3D Visualization for Matlab

Erich Birngruber, René Donner, Georg Langs

Computational Image Analysis and Radiology Lab
Medical University of Vienna, Austria
http://www.cir.meduniwien.ac.at/

CSAIL, Massachusetts Institute of Technology, USA
Motivation

Previous work

• Gobbi et al. 2008: Simulink Libraries for Visual Programming of VTK and ITK

Need

• Work on (large) medical datasets
• Need for “visual debugging” / big picture
• Easy usability for the Matlab user
Motivation

Matlab

- Rapid prototyping / algorithm development
- Limited visualization for 3D data
Motivation

Matlab

- Rapid prototyping / algorithm development
- Limited visualization for 3D data

VTK

- Powerful, good performance
- Platform independent
Outline

- Development foundations
- Resulting Framework
- Use cases
Foundations

- Mex interface allows access to C/C++ libs
  - Use Matlab memory management
  - Mex entry point function
    
    ```c
    void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])
    {
      ...
    }
    ```

- VTK “pipes & filters” concatenate data processing
Obstacles

• Keeping state:
  no direct support for handles in Mex

• VTK rendering loop interferes with Matlab
matVTK Framework

- Handle based:
  implicit creation, automatic destruction
- Imitates Matlab plot commands:
  prefixed with `vtk`

`vtkplotpoints(pointset)`
`vtkquiver(points, vectors)`
Parameters

- No code generation: explicit mapping
- Easily extendable
- Basic settings mapping:
Parameters

• No code generation: explicit mapping
• Easily extendable
• Basic settings mapping:

VTK
  scalar value
  triplet value
  different modes
  (Constants)
Parameters

- No code generation: explicit mapping
- Easily extendable
- Basic settings mapping:

VTK
- scalar value
- triplet value
- different modes
  (Constants)
Parameters

- No code generation: explicit mapping
- Easily extendable
- Basic settings mapping:

**VTK**
- scalar value
- triplet value
- different modes (Constants)

**Matlab**
- 5
- [1, 2, 3]
- “String”
matVTK Framework

Matlab

Input Data → Function calls → Handle → Numeric Output

VTK

Handle

Algorithms
triangulation
pointselection

Graphic primitives
points
lines
surfaces
volumes

Visual output
Primitives

Example from work of E. Dittrich
Primitives

• Points

Example from work of E. Dittrich
Primitives

- Points
- Lines

Example from work of E. Dittrich
Primitives

- Points
- Lines
- Volumes
- Meshes (surfaces)

Example from work of E. Dittrich
Primitives

- Points
- Lines
- Volumes
- Meshes (surfaces)
- Vectors (as arrows)

Example from work of E. Dittrich
Primitives

- Points
- Lines
- Volumes
- Meshes (surfaces)
- Vectors (as arrows)
- Tensors

Example from work of E. Dittrich
Combining Primitives

- Combinations of primitives can construct complex scenes

Example from work of E. Schwartz
Additional Functionality

- Widgets for data inspection & annotation
  - Cropping
  - Labels
  - Axes
- Data export
- Restore scenes
% plot and save components to handle
hv = vtkplotvolume(volume, 'SoftSkin')
hm = vtkplotmesh(vertices, faces 'color', [0 0 1])
% crop volume and mesh with plane widget
vtkcrop(hv, 'plane')
vtkcrop(hm)
% show axes
vtkgrid
% interactive display, free resources
vtkshow
vtkdestroy
Example

% plot and save components to handle
hv = vtkplotvolume(volume, 'SoftSkin')
hm = vtkplotmesh(vertices, faces, 'color', [0 0 1])
% crop volume and mesh with plane widget
vtkcrop(hv, 'plane')
vtkcrop(hm)
% show axes
vtkgrid
% interactive display, free resources
vtkshow
vtkdestroy
Performance

Isosurfacing for different volume sizes

Dienstag, 29. September 2009
Availability

- Developed on MacOS X 10.5
- Linux 32bit, 64bit
- Windows XP, Vista, 7

*as platform independent as Matlab and VTK*
Outlook

- Streaming visualization
- Saving widget states
- Remove scene objects
- Event recording / movies

Example from work of G. Langs
Thanks

• Available online:

http://www.cir.meduniwien.ac.at/matvtk/

• Feedback and ideas welcome

erich.birngruber@meduniwien.ac.at