Volume modeling of soils using GRASS GIS 3D-Tools

presented at
Second Italian GRASS Users Meeting,
University of Trento, Feb. 1-2 2001

Markus Neteler

University of Hannover – Geographical Institute
Schneiderberg 50 – 30167 Hannover, Germany
neteler@geog.uni-hannover.de

July 2001

Abstract

The analysis of spatial variability of soils has been of interest to soil scientists and geographers for quite some time. Information on soil properties are usually available from a limited number of point measurements and spatial estimates are prepared in two dimensions (e.g. by interpolation or other technique). However, soil is essentially a 3D object with varying properties in all spatial dimensions.

This study focuses on 3D capabilities of GRASS GIS providing new 3D tools to manipulate, analyse and model 3D landscape phenomena. For example, the multivariate interpolation method – regularized spline with tension (RST) – has a capability to interpolate and analyse geometric properties of selected soil properties in three-dimensional space. We investigate the options of modeling dynamic processes occuring in soil using simple 3D map algebra algorithms. An inherent part of scientific investigation and analysis is visualization. New GRASS visualization tools exploit 3D OpenGL graphics capabilities, coupling to external visualization software (Vis5D, Hibbard et al. 1994) allows animated views to time-dependent processes in soil volumes.

An increasing number of available 3D environmental data requires a complex GIS solution for manipulation, analysis and modeling. Using G3D library with new tools for modeling and visualization, GRASS has proved that it fulfills these requirements.

Key words: volumes, voxel, GRASS, GIS, soil representation, visualization, dynamic processes

1 Introduction to 3D raster modeling

Volume modeling is a more and more demanded feature of Geographical Information Systems (GIS). As most real-world phenomena are located within 3D space and usually also a time component (changes or fluxes), so the demand for a related 3D/4D data representation in GIS is increasing. While volume analysis is already common in geophysics and groundwater modeling software packages, such tools are not yet integral component of off-the-shelf GIS software.

The Geomatics Workbook N. 2 (2001)
Politecnico di Milano, Italy (ISSN 1591-092X)

There is a need for modeling tools as numerous countries have environmental laws which demand the assessment of the effects of certain plans and programmes on the environment. A recent example is the "Strategic Environmental Assessment for policies, plans and programmes" (SEA) from European community. On 31 May 2001 the European Parliament and on 5 June 2001 the Council formally adopted the SEA Directive 2001/42/EC. It shall ensure that significant environmental impacts are identified and assessed and taken into account in the decision-making process to which the public can participate[1].

This paper focuses on GIS-technical aspects of 3D modeling. Depending on the scientific context, two main approaches are used for volume representations:

- three dimensional triangulated irregular networks (3D TINs) based on vector/point information, and
- 3D raster pixels (voxels).

The first approach commonly uses 3D Delaunay triangulation with minimized interior angles and with the property that a circle around three points of any triangle doesn't include any other points (Tucker et al. 2000). As a major advantage TINs support dynamic resolutions: An optimized data representation is possible through variable resolution, i.e. different triangle sizes. However, the general structure of TINs leads to complex algorithm in case of topological analysis or transport processes simulation. Usually finite elements or finite differences are used demanding high computational power as provided by parallel computers.
A different approach of volume discretization are volume pixels, so-called "voxels". As the cube edge lengths are fixed, the resolution is common within the volume. This results in a cubic growing memory demand. This disadvantage is nowadays solved technically as ordinary PC workstations provide a large amount of memory. The major advantage of voxels is the simple internal structure with implicit topology. So the voxel format reduces the complexity of data management as no mesh is needed to be established. The voxel approach implemented in GRASS-3D performs on rather any (PC) workstation. In contrast to TINs the problem of resolution definition has to be addressed before starting the modeling process. As known from 2D raster processing the 3D raster resolution has to be defined according to the highest resolution needs. The user should consider the relation between optimal resolution and increasing file memory and computation power demands - a balance between performance and accuracy.
The basic voxel library development has been undertaken a few years ago, along with data management and interpolation tools voxel support was integrated into GRASS 5.0.0 recently. Beside data import and export modules two volume modeling tools are available. They allow to interpolate volumes from 3D sites data (point data with three spatial dimensions and values), either utilizing the 3D-IDW (inverse distance weighted) or the 3D-RST (regularized splines with tension) algorithm. Spatially dependent analysis can be performed with a 3D map calculator which allows to use common algebraic, trigonometric and binary functions and operators. Finally two visualization approaches are available: A GRASS internal OpenGL-based viewer for isosurface visualization and an interface to the external Vis5D OpenGL-based visualization and query tool initially developed for meteorological applications.

[1] http://europa.eu.int/comm/environment/eia/sea-legalcontext.htm

2 The “Ambergau” study site

The project area is situated 16km south-east to Hildesheim southern to Hannover in Lower Saxony. It is part of the Lower Saxonian hill country with an average elevation height of 200m and average slopes around 4 degrees. Due to the loessial cover and the long-term intense agricultural landuse, soils have been transformed to para-brown earths. For these investigations 78 soil samples were used from a survey undertaken at Institute of Physical Geography and Landscape Ecology, University of Hannover. The survey area size is 3.5km x 3.5km. Per drill hole three to five samples have been taken at different depth and analysed in laboratory. Various soil parameters have been measured to describe soil characteristics. However, the present study focuses on technical aspects rather than soil scientific questions.

3 3D data handling

Volumetric sample data are collected at 3D spatially distributed sampling points. Usually, especially in soil sciences, samples are taken along the third dimension (depth) in irregular spacings. The data representation for GRASS volume modeling uses 3 spatial dimensions (x, y, z) and one or more data dimension (w) (Neteler 2000):

$$u = f(x, y, z, w)$$

The well known “sites lists” can be used to generate such sample lists, they can be extended to any dimension and hold multiple attributes for one point. The format for this record is:

```
east|north[|dim]...|#cat %double [%double] @string [@string]
```

Note, that the pipe character ("|") is used to separate the dimension fields in the records, blank spaces are used to separate decimal and string descriptions. This format can be written either directly and the file stored in `$LOCATION/site_lists/` or data lists imported with *s.in.dbf* or *s.in.ascii*. It is important that the column order follows the example above.

4 3D modeling tools

4.1 From point data to volumes

Two basic methods have to be distinguished when transforming 3D point data to volumetric data:

- direct conversion of 3D points to their 3D voxel representation restricted to existing data values
- rendering of a full volume through 3D interpolation including missing-values estimation

Direct conversion can be performed with *s.to.rast3*. Before invoking the process, the 3D region has to be defined with *g3.region* or, scriptable, *g3.setregion* similar to 2D GRASS regions. If no 3D region yet exists, it may be created with *g3.createwind*. This module reads the 2D region definitions specified at GRASS startup and extends it by the third spatial dimension along with a user defined voxel resolution. In case of soil data modeling negative z values will be defined, in case of meteorological or other above-surface applications positive z values will be used. All voxels not being related to a sites list entry receive a NULL (no data) value during conversion. The user has to take care to choose the resolution as high as needed to avoid several data points falling into one voxel (information loss).

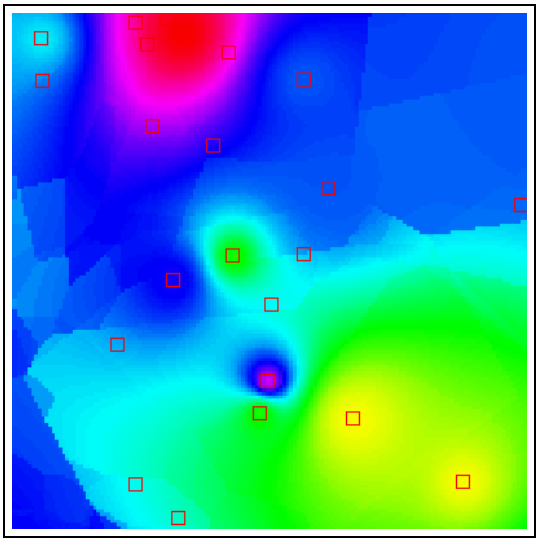

Figure 1: Nadir view on 3D-IDW volume cross section from interpolated pH values based on 3D soil sites.

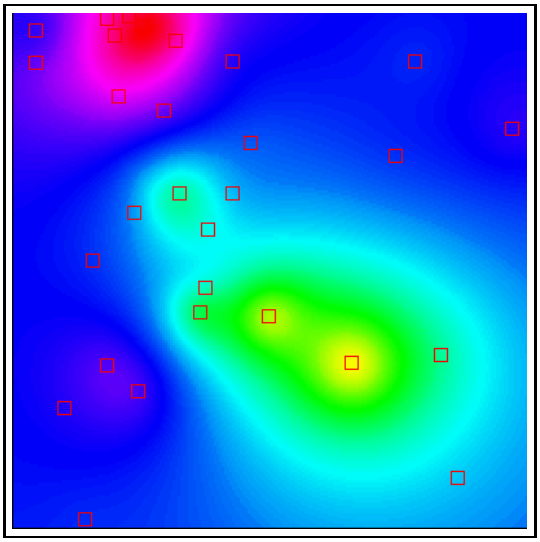

Figure 2: Nadir view on 3D-RST volume cross section from interpolated pH values based on 3D soil sites.

In contrast to direct conversion the *s.vol.idw* module (developed by Hofierka 1999 from 2D *s.surf.idw* algorithm) interpolates missing values. Fig. 1 demonstrates a horizontal cutting plane through the interpolated volume. However, it is obvious that the search radius of the moving window leads to some unexpected results, the 3D-IDW tends to cluster in case of distant sample points.
A much more sophisticated interpolation tool is *s.vol.rst* (developed by Mitás, Mitásová and others, see also Hofierka et. al. 2002). It uses the regularized splines with tension (RST) algorithm (Mitásová & Mitás 1993, Mitásová & Hofierka 1993) in three dimensions. Beside the core interpolation method it offers calculations of various geometric parameters: magnitude of gradient, horizontal and vertical aspects, change of gradient, Gauss-Kronecker and mean curvatures. Of interest for a mixed 2D/3D analysis are cross sections, the module will produce 2D maps but utilizing the 3D-RST algorithm for calculations (for applications see Hofierka et. al. 2002). Fig. 2 shows the same soil pH values interpolated with *s.vol.rst.*

4.2 Further 3D management and modeling tools

Similar to 2D GRASS masks can be defined in 3D space using *r3.mask*. Such masks may be developed with *r3.mapcalc* module (developed by Paudits & Hofierka 2000 from *r.mapcalc*). However, the strength of this module lies in it's capability for 3D data manipulation and exploration as algebraic, trigonometric and binary functions and operators are provided. For complex operations a 3D neighborhood modifier is available: voxel positions relative to the moving voxel center can be defined as map[r,c,d] (row, column, depth). In iterative environments like scripts *r3.mapcalc* can be used to dynamically simulate time variant processes in a volume.
A simple example shall demonstrate how to achieve dynamic modeling using *r3.mapcalc*. The script is adapted from the hydrologic model presented by Shapiro & Westervelt 1992 and extended to 3D.
While the 2D *r.mapcalc* uses a 3x3 moving window with a centered pixel, the 3D module *r3.mapcalc* used a 3x3x3 cube with a centered voxel. Considering a soil volume, the cube shows three planes. For a dynamic model the upper and the middle plane shall be considered. Related to the center voxel of the cube (located in middle plane) fluxes from and to this middle voxel of the cube are considered depending on the local gradient. Generally a cube offers 3^3-1 = 26 directions related to the middle voxel

(3^2-1=8 directions in 2D space). For the simple dynamic model presented here 17 flow directions are taken into account as the moving cube will itself decent within the soil volume. Capillary rise it not implemented here. For a more realistic flow distribution, the 17 fluxes are weighted according to their direction.

The overall contribution to the middle voxel is set to 100%. Following assumption is implemented within the model: The upper plane shall contribute 70% of flows into vertical directions, while the middle plain (except the middle voxel) contributes 30% of lateral fluxes. This leads to a set of conditions, nine conditions for the upper plane of the 3x3x3 cube and 8 conditions for the middle plane, altogether 17 if-conditions as described below. The individual voxel contributions are dependent from their geometrical position to the middle voxel. So the individual weighting is calculated according to the contribution-per-plane and position. This leads to three equations for the upper plane and two equations for the middle plane which can be easily solved. Each equation for the outer voxels is used four times, so the weight coefficient is divided by four to receive the final weights as they can be found in the script below.

The example below has been run on a volume of permeability coefficients (rendered with *s.vol.rst*) which have been measured in the soil profiles of the Ambergau study site. During the simulation a water flow (volume "water") is drained through the volume, controlled by the permeability coefficients stored in the volume (volume "pcoeff"):

```
water = water + eval(x = pcoeff + water, \
 if (x > (y = pcoeff[0, 0, -1] + water[0, 0, -1]), \
   -.3064 * if (pcoeff > y, water, x - y), \
   .3064 * if (pcoeff[0, 0, -1] > x, water[0, 0, -1], y - x))+ \
 if (x > (y = pcoeff[0, -1, -1] + water[0, -1, -1]), \
   -.0542 * if (pcoeff > y, water, x - y), \
   .0542 * if (pcoeff[0, -1, -1] > x, water[0, -1, -1], y - x))+ \
 if (x > (y = pcoeff[1, 0, -1] + water[1, 0, -1]), \
   -.0542 * if (pcoeff > y, water, x - y), \
   .0542 * if (pcoeff[1, 0, -1] > x, water[1, 0, -1], y - x))+ \
 if (x > (y = pcoeff[0, 1, -1] + water[0, 1, -1]), \
   -.0542 * if (pcoeff > y, water, x - y), \
   .0542 * if (pcoeff[0, 1, -1] > x, water[0, 1, -1], y - x))+ \
 if (x > (y = pcoeff[-1, 0, -1] + water[-1, 0, -1]), \
   -.0542 * if (pcoeff > y, water, x - y), \
   .0542 * if (pcoeff[-1, 0, -1] > x, water[-1, 0, -1], y - x))+ \
 if (x > (y = pcoeff[-1, -1, -1] + water[-1, -1, -1]), \
   -.0442 * if (pcoeff > y, water, x - y), \
   .0442 * if (pcoeff[-1, -1, -1] > x, water[-1, -1, -1], y - x))+ \
 if (x > (y = pcoeff[1, -1, -1] + water[1, -1, -1]), \
   -.0442 * if (pcoeff > y, water, x - y), \
   .0442 * if (pcoeff[1, -1, -1] > x, water[1, -1, -1], y - x))+ \
 if (x > (y = pcoeff[1, 1, -1] + water[1, 1, -1]), \
   -.0442 * if (pcoeff > y, water, x - y), \
   .0442 * if (pcoeff[1, 1, -1] > x, water[1, 1, -1], y - x))+ \
 if (x > (y = pcoeff[-1, 1, -1] + water[-1, 1, -1]), \
   -.0442 * if (pcoeff > y, water, x - y), \
   .0442 * if (pcoeff[-1, 1, -1] > x, water[-1, 1, -1], y - x))+ \
 if (x > (y = pcoeff[0, -1, 0] + water[0, -1, 0]), \
   -.0439 * if (pcoeff > y, water, x - y), \
   .0439 * if (pcoeff[0, -1, 0] > x, water[0, -1, 0], y - x))+ \
 if (x > (y = pcoeff[1, 0, 0] + water[1, 0, 0]), \
```

```
   -.0439 * if (pcoeff > y, water, x - y), \
   .0439 * if (pcoeff[1, 0, 0] > x, water[1, 0, 0], y - x))+ \
 if (x > (y = pcoeff[0, 1, 0] + water[0, 1, 0]), \
   -.0439 * if (pcoeff > y, water, x - y), \
   .0439 * if (pcoeff[0, 1, 0] > x, water[0, 1, 0], y - x))+ \
 if (x > (y = pcoeff[-1, 0, 0] + water[-1, 0, 0]), \
   -.0439 * if (pcoeff > y, water, x - y), \
   .0439 * if (pcoeff[-1, 0, 0] > x, water[-1, 0, 0], y - x))+ \
 if (x > (y = pcoeff[-1, -1, 0] + water[-1, -1, 0]), \
   -.0311 * if (pcoeff > y, water, x - y), \
   .0311 * if (pcoeff[-1, -1, 0] > x, water[-1, -1, 0], y - x))+ \
 if (x > (y = pcoeff[1, -1, 0] + water[1, -1, 0]), \
   -.0311 * if (pcoeff > y, water, x - y), \
   .0311 * if (pcoeff[1, -1, 0] > x, water[1, -1, 0], y - x))+ \
 if (x > (y = pcoeff[1, 1, 0] + water[1, 1, 0]), \
   -.0311 * if (pcoeff > y, water, x - y), \
   .0311 * if (pcoeff[1, 1, 0] > x, water[1, 1, 0], y - x))+ \
 if (x > (y = pcoeff[-1, 1, 0] + water[-1, 1, 0]), \
   -.0311 * if (pcoeff > y, water, x - y), \
   .0311 * if (pcoeff[-1, 1, 0] > x, water[-1, 1, 0], y - x)))
```

This model has been stored as "3dvol_water.mapcalc" and used as input for *r3.mapcalc*. To achieve dynamic modeling the script needs to run several times in a loop, which leads to flows passing through the volume. The script "3dflow.sh" to run the simulation have been written as follows (adapted from Shapiro & Westervelt 1992):

```
#!/bin/sh
# 3dflow.sh to run 3dvol_water.mapcalc in r3.mapcalc

#initialize surface water:
r3.mapcalc water="if(depth()>1, 0, waterraw)"
i=1
# next number must match the number of depth levels in volume:
while [ $i != 100 ]
do
  n=1
  while [ $n != 10 ]
  do
   # run the simulation:
   # (note: don't use space(s) after \ characters in model):
    r3.mapcalc < ./3dvol_water.mapcalc
    n=`expr $n + 1`
  done

  #reset NULL to 0 water for next iteration:
  r3.null grid3=water null=0
  #export every 10th volume to vis5d for visualization:
  r3.out.v5d grid3=water out=water.$i.v5d
  i=`expr $i + 1`
done
```

After setting up a 3D region with *g3.setregion*, the flow model requires an initialization with a water layer on the surface. Such a map can either be provided by 2D GRASS and converted to 3D format with *s.to.rast3* or

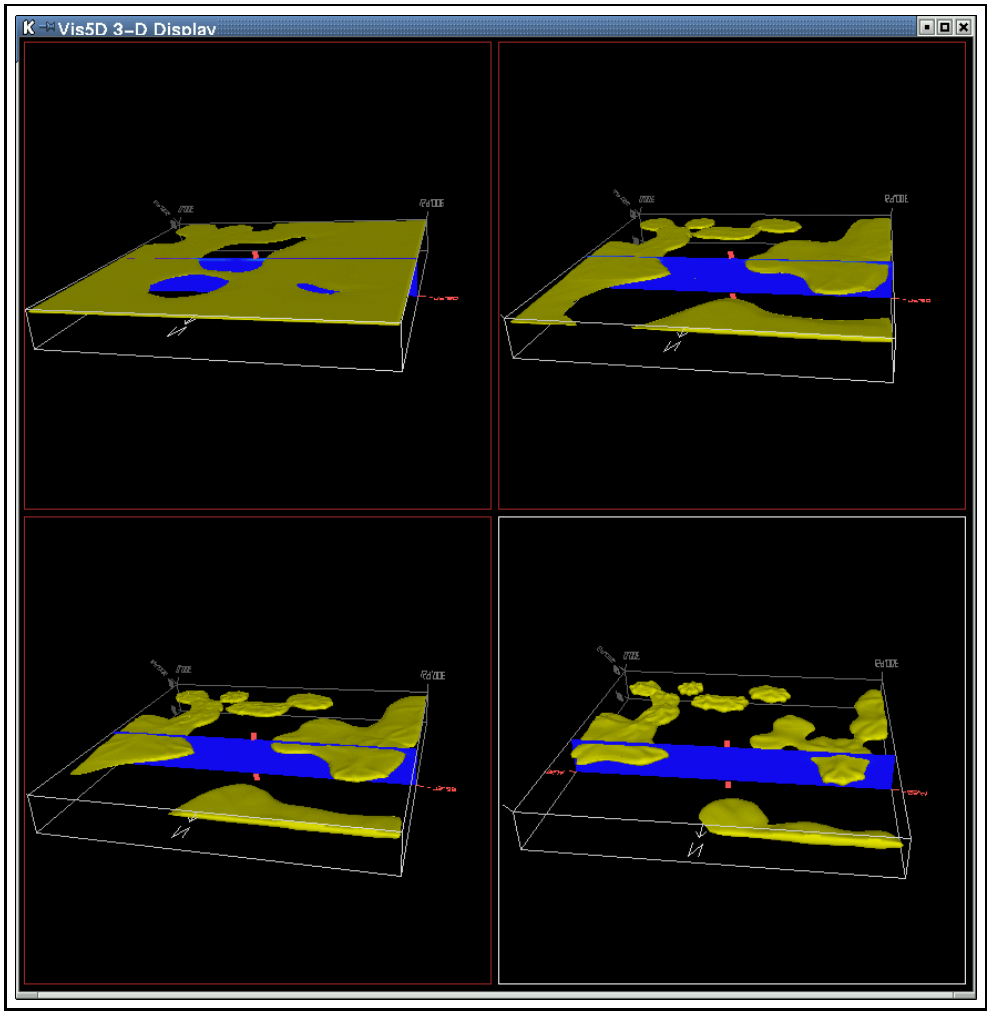

Figure 3: Dynamic water flow through a soil volume using r3.mapcalc – displayed with Vis5D visualization tool.

directly generated with *r3.mapcalc*. A simple example is the initialization with an overall identical water input. First a volume of water in GRASS 3D is generated (5mm):

```
r3.mapcalc waterraw="5"
```

Then this volume needs to be reduced to the surface, below the voxels are initialized with 0:

```
r3.mapcalc water="if(depth()>1, 0, waterraw)"
```

The contents of the volume “water” may be verified either with *r3.out.ascii* or by using visualization tools described below. When running the “3dflow.sh” every 10th volume will be exported into Vis5D format using the “r3.out.v5d” module. These volumes will show the waterfront passing through the soil volume depending on the local “pcoeff” values within vertical direction. An example can be seen in figure3.

5 Data export and visualization

The new volume analysis and visualization tools are of special interest as they are seamless integrated into a common GIS environment. This minimizes efforts of data conversion between data import, analysis and visualization. Two different methods are available: A GRASS built-in volume viewer which is still in experimental stage and an interface to the external, freely available, Vis5D volume viewing and query tool.

5.1 GRASS built-in viewing tools

A low level option to display the spatial distribution (but not the attributes) is to convert the calculated volume back to 3D sites using *r3.to.sites*, then to display these sites in NVIZ (see fig.4).
Alternatively GRASS provides the experimental OpenGL viewing tool *r3.showdspf*. First a "display file" has to be generated with *r3.mkdspf* to define levels of isosurfaces. In volumetric environments isosurfaces are the analogue to isolines in 2D environment surfaces of identical value. Those can then be displayed (zoom, rotation, selective display) through *r3.showdspf* (see fig.5 for 3D pH values).

5.2 Coupling to external OpenGL viewer Vis5D

After exporting with *r3.out.v5d* GRASS volumes can be displayed in Vis5D visualization software (Hibbard et al. 1994). This tool offers various methods to render rotatable semi-transparent volumes, isosurfaces, movable cutting planes and isolines. The code is based on OpenGL which may use hardware acceleration for volume display if a special video card is used. Of special interest is the feature of 3D queries within the volume. Fig. 6 shows an isosurface view onto the pH value interpolated volume. The same volume can be displayed as semi-transparent volume (fig. 7).

6 Future needs

To improve the interpolation of volumes more closely to the analysed phenomena, a constraint interpolation would be needed. In case of improved soil volume modeling the introduction of 3D boundaries probably in vector format, could support the quality of regionally restricted interpolation keeping soil horizon boundaries or stratifications.
Due to different data availability the implementation of multiple resolution within one volume may achieve the advantage of dynamic resolution (comparing to TINs) while keeping the simply intrinsic topology of voxels. A way to support dynamic resolutions may be reached through voxel management in oct-tree structures. The module *s.vol.rst* internally already uses oct-trees for large data processing, so the basic routines are already present in GRASS 5.0.0.
The powerful voxel calculator *r3.mapcalc* needs a built-in access to 2D raster data (as already found in *s.vol.rst*). In terms of soil modeling 2D data may be used as "seed information" to simulate fluxes within the volume which start on top of it.

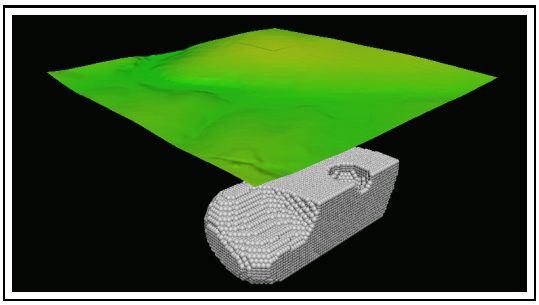

Figure 4: Volume visualized as 3D sites in NVIZ visualization tool.

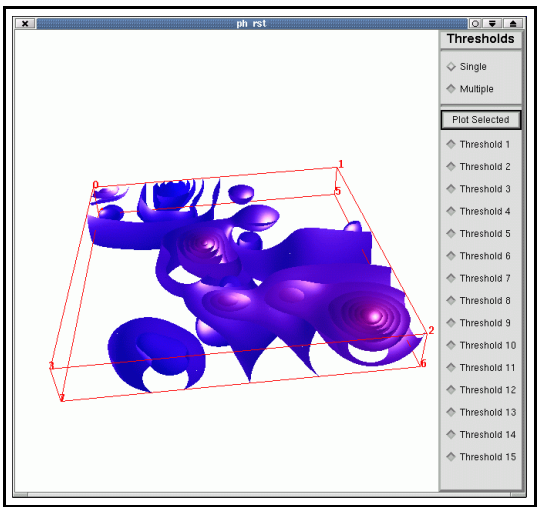

Figure 5: Volume visualized with r3.showdspf visualization tool.

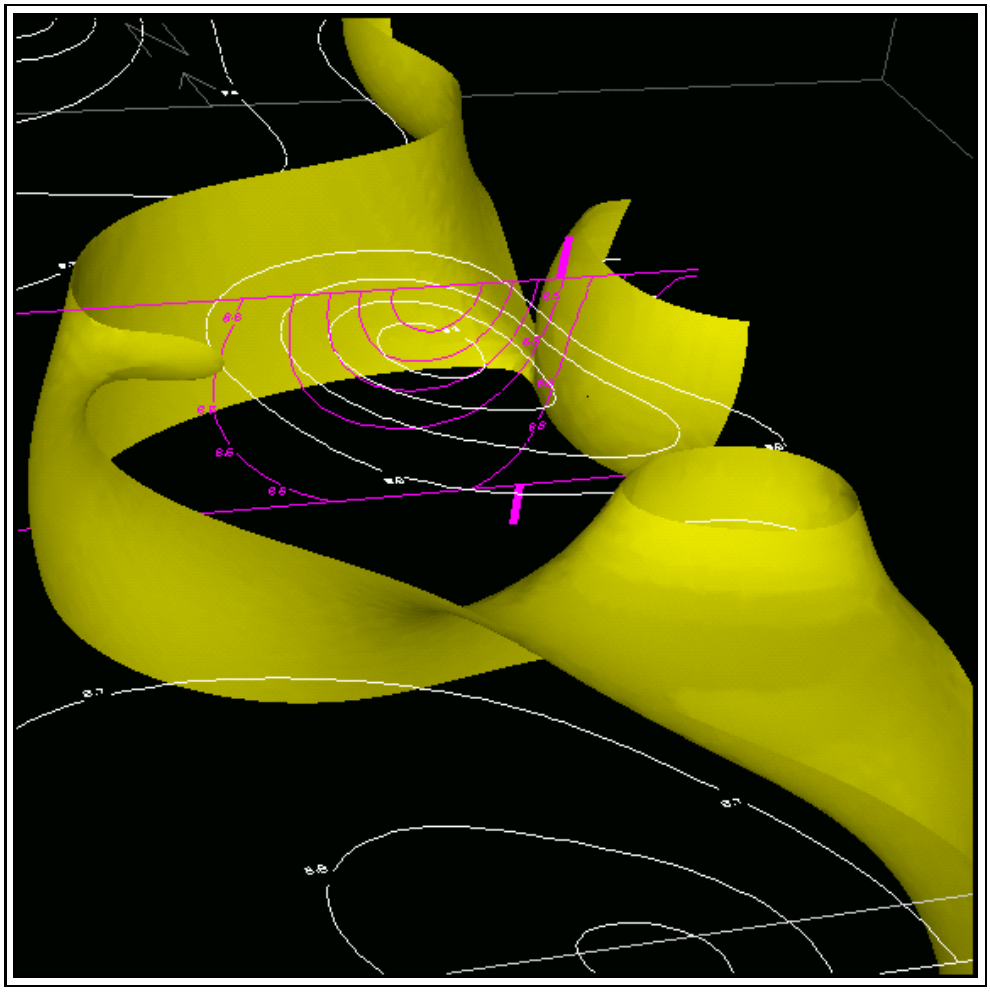

Figure 6: 3D pH values displayed in Vis5D visualization tool: isosurface view

At time of this writing still inconsistencies between the 2D and the 3D environment have to be fixed. This will be addressed in a future release.

7 Summary

From the technical point of view the implemented voxel technology performs on common PC workstation. The present modules are much more than a basic environment for 3D modeling, the strength lies in the seamless integration into a GIS. Beside volume interpolation GRASS offers a comprehensive voxel calculation tool which is the 3D version of the well known 2D map calculator. Volumes can be visualized internally and also exported to Vis5D tool.

There is a need for integration of tilted or curved surface boundaries, which can eventually be achieved through vector representation.

From the scientific point of view the main problem is the availability of 3D data. However, this problem may be solved only by the GIS user through intense surveys rather than through modified programs. It is important to note that full-volume-interpolations don't represent natural phenomena if the raw data are sparse. Due to the current module limitations 3D modeling within boundaries (e.g. soil or geological units) is not possible. For dynamical modeling the time representation is incomplete, however, the current version already supports timestamps. As GRASS is open source software, programmers interested in 3D development find full access to code and algorithms.

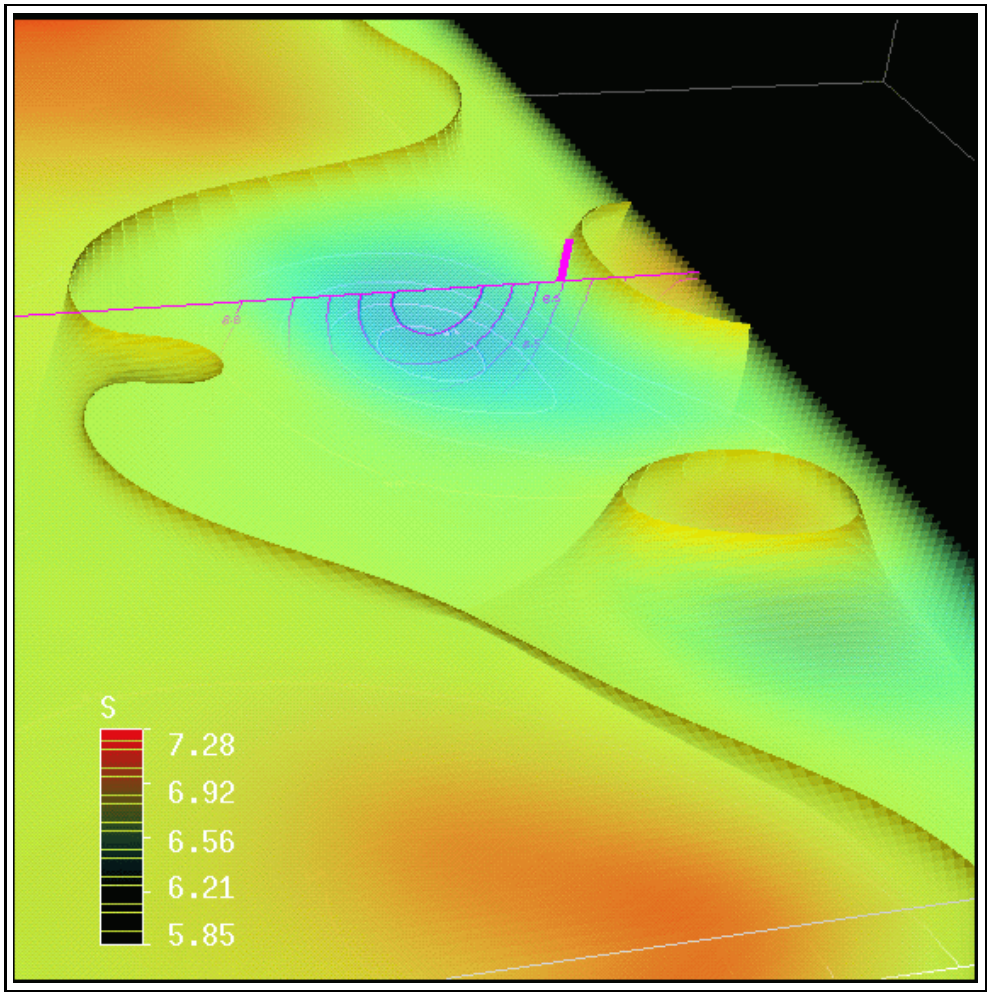

Figure 7: 3D pH values displayed in Vis5D visualization tool: volumetric view

8 References

Hibbard, W. L., B. E. Paul, D. A. Santek, C. R. Dyer, A. L. Battaiola, M.-F. Voidrot-Martinez (1994) – Interactive Visualization of Earth and Space Science Computations, Computer 27, No. 7, July 1994, 65-72. http://www.ssec.wisc.edu/~billh/vis5d.html

Hofierka, J., J. Parajka, H. Mitásová, L. Mitás (2002) – Multivariate Interpolation of Precipitation Using Regularized Spline with Tension, Transactions in GIS, (accepted for publication).

Mitás, L., W.M. Brown, H. Mitásová (1997) – Role of dynamic cartography in simulations of landscape processes based on multi-variate fields. Computers and Geosciences, Vol.23, No. 4, pp. 437-446, http://www.elsevier.nl/locate/cgvis.

Mitásová H. L. Mitás (1993) – Interpolation by Regularized Spline with Tension: I. Theory and Implementation, Mathematical Geology 25, 641-655.

Mitásová H., J. Hofierka (1993) – Interpolation by Regularized Spline with Tension: II. Application to Terrain Modeling and Surface Geometry Analysis, Mathematical Geology 25, 657-667.

Mitásová, H., W.M. Brown, J. Hofierka (1994) – Multidimensional dynamic cartography. Kartografické listy 2, pp. 37-50.

Mitásová, H., L. Mitás, W.M. Brown, D.P. Gerdes, I. Kosinovsky (1995) – Modeling spatially and temporally distributed phenomena: New methods and tools for GRASS GIS. International Journal of GIS, 9 (4), special issue on integration of Environmental modeling and GIS, p. 443-446.

Neteler, M. (ed.) (2000) – GRASS 5.0 Programmer's Manual. Geographic Resources Analysis Support System. University of Hannover.
http://grass.itc.it/grassdevel.html

Shapiro, M., J. Westervelt (1992): r.mapcalc. An Algebra for GIS and Image Processing. U.S.-CERL, Champaign Illinois, 22 pp.
http://grass.itc.it/gdp/

Tucker, G., N. Gasparini, R. Bras, S. Rybarczyk, S. Lancaster (2000) – An Object-Oriented Framework for Distributed Hydrologic and Geomorphic Modeling Using Triangulated Irregular Networks, Computers and Geosciences, in press.