CSC 7700: Scientific Computing

Module D: Scientific Visualization
Dr. Werner Benger

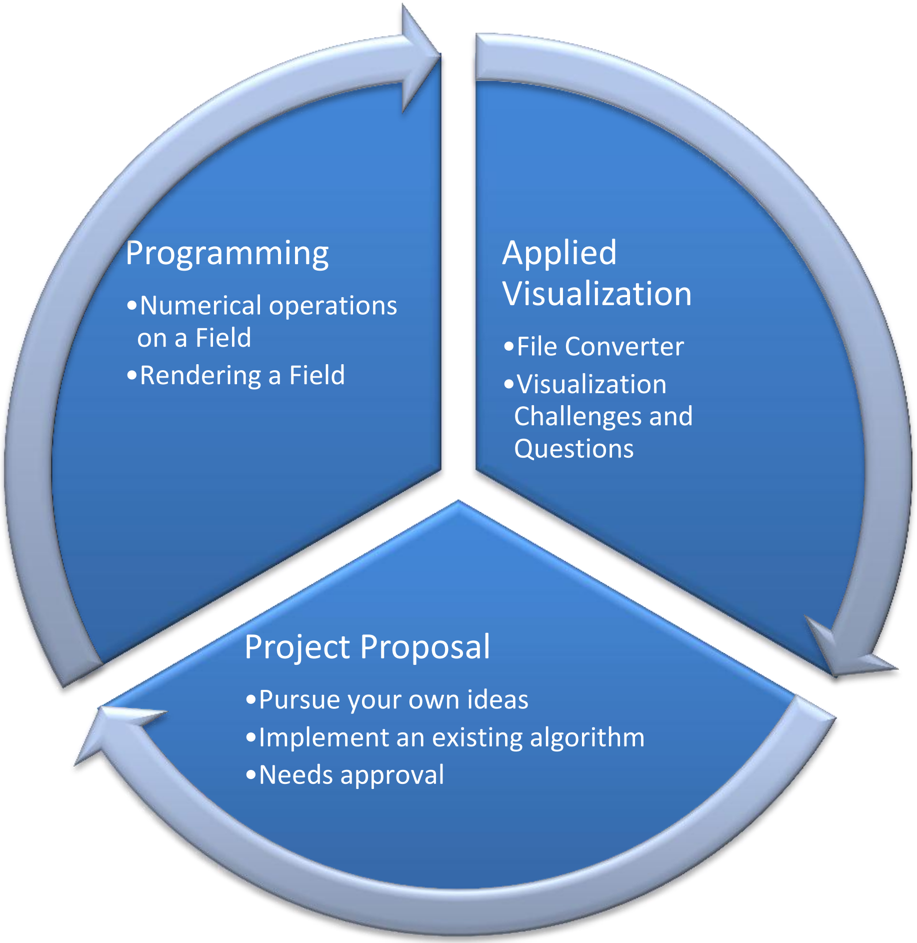
Final Project Description
Due Nov 18th

Project Options

The final project is one out of three options:

- *Programming*: requires C++ skills and consists of implementing two Vish objects (one numerical, one rendering)
- *Applied Visualization*: only minimal coding requirements, outcome needs to be an animation sequence
- *Project Proposal:* requires major C++ skills and may include opengl and gpu shader programming – most effort, most flexibility

Choose one of them. Part of the work is to include a report documenting the found challenges and how they were solved. Exceptional originality and quality will be required for "A" grade.



Programming

- Numerical operations on a Field
- Rendering a Field

Applied Visualization

- File Converter
- Visualization Challenges and Questions

Project Proposal

- Pursue your own ideas
- Implement an existing algorithm
- Needs approval

Applied Visualization

(minimal to no coding required)

Provide a visualization of an existing dataset from a cactus simulation run, dataset will be provided by Dr. Schnetter & Dr. Loeffler

Need to convert data from Cactus-specific HDF5 into F5 format

- Source code for data converters can be found in fish/fiber/F5apps/converters/CactusCode/
- Various converters available; will need CarpetInterpToF5 . Some adjustments of converter code might be requirement.

Dataset will involve several scalar (density, internal energy, log(lapse)) and at least one vector field (velocity).

- Talk with Dr. Loeffler about the physical meaning of these quantities if unclear and what features are to be expected
- Explore the visualization options, for instance rendering as slices, isosurfaces, volume; some vector field visualization modules (arrows, streamlines) exist; some experimental modules might require some work to be applicable to this data set (improvements of visualization modules are welcome, but not required)
- Investigate combined visualizations of all available fields versus separate visualization of each field by its own

Provide an animation sequence

- might want to include a camera path – expand camera path functionality as needed.
- Final result should come as .mpeg, .wmv or .avi file with title and credits. For instance, windows moviemaker provides easy movie-making facilities.

Programming, Part I

Implement a Vish object that performs an operation from Geometric Algebra on a Field:

- Unary operators: use a single Field as input
  - Example: Convert a Field of vectors to bivectors (or back) using the “*”-operator
- Binary operators: use two Fields as inputs
  - Example: compute wedge product of two vector fields, resulting in a bivector Field; compute a rotor field from a scalar and a bivector field; apply a rotor field to a vector field.
- Hint: Coding details on how to output a Field object are provided in the documentation file fish/lakeview/tutorial/FishCoding.pdf
- The provided Vish object does not need to provide all details described in the tutorial pdf, basic functionality is sufficient.
- Use the Particles.f5 dataset as test; it contains scalar and (tangential) vector fields.
- Collaboration on project work is highly encouraged, but everyone should implement another operator, no operator should be implemented twice. Ideally each contribution is complementing others.
- Most important: ask questions as they arise!

Programming, Part II

Implement a Vish object that implements a rendering operation of a Field (chose one):

- Display a vector field via arrows at each point
- Display a bi-vector field via small rectangles at each point

Hints:

- The lecture example was demonstrating how to get an arbitrary field. The *coordinates* of point can always be retrieved implicitly by code like this (insert in render() or update() function, use “Field” object as described on page 46 of last lecture):

```
FieldSelector FS = getFieldSelector(Context);
RefPtr<Grid> G = FS.getGrid();
RefPtr<Field> Coordinates = G->getCartesianPositions();
```

- Via OpenGL 1: use glBegin()/glEnd() and draw small line segments or rectangles at each location
- Via OpenGL 2+: use vertex buffers, computing the geometry of all line segments or rectangles at first in an array of coordinates and send them all into one vertex buffer for rendering

Project Proposal

Come up with your own ideas! Possible routes are:

- Visualize a dataset from an application scientist that you are working with
- Loading matlab-generated HDF5 data in Vish for processing
- Investigate performance and limitations of volume rendering
- Implement/expand a Vish object, for instance a visualization algorithm or rendering method found elsewhere such as:
  - Line-integral convolution / LIC http://www.zhanpingliu.org/research/flowvis/LIC/LIC.htm
  - Pre-integrated volume rendering (http://www.vis.uni-stuttgart.de/~engel/pre-integrated/)
  - Depth peeling (http://developer.nvidia.com/object/Interactive_Order_Transparency.html)
  - Application of an Nvidea demo for scientific visualization (porting, applying to a dataset): http://developer.download.nvidia.com/SDK/10.5/opengl/samples.html

Proposed projects need to be similar in effort to the other two project suggestions

Approval of project proposals will be required!

Office hours and communication

- Email:
  - werner@cct.lsu.edu for specific questions
  - sci-comp-instructors@cct.lsu.edu for general questions
- Room 211 Johnston Hall
- Office hours: Tue/Thu 2pm-3pm (or anytime with prior arrangement)
- AIM Messenger ID: aeiwerner
- Skype ID: cctwerner
- Note that I will be away and out of email/chat contact the 1st week of November (Oct 30th to Nov 5th), and unpredictable availability during Nov 15th to 18th (SC10 conference)