VisNow – a Modular, Extensible Visual Analysis Platform

Krzysztof S. Nowinski
University of Warsaw, ICM
Prosta 69
00-838 Warsaw, Poland
know@icm.edu.pl

Bartosz Borucki
University of Warsaw, ICM
Prosta 69
00-838 Warsaw, Poland
babor@icm.edu.pl

ABSTRACT

A new, dataflow driven, modular visual data analysis platform with extensive data processing and visualization capabilities is presented. VisNow is written in Java, easily extendable to incorporate new modules and module libraries. Dataflow networks built with the help of interactive network editor can be wrapped into stand-alone application for the end users.

Keywords

Scientific visualization, modular systems, dataflow driven system, medical imaging

1. INTRODUCTION

The VisNow system is based on longtime experience of usage and development of AVS systems (Advanced Visual Systems Inc., AVS 3-5 and AVS Express), IBM Data Explorer and several other general visualization systems [Peik07] [Hans04] [Para10][HPV13]. VisNow implements the concept of dataflow driven, modular system. The user builds a network of modules reading, processing and mapping data. Generic data structures are passed from output ports to input ports and the data processing is controlled by the user-manipulated parameters.

2. MODULAR STRUCTURE

All functionality of VisNow, including data input/output, logical and numerical processing (a.k.a. filtering), mapping of numerical data into geometric object and rendering these objects, are implemented as modules. Each module is capable of processing of data according to current parameter setting and outputting the results. In the case of input modules processing data means usually reading them from local disk or from some remote source. VisNow modules process two types of data: a **Field** encapsulating a discrete representation of data defined over an 1-, 2- or 3-dimensional area and a **GeometryObject** encapsulating a Java3D geometry. It should be noted that the distinction between filtering and mapping modules is somewhat blurred: filtering modules output simultaneously graphical rendering of the results to be shown in the viewer window and mappers usually output the results as data object. For example, the module interpolating data to a regular mesh outputs the graphic rendering of the resulting field and the isosurface module outputs both a geometry object and an irregular mesh with interpolated data. Thus, the GUI of a module consists usually from **Computation UI** controlling module parameters, e.g. axis and location of a slice or input file name, and **Presentation UI** controlling color mapping of data and details of the presentation of points, lines, surfaces and volumes.

2.1. VisNow Granularity

The majority of existing systems based on the dataflow paradigm implement a fine-grained concept of relatively small building blocks. This model requires building of complicated networks to perform even simple tasks. In particular SciRun, IBM DX and other systems, require to instantiate "technical" modules like a colormap manager, 3D scene etc. to make the first geometrical object visible.

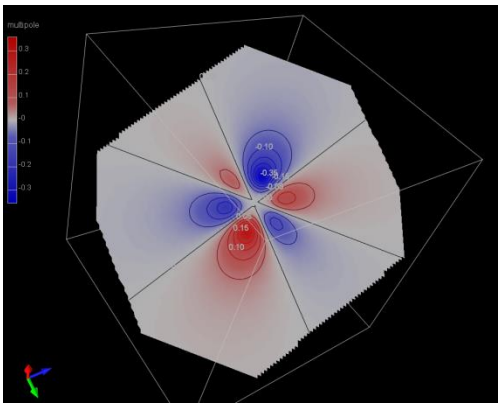

Figure 1. An example of VisNow visualization

In contrast, the VisNow system uses simple networks of high level modules: it is enough to use a reader, a slice module, an isolines module and the build-in viewer to obtain a reasonable visualization of a 3D data set – an application of read-in and see principle.

Figure 2. VisNow network producing Fig.1

VisNow provides fairly precise estimation of default parameter values. In particular, the mapping modules that could create extremely large geometries automatically downsize the input to reasonable dimensions leaving the final control over the geometry size to the user. As an example, glyph visualization of a vector vield over an 512x512x512 mesh will be automatically downsampled to 64x64x64 mesh.

To simplify the interaction of the user with the network and module controls network area, module controls and the viewer display are synchronized. The user can pick a geometry object in the 3D window highlighting in the network area the module that created this object and bringing up the module controls.

3. VISNOW DATA STRUCTURES

VisNow uses internally a single, universal, abstract **Field** data type describing a discretization of an area in the Euclidean 1-, 2- or 3-space together with a set of numeric and non-numeric data defined over this area. The Field data type has two implementations – a **Regular Field** type and an **Irregular Field** type. Basically, a field consists of three basic components – the obligatory **structure**, the (explicitly provided or implicitly defined) **geometry** and a (possibly empty) set of **values**.

3.1. Regular Field

The Regular Field type covers all data sets with a **regular structure**, that is, a one- two- or three-dimensional table structure that is characterized by a simple list of **dimensions** – e.g. a 100,000,000-long time series, an 1920x1080 HD image or a 512x512x300 CT scan. In the simplest case, the geometry is defined implicitly with the node p_{ijk} located at the point (i,j,k). In the general case of a regular **curvilinear field** all coordinates of all points can be provided explicitly, and in an intermediate setting the user can set **affine coordinates** consisting of the origin point p and one, two or three cell vectors v_0, v_1, v_2.

3.2. Irregular Field

The Irregular Field type requires explicit definitions of both structure and geometry. The structure is determined by the number of nodes and a list of **Cell Sets**. Each cell set is a collection of cells (point cells, segments, triangles, quadrangles, tetrahedral, pyramids, prisms and hexahedra). This data type covers basically all data sets occurring in FEM structural and CFD computations, but can be (and is) used for visualization of molecular structures, abstract graph presentation etc. Currently, VisNow does not support arbitrary polygons/polyhedral requiring off-line triangulation of such sets.

3.3.VisNow Data Values

VisNow uses a generic **Data Array** type to hold a variety of simple numeric (unsigned byte, short, int, float and double), complex, logical, string and generic Java object data. A data array holds the proper (flat) array of values of corresponding type and some additional data, e.g. name, physical unit, minimal and maximal values etc. Each data item can be a scalar, a vector or a (possibly symmetric) matrix. Thus, vector or tensor data can be processed natively with VisNow.

3.4. Time Dependent Data

The data values and node coordinates can be static or dynamic (defined for a list of **time moments**). It is sometimes important to have different data defined for different lists of time moments (e.g. numeric forecast outputs provide atmospheric pressure and ground level temperature for each five-minute time step while precipitation data are integrated over one hour intervals and orography or land cover are definitely static.

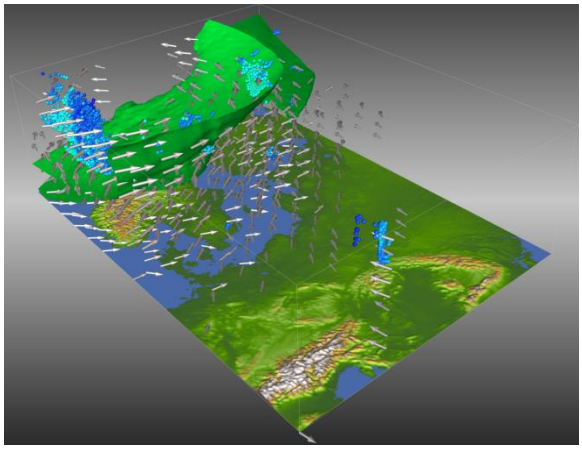

Figure 3. Numeric weather forecast data in the (x,y,t) coordinates: orography shown at the base, a pressure isosurface shows emergence of low pressure area,

wind and rain shown as glyphs

VisNow allows different lists of time moments for each variable (data array or node coordinates) providing a reasonable, piecewise linear interpolation for a given time moment. The data can be dynamically modified with time steps added or removed, and VisNow takes care for consistent interpolation. In the case of 2D fields, the time dependent data can be converted to a stack of slices of a 3D field – see Figure 3.

The only limitation of the VisNow time dependent data capabilities is the requirement of constant data structure (topology).

4. STANDARD VISNOW MODULES

4.1. Input/Output

VisNow supports a basic set of data formats with all types of images, simplified raw volume, AVS field and own VisNow regular field metafile format. The last file format is designed as a universal metafile allowing to read in ASCII or binary files containing static or time dependent coordinates and values of a regular field. In addition, a reader capable of browsing DICOM data is available for medical imaging data input, and an interactive trial-and-error method of ingestion of volumetric data of unknown dimensions is provided.

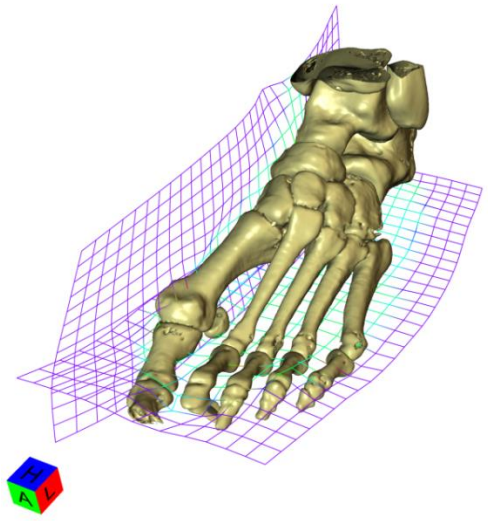

Figure 4. CT data read in from a DICOM file and deformed to match a patient image.

Unfortunately, the irregular data are so complicated that there is no way of developing a metafile that could describe all particular file formats. Thus, VisNow offers a set of modules capable of reading in AVS UCD, Ansys Fluent, EnSight and VTK data, usually with some limitations. Every partial result of VisNow data processing can be stored in the VisNow (ir)regular field format.

VisNow can access data either by a local disk path or from remote URL. In addition, VisNow supports browsing of UNICORE grid resources either with the use of a grid virtual filesystem browser or with a grid bean for inserting a GridFTP data transfer as part of an UNICORE workflow.

The content of the viewer window can be written in any of the Java supported image formats in arbitrary resolution (not limited to the screen resolution). In addition, an animation can be stored and converted to the MPEG-4 movie format.

4.2. Data Modification (Filtering)

VisNow provides standard operations (downsizing and cropping regular data, interpolation to user defined meshes, simple arithmetic on data etc.). Advanced numerical processing modules include differential operations on regular fields with arbitrary geometry, FastFourier transform, convolution with user defined and editable kernel etc. Data calculations can be performed by simple mathematical formulas.

Advanced image denoising, segmentation and skeletonization algorithms have been implemented as elaborated data filtering modules. These modules are mainly used in medical applications.

4.3. Mapping of Data to Geometries

Each type of VisNow data (IrregularField, 1D-, 2D- and 3D-RegularField) has a default visualization mode. VisNow provides elaborated methods of mapping component values to colors including easily modifiable continuous or quantized colormap.

In the case of more than one data component it is possible to map one component to hue and modifu the brightness or saturation by another component, e.g. mapping electrostatic potential to hue and field intensity to brightness. Blending of grayscale mapped anatomical data with physiological data mapped with a rainbow datamap is also a standard VisNow feature. In addition, the object transparency can be controlled by yet another selected data component.

The standard mappers library includes slicing, volume rendering and isosurfacing of 3D data, graphing and isolines creation in the case of 2D data and simple graphing of 1D data. Glyphs and text glyphs can be used for the representation of data at selected nodes. Streamlines, animated streamlines and object flow animations are available in the case of vector data components.

VisNow provides extensive capabilities of annotating the generated images. Colormap legends and coordinate axes are available with much attention paid to the clean and flexible labelling. The user can label field values by text glyphs, 3D annotations can

be located in arbitrary points of the 3D scene and 2D annotations (titles) can be displayed .

4.4. Viewers

VisNow provides a specialized 2D viewer and a simple graph viewer in addition to the main 3D viewer. A configurable 3D field viewer providing volume rendering and/or a set of orthogonal slice views helps to visualize 3D medical imaging data and an orthogonal viewer generates engineering type presentations.

The 3D viewer is the basic visual interaction node with the standard picking, geometric modification and (to the limited extent) object drawing.

5. IMPLEMENTATION AND EXTENDABILITY

VisNow is implemented in Java with Java3D as its graphic interface to GL and is structured as a set of NetBeans projects.

Each VisNow module is encapsulated in a Java package holding its main clas, an XML description file, and (usually) classes holding module parameters and a GUI panel. In addition a set of XML library descriptions allow to choose between basic, standard and enhanced module libraries.

5.1. Extending VisNow

New module libraries (plugins) can be created as separate NetBeans projects importing standard VisNow JAR files and following the package structure described above. New libraries can be dynamically added and modified at the runtime. Fast Java compilation opens an interesting possibility of usage of VisNow as a sort of an integrated development environment. The user can encapsulate newly implemented algorithm into the VisNow module and use sophisticated GUI for program parameters and visual debugging. As an example, the development of non-rigid 3D registration of CT data (see fig. 4) has been done entirely within the VisNow system with many algorithmic and implementational problems found and resolved under constant visual control.

Any module network created in VisNow can be easily converted to a Java main class and released as a stand-alone application without the network GUI. Such form of the application suits well the needs of an end user.

5.2. Java Specific Problems

Java, together with NetBeans provides excellent environment for the development of large, complicated projects. Nevertheless, some basic limitations should be taken into account or overcome. The object paradigm of the language with significant memory overhead and the Java3D data access forced to use flattened data arrays. For example, coordinates of a field of N nodes are processed as a single array of the length $3*N$. On the other hand, the current hardware developments and increasing data size displayed a very serious Java language limitation. Array indices and collection sizes are 32-bit integers as. In effect, all Java data structures are limited to the size of at most 2^{31} items.

We are currently in progress of overcoming this limitation by the use of a library of non-standard array-like data structures developed in ICM. The JLargeArrays library allows to declare and use arrays indexed by long integers and thus the computational capabilities are limited only by physical memory size. The basic data structures and several basic data reading, filtering and visualization modules are already converted to JLargeArrays and we expect the next release of VisNow at least partly capable of large data processing.

6. VisNow AVAILABILITY

VisNow is available under GPL Classpath Exception public license with the binary installers for Linux, Windows and Mac OS, accessible from http://visnow.icm.edu.pl. The sources of VisNow and JLargeArrays are available at the GitHub repository.

7. REFERENCES

[Peik07] Peikert R. Visualization systems. SciVis 2007, http://graphics.ethz.ch/teaching/former/scivis_07/Notes/Handouts/11-visSystems.pdf

[Hans04] The Visualization Handbook, ed. by C.D. Hansen, C.R.Johnson, Elsevier 2004

[Para10] ParaView guide, Kitware, Inc. Version 4 (August 2012)

[HPV13] High Performance Visualization, ed. by E. Wes Bethel et al., CRC Press 2013