Implementing Oceanographic Analyses Using ArcGIS Engine and Java

Tiffany C. Vance[1], Christopher W. Moore[2] and Nazila Merati[2]
[1]NOAA/National Marine Fisheries Service, Seattle, WA
[2]Joint Institute for the Study of the Atmosphere and Ocean, University of Washington, Seattle

Abstract

Geographic information systems provide a high level of functionality for spatial analyses but are not yet able to provide extended functionality needed to create a truly "scientific GIS". Functionality that is lacking include time series analyses, calculation of the volume of the overlap between two volumes or calculation of the intersection of a vector path with a volume. Other functions include the ability to specify a slice through a three dimensional lattice of model output data and to make various analyses along that slice. In this project, we use Java to integrate GIS functionality with Java3D (and Java-wrapped OpenGL) visualization capabilities. Specifically we are using a combination of Java/Java3D and ArcGIS Engine to create a scientific GIS. We combine the tools in the ArcGIS Engine/Java API with the capabilities of algorithms written in Java with the visualization capabilities of Java3D.

Introduction

The Java tools described here were designed using a framework approach that enables them to be integrated into an application (OceanGIS), or interfaced with ArcEngine. We aimed to develop these tools for a broad range of potential users. This project focuses on tools that convert non-spatial data into GIS compatible data, expedite the transfer of spatial data to coastal professionals and emergency managers, and enhance analyses used for disaster preparedness and response activities. As the tools we develop can be deployed without a full ArcGIS license, we hope to make them widely available to better integrate field activities during disaster responses.

Java can be used to program scientific calculations and analyses, but it isn't inherently spatial. Datasets can have a spatial component, but Java treats this as it would any type of coordinate system. Topology is not stored with data. Functions such as map projections, slope calculations and spatial intersections are not native to the language. However, Java *is* easily extensible and functions written in other languages can be integrated.

VRML provides the ability to visualize scientific data and to allow the user to interact with the data by rotating, zooming and panning, but you cannot easily query VRML objects. Ideally one would be able to point at a three dimensional location in a VRML view and return a reference to the object that lists the analytical methods that act on the data, and allows the user to execute these method. Recent developments in Java3D extend the functionality of VRML and answer a number of the limitations mentioned.

Technology

In this project, we use the flexibility of Java to integrate GIS functionality with Java3D (and Java-wrapped OpenGL) visualization capabilities. Specifically we are using a combination of Java/Java3D and the recently introduced ArcEngine product to create a prototype of a scientific GIS. We combine the spatial tools exposed through the ArcEngine Java API with the analytical capabilities of algorithms written in Java with the complex visualization capabilities of Java3D. Modules from each of these technologies will be combined to create innovative tools to allow users to import georeferenced data, make spatial selections, perform spatial and scientific analyses and output the results as visualizations for further examination. Use of the ArcIMS Java Connector will allow these modules to be implemented in ArcIMS sites for web-based analysis.

ArcGIS Engine is an ESRI developer product for creating and deploying ArcGIS solutions. It is a simple API-neutral cross-platform development environment for ArcObjects - the C++ component technology framework used to build ArcGIS. ArcObjects are the core of the ArcGIS functionality and include tools such as overlay - union, intersect; proximity - buffer, point distance; surface analysis - aspect, hillshade, slope; and data conversion - shapefile, coverage and DEM to geodatabase. ArcEngines' object library makes full GIS functionality available through fine- and coarse-grained components that can be used in Java and other environments. Using ArcEngine, one can build solutions and deploy them to users without requiring the ArcGIS Desktop applications (ArcMap, ArcCatalog) to be present on the same machine. It supports all the standard development environments, including Java, C++, and .NET, and all the major operating systems. In addition, one can embed some of the functionality available in the ArcGIS extensions. This product is a developer kit as well as deployment package for ArcObjects technology. Using ArcEngine we integrate GIS functionality into an application with the data being available for calculations in non-GIS components. We also are able to make these tools available to ArcIMS sites. ArcIMS has a limited set of spatial capabilities but it is capable of interfacing with the Java3D API vis the Java Connector. This will allow the ArcIMS community to utilize tools built with this project.

Java allows us to make a variety of scientific calculations on the data and to provide the results back both to the GIS component and to a Java3D based visualization component. We are able to take advantage of a number of Java utilities such as Unidata IDV, OceanShare, ncBrowse, SGT toolkit, and TimeSeries applet. The ArcIMS Java Connector could be used to produce a map coordinates base that would allow data retrieval from a DODS/OPeNDAP server, and subsequent plotting with tools designed for interaction with gridded fields.

The Java3D API is an application programming interface used for writing stand-alone three-dimensional graphics applications or Web-based 3D applets. It gives developers high level constructs for creating and manipulating 3D geometry and tools for constructing the structures used in rendering that geometry. With Java3D API constructs, application developers can describe very large virtual worlds, which, in turn, are efficiently rendered by the Java3D API. The Java3D API extension is designed as a high-level platform independent 3D graphics programming API and is amenable to very high performance implementations across a range of platforms.

We also explored the use of a second 3D API called the Visualization Toolkit (www.kitware.com). Like Java3D, VTK is a cross-platform 3D application programming interface built upon, and independent of, the native rendering library (OpenGL, etc). It exposes Java bindings (as well as Tcl and Python). It is written in C++ and includes scene-graph, lighting models, and graphic primitives similar to those in Java3D. What VTK does that Java3D doesn't (yet) do is boolean operations on 3D volumes (intersection, union), volume rendering, filtering, including convolution, FFT, Gausian, Sobel filters, permutation, high- and low-pass

Butterworth filters, and divergence and gradient calculation.

Applications

Combining these technologies, we will be creating applications for the NOAA nowCoast project and for the NOAA National Marine Fisheries Service (NMFS). The nowCoast project is an Office of Coast Survey effort to provide forecast model developers and the coastal community with centralized access to real-time physical, meteorological, oceanographic, river, and air/water quality information. The Web portal also provides NOAA forecasts for major estuaries, seaports, and adjacent coastal regions as well as the Great Lakes. The application this project develops will serve 3D rendered objects of model data. It will enhance nowCoast by the addition of a number of 3D visualization and spatial analysis tools. These will include creating 2D and 3D plots from a polyline, creating 2D velocity plots from point input (columnar plot) and a capability for viewing model data in 3D. A second application, for the NOAA/NMFS Alaska Fisheries Science Center, will allow scientists to calculate a variety of statistics and measures about the intersection of vector and volumetric objects with volumes in 3D. The objects could include marine mammal tracklines and schools of fish and the 3D volumes might include schools of prey and water masses such as cold pools. These types of volume on volume intersections could be generalized to a number of coastal and offshore applications. They will also serve as templates for many other tasks.

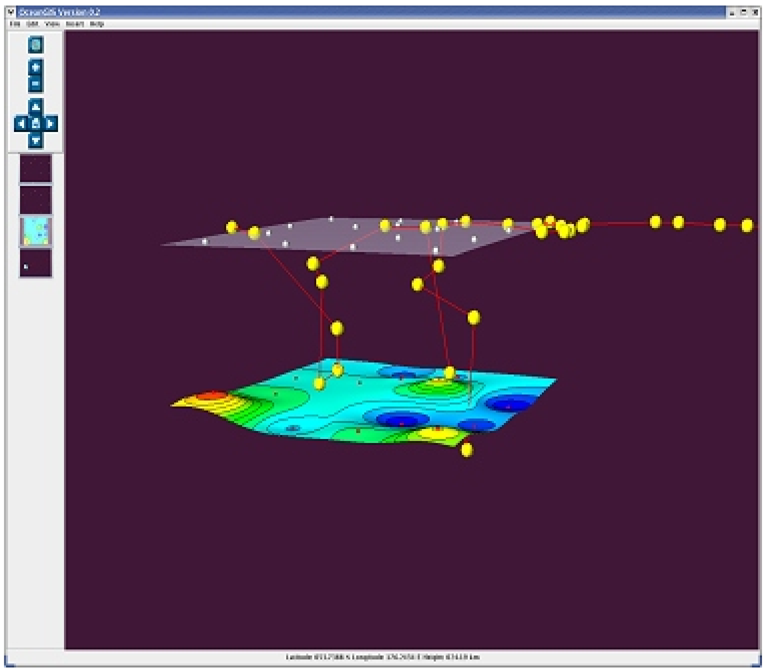
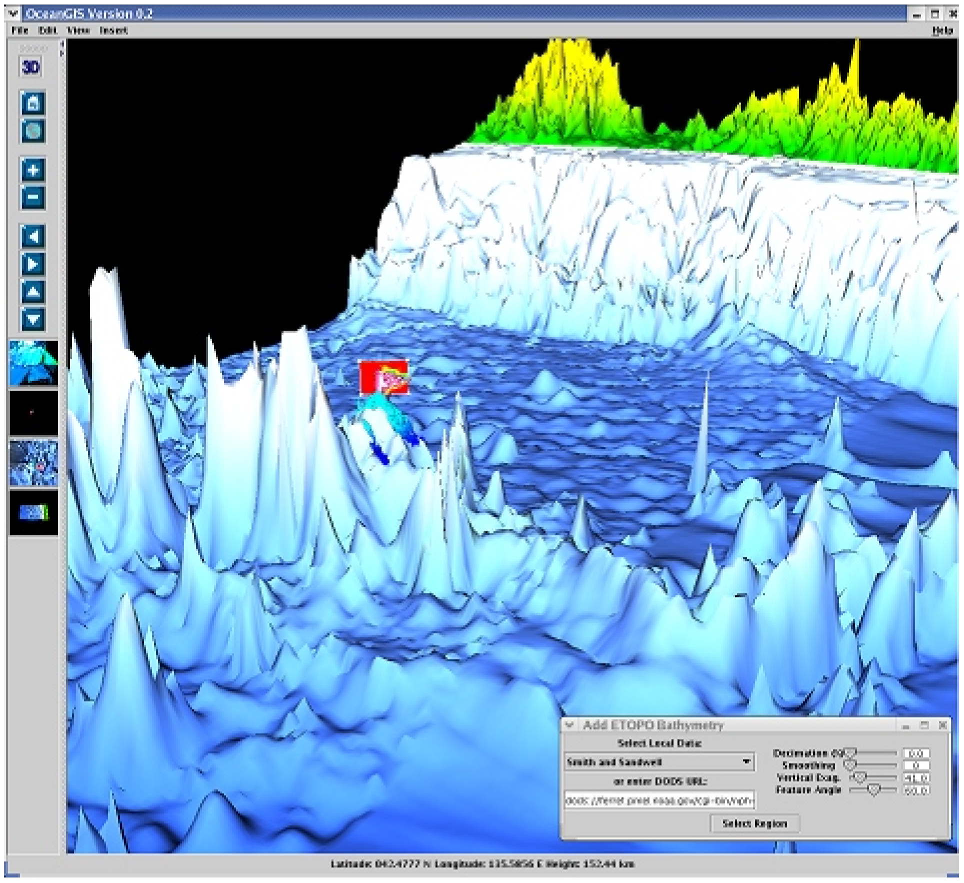


Figure 1 - Example of a 3D track and a kriged surface shown in OceanGIS.

Initial results

A test application using the project framework was put together to view high-resolution global topography/bathymetry, ocean model output, and standard ocean hydrographic data. For this application (dubbed OceanGIS) we created a number of Java objects that wrap data and methods for acting on the data. For bathymetry data, this was just the ability to exaggerate the vertical coordinate and to decimate and smooth the resulting mesh. But for more complicated data like hydrographic surveys, the objects allow for the dynamic creation of such typical oceanographic parameters as dynamic height, mixed-layer depth, and geostrophic flow. For model output, variables such as salinity can be shown as an isosurface or vector fields such as water velocity can be shown as a plane of 3D vectors that the user can move to "probe" the data. The user can seed the flow field with Lagrangian floats that are time-stepped to show particle paths.
Figure 2 - OceanGIS interface showing graphical objects associated with tools.



Smith and Sandwell
dods://ferret.pmel.noaa.gov/cgi-bin/nph-
Decimation 0.0
Smoothing 0
Vertical Exag. 41.0
Feature Angle 60.0

Latitude: 042.4777 N Longitude: 135.5856 E Height: 152.44 km

Conclusions

While none of the various technologies we have used is complete in and of itself, the linking of GIS, Java, Java3D and VTK provides a powerful mechanism to create the beginnings of a "scientific GIS". Once the technologies have been linked, the form of the final tools deployed will be dependent upon the users needs and the relative costs of the GIS portions of the technologies. Because we are creating scientific tools rather than a commercial product, license costs may have to play a large role in our choice of which GIS tools to use.

Acknowledgments

This publication was supported by the Joint Institute for the Study of the Atmosphere and Ocean (JISAO) under NOAA Cooperative Agreement #NA17RJ1232, Contribution #nnnn. PMEL contribution 2831. The views expressed herein are those of the author(s) and do not necessarily reflect the views of NOAA or any of its sub-agencies. This work was funded by NOAA's HPCC program. Mention of a software package does not imply endorsement of the package.

References

PMEL scientific visualization website:
http://www.pmel.noaa.gov/vrl/OceanGIS
VTK website:
http://www.kitware.com/VKT
Java3D website:
http://java.sun.com/products/java-media/3D

Author Information

Tiffany C. Vance
Computer Specialist
NOAA/NMFS/AFSC
7600 Sand Point Way NE
Seattle, Washington, 98115
206-526-6767
tiffany.c.vance@noaa.gov