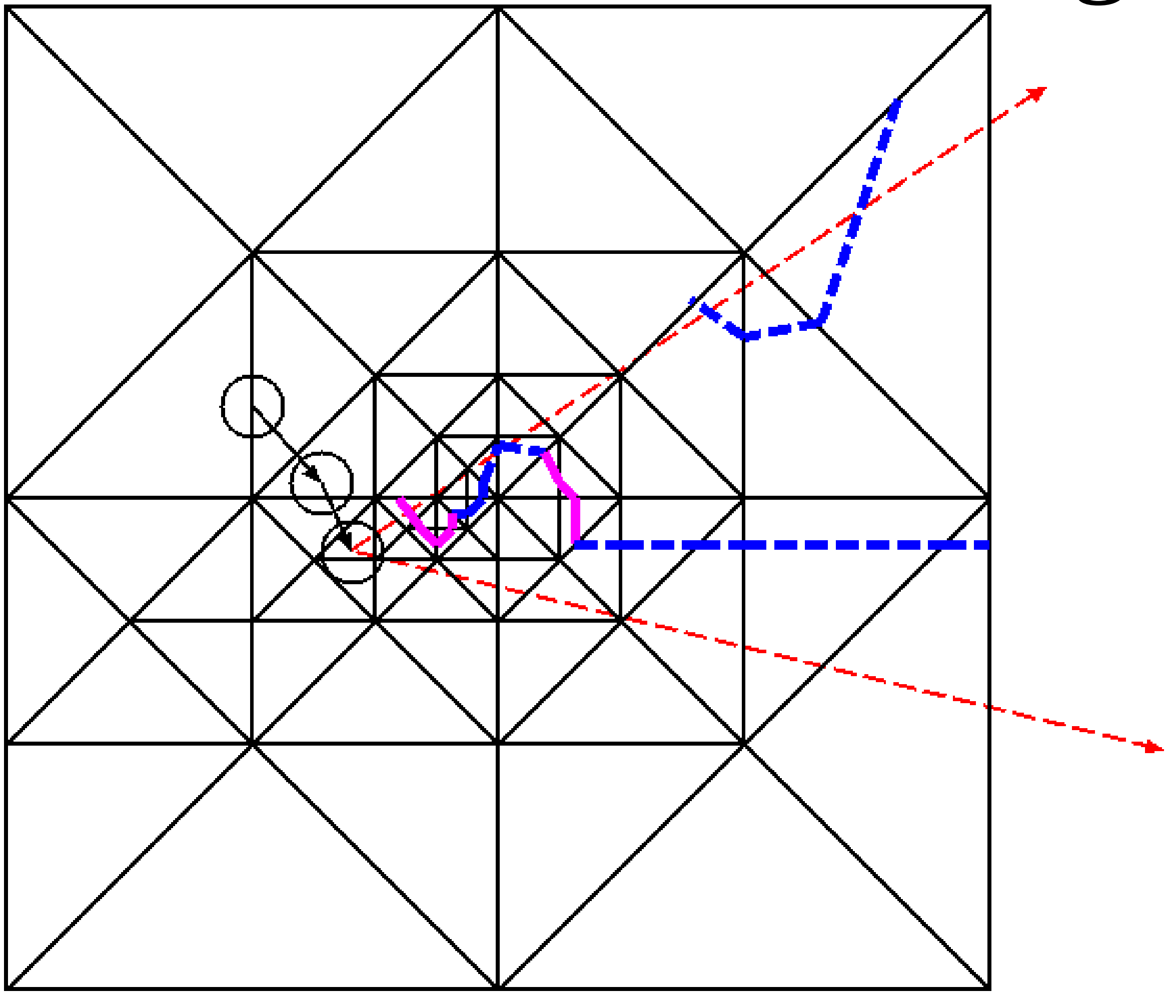
Initial Refinement



Viewpoint Change



Viewpoint and Frustum Change



Overall Algorithm

- Given an error ***E***
- Error recomputation
  - Recompute for splitable and mergable diamonds
  - *Invisible* and *empty* diamonds have an error of 0
- Mesh Refinement
  - Split diamonds with **error > E,** merge diamonds with **error < E**
- Stopping Criteria
  - Error tolerance, triangle count, processing time for current frame
- Extract isosurface from new leaf tetrahedra

Buckyball Walkthrough

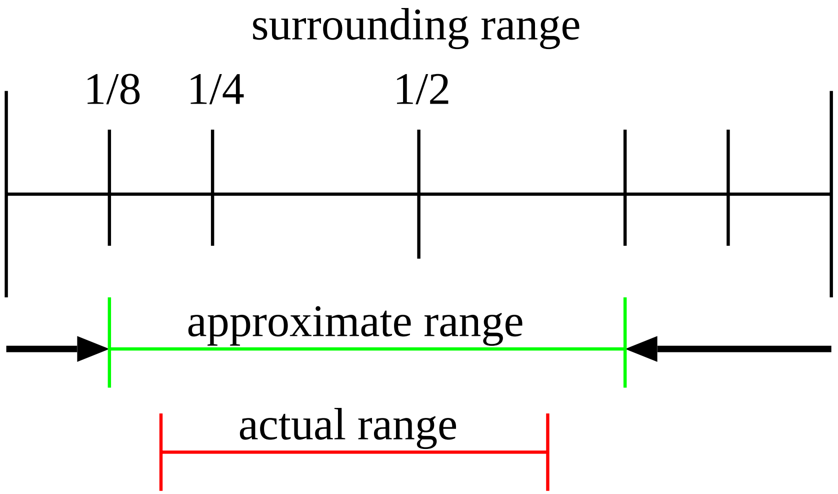
- Pseudo buckyball dataset
  - Made with gaussian functions
  - 256^3 tiled to form a 512^3 dataset
  - Gradient based shading

Precomputed Data Values

Assumptions
- Dataset is $2^n \times 2^n \times 2^n$
- Periodic boundary conditions
  - For 129^3 0 and 128 are the same

Per diamond information (32 bits)
- Data values (8 bits)
- Gradient value (14 bits)
- Error Value (6 bits)
- Min/Max Values (4 bits)

Error encoding
- Log scale per octree level



Min/Max encoding
- Relative to surrounding diamond S
- 2 bits encode offset from min/max of S

Memory Layout Scheme

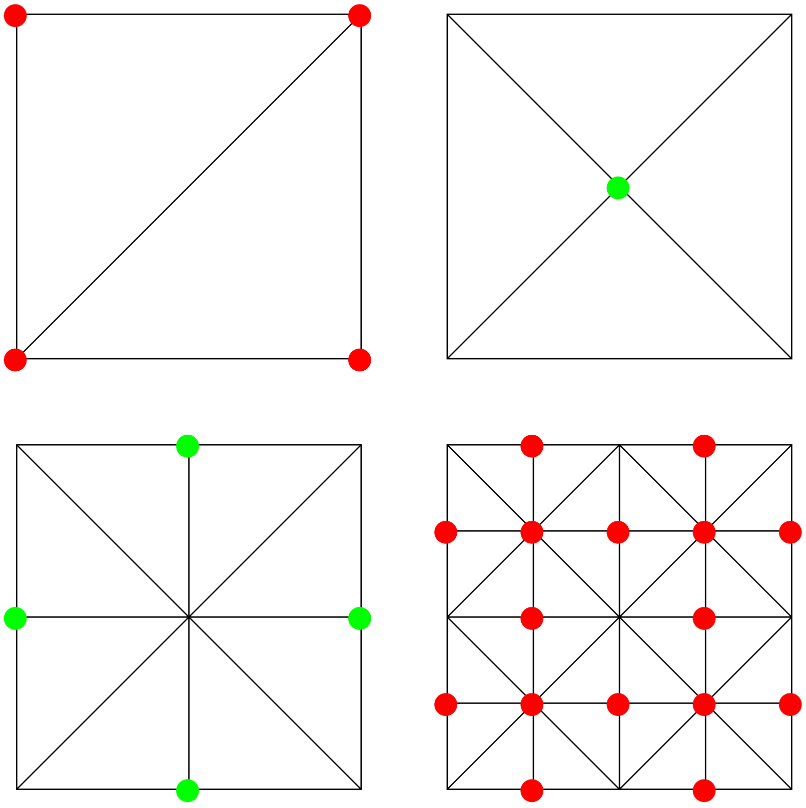
Data layout follows mesh refinement

Hierarchical z-order, Lindstrom and Pascucci (Vis 2001)

- Breadth first data layout on disk
- Fast conversion between (i,j,k) index and z-order index

Easy, scalable out-of-core paging

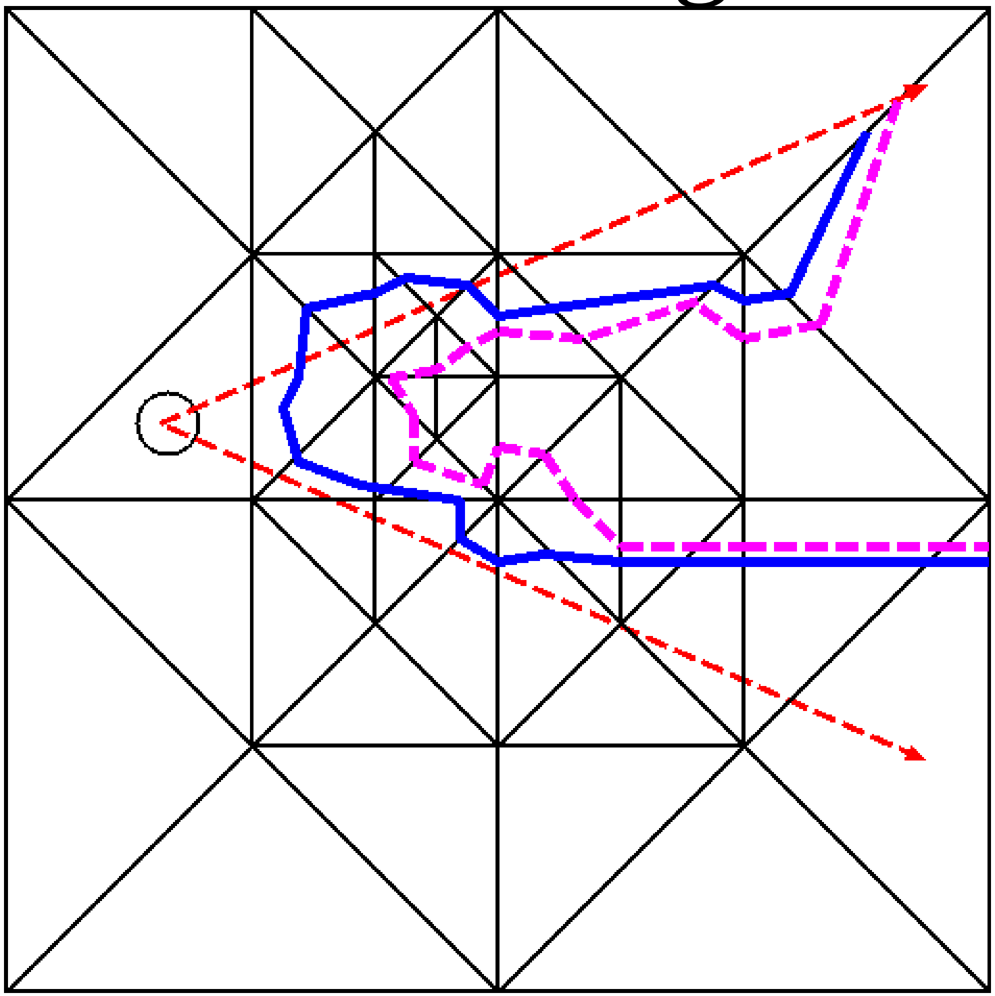
- memory mapped file



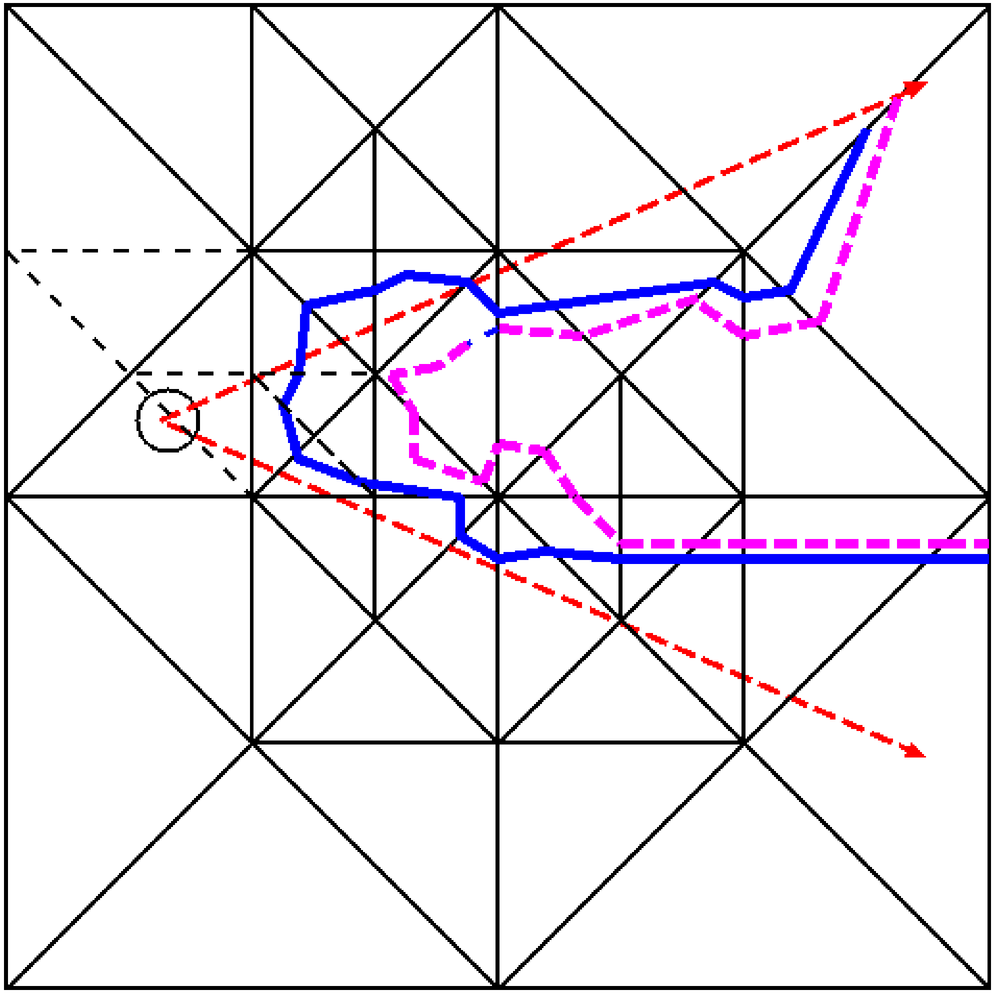
Changing the Isovalue

- Start from current mesh
  - Remove old triangulation
  - Split diamonds where new isosurface exists
  - Merge diamonds where isosurface does not exist
- Start from base mesh
  - Remove old triangulation
  - Clear all data structures
  - Split diamonds where isosurface exists

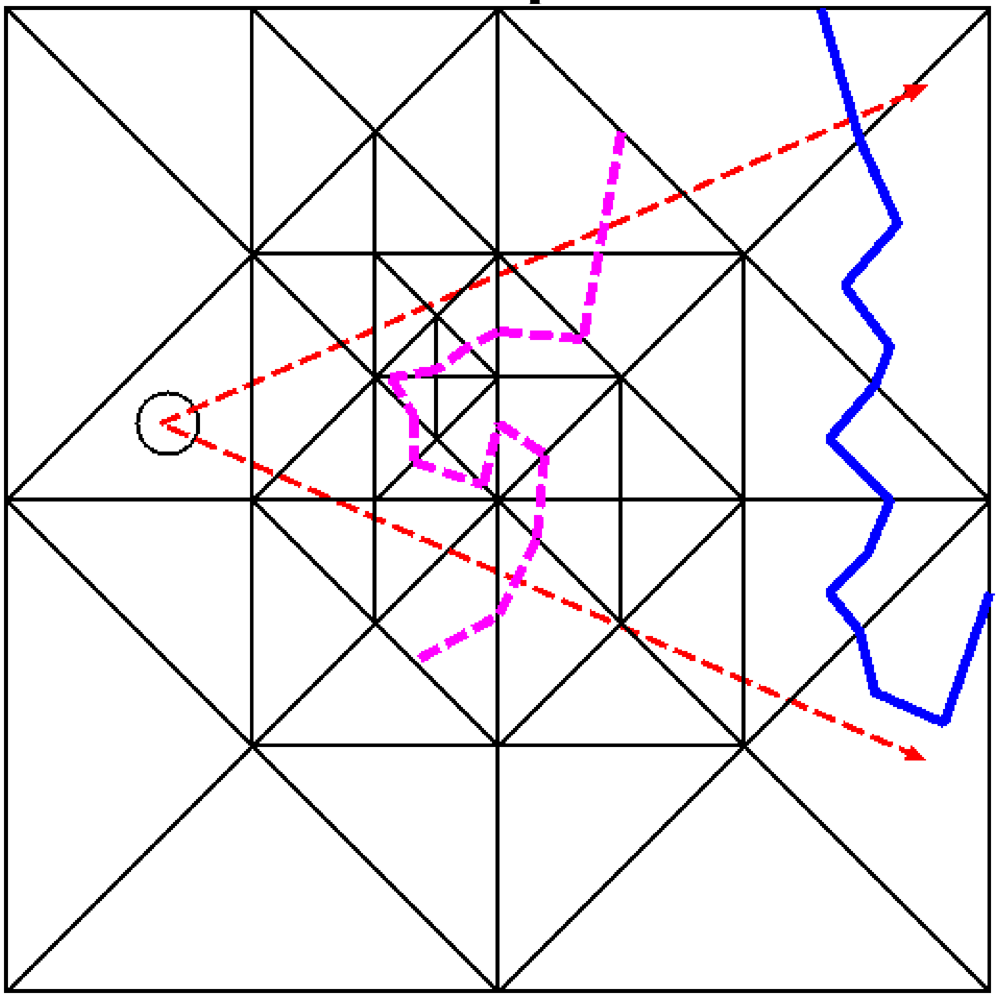
Contours are close together



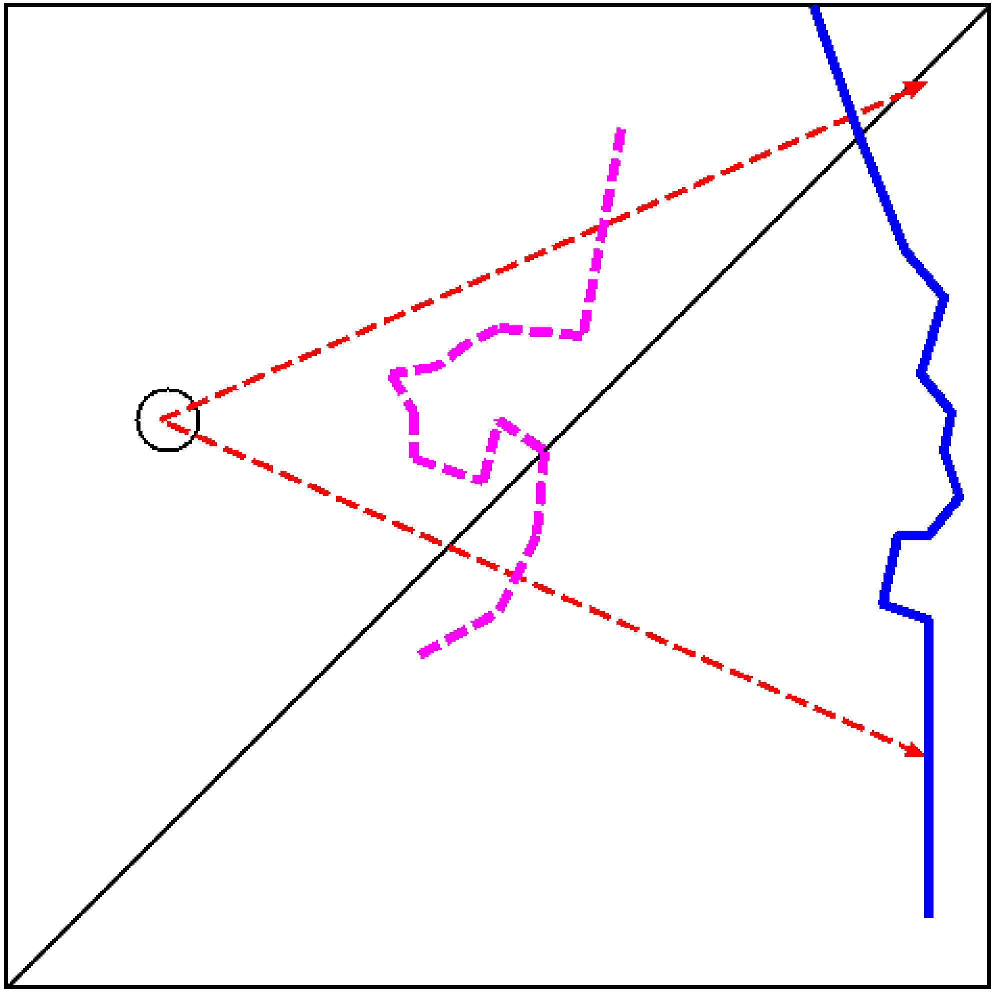
Refine around new isosurface



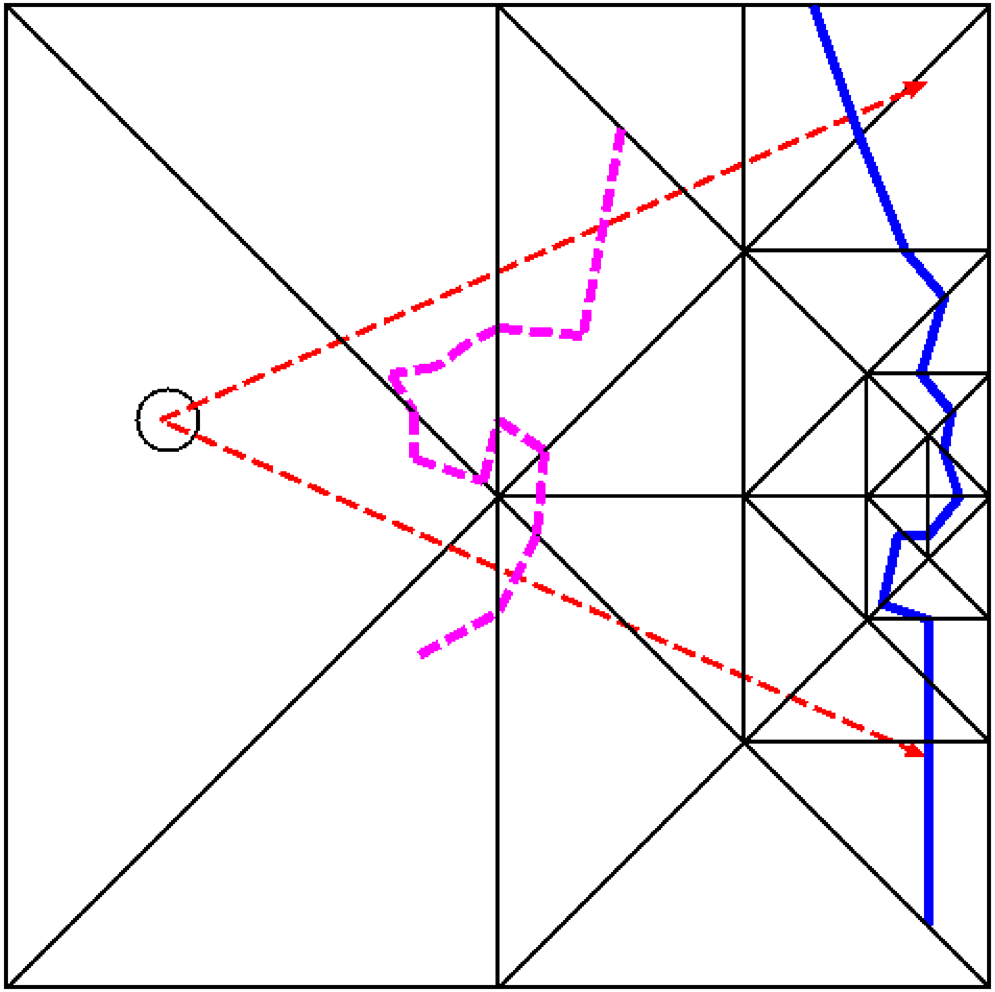
Contours are far apart



Start from base mesh



Refine around new isosurface



Buckyball Movie

Fixed viewpoint and error value

Changing isovalues

Performance

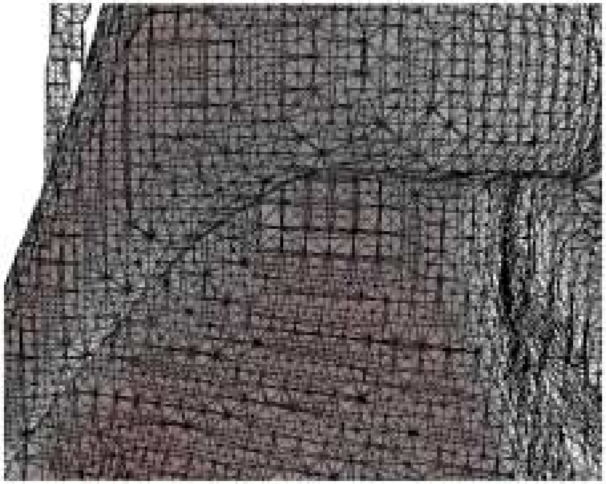
- Test datasets
  - 512^3 chunks (138MB), Preprocessed data (552MB)
- Performance (per second)
  - GNU/Linux: 1.5 GHZ Pentium, 512 Mb, GeForce 2 GTS
  - Frustum culling and Error computation: $1e^6$ – $2.2e^6$
  - Splitting and Merging: 20K – 40K
  - Rendering (immediate mode): 550K – 650K triangles

Richtmyer-Meshkov Movie

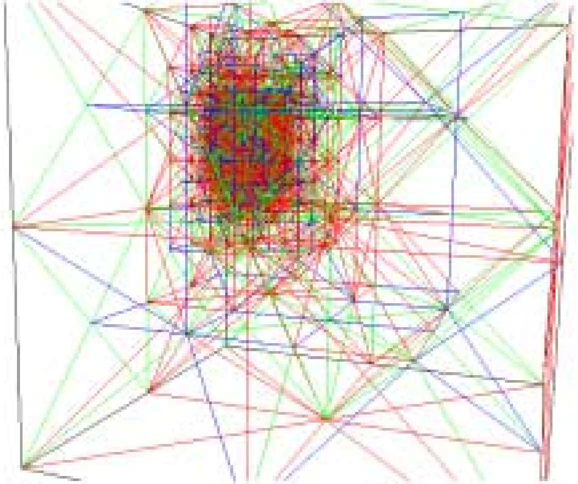
- Gordon Bell prize winner 1999
- Simulation of mixing gases in a shocktube experiment
  - 2048x2048x2048 x 273 timesteps
- 512^3 chunk of original dataset at timestep 273

Adaptive Mesh Example

Shaded surface



Triangle mesh



Adaptive mesh

Conclusions and Future Work

- Interactive, view-dependent, out-of-core isosurface extraction
  - Fast, crack-free refinement and coarsening using diamonds
  - Minimal number of mesh updates per frame
  - Exploits frame-to-frame coherence
  - Data ordering for scalable out-of-core isosurfacing

- Time varying data
- Rendering Techniques
  - Point, volume, programmable shading for isosurfaces
- Higher Order Field Representations
  - Cliffs and discontinuities (similar to material interfaces)
    - Extended Marching Cubes (Kobbelt et al.), Hermite interpolation (Ju et al.)

Acknowledgements

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. We also thank the people at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California Davis for all of their help and suggestions with this work.

Oliver Kreylos for buckyball datasets