Displaying 3-D seismic models through the VRML and GOCAD

Václav Bucha

Department of Geophysics, Charles University, Ke Karlovu 3, 121 16 Praha 2, Czech Republic, Fax: +420-2-2191 1214, E-mail: bucha@seis.karlov.mff.cuni.cz

Summary

Interface for displaying various 3-D objects in seismology using the Virtual Reality Modeling Language (VRML) or the Geological Object Computer Aided Design (GOCAD) is described and its application illustrated on numerical examples.

Keywords

Seismic models, VRML, GOCAD, SW3D software, computer graphics

1 Introduction

Software package SW3D has now two possibilities how to interactively display 3-D seismic objects defined by SW3D algorithms. The first one is the Virtual Reality Modeling Language (VRML). The conversion programs and examples were introduced on SW3D-CD-3 (Bucha, Bulant & Klimeš (eds.) 1999) but were not accompanied by an introductory paper. The second possibility how to display seismic objects is new and uses the Geological Object Computer Aided Design (GOCAD). This paper describes both possibilities.

Section 2 is devoted to the main features of VRML. Section 3 shortly describes the GOCAD visualization program and GOCAD objects and attributes used by SW3D conversion programs. Section 4 shows the use of SW3D programs to built VRML or GOCAD output files. Programs are executed through history files and one example is described step by step in section 5. The last section presents an overview of the pictures of selected SW3D models.

2 VRML

The Virtual Reality Modeling Language is designed for describing sets of simple geometrical 3-D objects in Cartesian coordinates (see the World Wide Web site of the VRML Consortium *http://www.vrml.org*). The former version VRML 1.0 has been replaced by the VRML97 (or VRML 2.0) international standard ISO/IEC 14772-1 (1997). This document contains a detailed description of the language (fundamentals, datatypes, syntax, examples, etc.). The language is capable of representing both static and animated 3-D and multimedia objects with hyperlinks to other media such as texts, sounds, movies, and images. The views of the objects, perceived by an observer interactively navigated in 3-D, can be displayed by various VRML browsers on various computers under various operating systems. At this time we have not found any free browser with the complete support of all VRML fundamental components - nodes (e.g., points, polylines, polygons, lights, material, parameters) under Linux (see browser listings at *http://www.web3d.org/vrml/browpi.htm*). We are using `vrmlview` free browser

In: Seismic Waves in Complex 3-D Structures, Report 11 (Department of Geophysics, Charles University, Prague, 2001), pp. 337–355

(*http://www.sim.no*), that is based on free Mesa 3-D graphics library (very similar to OpenGL). For VRML definitions used in SW3D programs and some remarks concerning the `vrmlview` browser refer to Appendix 1.

3 GOCAD

Visualization software GOCAD (Geological Object Computer Aided Design) is a product of the GOCAD consortium. GOCAD is an integrated and geologically oriented CAD software that provides various tools to construct a wide range of earth models, for application in geology, geophysics and reservoir engineering (ASGA 1997). GOCAD has been developed for many operating systems (see *http://www.t-surf.com*).

GOCAD objects and attributes described in ASCII format are interpreted by the GOCAD visualization program. There is no complete manual for GOCAD objects and attributes. Base for SW3D conversion programs were the GOCAD++ Developer's Guide chapter 7.1 ASCII Interface (ASGA 2000), GOCAD tutorial files, `Gocad.attributes` file that defines the layout of the standard GOCAD attribute editor and finally a lot of tests.

The GOCAD program is commercial and the possibilities of displaying SW3D seismic objects are much wider in comparison with free VRML browsers that we have tested. We are using GOCAD version 1.5.8 for Linux, which is dependent on Mesa 3-D graphics library and Motif library (we use OpenMotif). For the list of GOCAD objects and attributes used by SW3D conversion programs and some remarks concerning the GOCAD program see Appendix 2.

4 Use of SW3D programs to convert 3-D seismic models into VRML or GOCAD

The application of the VRML and GOCAD is illustrated using the current version of the SW3D software, containing programs, written in Fortran77, for conversion of points, lines and triangulated surfaces into the VRML or GOCAD and for conversion of 3-D grid of points into GOCAD Voxet. A single VRML or GOCAD file is built step by step, adding further objects by means of a further invocation of the corresponding conversion program.

Initiation of the VRML or GOCAD file is performed by the program `iniwrl.for`. Both files start with a single line that identifies the type of file. The VRML file continues with the specification of the positions and optical properties of cameras (viewpoints), lights illuminating the objects, the colour of the background, etc., according to its input parameters. `iniwrl.for` also adds prototype definitions of points, lines and surfaces (prototype is used in VRML for the definition of a new node type) to the VRML file. It is impossible to define cameras and lights directly in a GOCAD input file.

Program `ptswrl.for` converts points (sources, receivers, points to be fit by interfaces, etc.), stored in uniform files, into VRML or GOCAD. Program `linwrl.for` does analogous conversion for lines (rays, edges of the model box, profiles, geographic information like rivers or cities, etc.) stored in uniform files. Program `srfwrl.for` converts triangulated surfaces (wavefronts, interfaces between geological blocks, 2-D velocity sections embedded in 3-D, etc.) into VRML or GOCAD. Program `grdwrl.for` converts gridded data into GOCAD Voxet (3-D grid) representation. The points, lines, surfaces and grids are coloured according to the input data, in order to display the properties relevant to the objects.

If structural interfaces are defined explicitly (e.g., by a regular grid of points, or by sets of triangles with coordinates of vertices given) they can directly be converted into VRML or GOCAD. The interfaces used by SW3D software are defined implicitly (covered by parts of zero isosurfaces of smooth functions, which is suitable for ray tracing). It is necessary to triangulate the interfaces. The algorithm of the triangulation has been designed by Bulant (1999). The resolution of the algorithm (maximum size of triangles) is prescribed a priori by a given 3-D regular rectangular grid of points. Details smaller then the gridcubes may remain hidden. The model volume is split into gridcubes corresponding to the given grid. The intersections of structural interfaces with the gridcubes are approximated by polygons, formed by points located at the gridlegs and at the gridfaces (program `modsrf.for`). The algorithm has also been modified to generate triangulated 2-D velocity and similar sections through the models. The polygons are then divided into triangles (program `trgl.for`) that can be sorted according to the values specified at their vertices (program `trglsort.for`) for GOCAD object selection.

To construct a VRML or GOCAD output file, it is recommended to use history files (data files interpreted by Perl scripts), containing both the input data and the information how to execute the programs. For detailed description of input and output data and use of the programs see the source codes on SW3D-CD-5 (Bucha, Bulant & Klimeš (eds.) 2001).

5 Examples

We shall describe step by step the construction of VRML or GOCAD representation of the model with a lenticular inclusion using history files. The second part of this section shows images of selected models. The figures used in this section are GOCAD output snapshots.

5.1 Model with a lenticular inclusion

Model with a lenticular inclusion is a synthetic demo model for SW3D package MODEL. There are two history files used for building VRML or GOCAD representation of this model. The first one `len-mod.h` contains parameters and commands to transform structural interfaces and 2-D velocity sections in the model (Figure 1). The history file `len-mod.h` also builds 3-D P-wave velocity cube (GOCAD Voxet), that is possible only in GOCAD (Figure 2). The second history file `len-crt.h` computes two-point rays and adds them into the file with surfaces (Figure 2).

```
# History file 'len-mod.h' transforms surfaces and velocity sections
# in the model with LENticular inclusion into VRML or GOCAD
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Input files required
 #chk.pl: "data/len/" "len-mod.dat"
 #chk.pl: "data/len/" "len-cam.dat"
 #chk.pl: "data/len/" "len-dli1.dat"
 #chk.pl: "data/len/" "len-dli2.dat"
 #chk.pl: "forms/"    "hsv.dat"
```

We shall repeat some general rules that are used in history files. History file `len-mod.h` begins with comments, which are preceded by "`#`". Everything from hash mark "`#`" to the end of line is assumed to be a comment and is ignored. However, comments containing no equals "`=`" nor colons "`:`" need not be preceded by a hash mark, because only equals "`=`" and colons "`:`" are interpreted as commands in the history files.

Each "`=`" (equal sign) immediately preceded by a string is interpreted as the PARAMETER=VALUE couple specifying the value of the parameter named PARAMETER.

Each "`:`" (colon) immediately preceded by a string is interpreted as the instruction to execute a program. The string between the last space (or the beginning of the line if there are no spaces to the left of "`:`") and the colon is the name of the program. If the name contains substring "`.pl`", it is assumed to represent a Perl script. Otherwise, it is assumed to represent executable program, compiled from the Fortran source. Everything to the right from the colon is assumed to represent the data for the program read from the standard input (Fortran) or command line (Perl).

For example, the lines containing "`chk.pl:`" are the instructions to execute Perl script `chk.pl`, but disabled by comment sign "`#`" in the second column. The lines serve as the list of input data files. If the lines are uncommented and Perl script `chk.pl` is located in the current directory, it checks for the presence of the data files in the current directory. Moreover, if the value of global variable `$SW3D` in Perl script `go.pl` is properly edited, Perl script `chk.pl` (calling subroutine `CHK` of `go.pl`) may be employed to copy the data files into the current directory. If the file is not found, `go.pl` should stop the execution. The string preceding the data filename is the path to the root directory of the SW3D software where the original of the data file is located. However, specification of the path to the root directory of the SW3D-CD-5 software requires to manually edit file `go.pl`. Moreover, stopping the execution sometimes does not work in the current version (e.g., under MS-DOS). That is why the instructions to run Perl script `chk.pl` are commented.

```
# General settings
# ~~~~~~~~~~~~~~~~
# Form of output files
  VRML='VRML2'
 #VRML='GOCAD'

# Input data
  MODEL='len-mod.dat'

# Temporary files (several times overwritten)
  VRTX='vrtx.tmp'  TRGL='trgl.tmp'  PLGN='plgn.tmp'
```

Above part of the history file contains general settings. Here you can select the format, VRML or GOCAD, of the output file (default format is VRML97 or VRML 2.0). Next to this we specify the name of file that defines the model with a lenticular inclusion, followed by some names of temporary files.

```
# Setting of cameras (viewpoints) and lights, only VRML
  UP1=0  UP2=0  UP3=1  CAMERA='len-cam.dat'
  DLIGHT='len-dli1.dat'  AMBIENT=0.00
  R=0.00  G=0.00  B=0.00  # Black background

# Colour properties
  COLORS='hsv.dat'
  TRANSP=0.00  SPECULAR=0.22  SHININESS=0.10
```

Setting of cameras (viewpoints) and lights affects only the VRML output. We use GOCAD version 1.5.8 that enables to define lights only in a separate script. The definition of a camera, also in a separate script, is available only in the newest version of GOCAD 2.0 that was not released for Linux yet. We suppose to solve this problem in future. Parameters `UP1,UP2,UP3` are components of a vector pointing upwards. It is used to properly rotate the camera. `CAMERA` contains the name of file with cameras. `DLIGHT` specifies the name of file with directional lights (light source illuminates along rays parallel to a given 3-D vector). Parameter `AMBIENT` determines the intensity of the ambient light (illumination results from the scattering and reflection of light originally emitted directly by light sources). `R,G,B` specify colour of the background. For detailed description of these and other parameters see file `iniwrl.for` of package `FORMS`. Parameter `COLORS` contains the name of the file containing the data describing the colour map (for details see `colors.for`). `TRANSP` specifies transparency of the surfaces, `SPECULAR` contains intensity of the specular reflections from glossy surfaces and `SHININESS` determines shininess of the glossy surfaces. When the angle from the light to the surface normal is close to the angle from the surface normal to the viewer, the specular color is added to the diffuse and ambient colour calculations. Lower shininess values produce soft glows, while higher values result in sharper, smaller highlights.

Next section of history file triangulates the structural interfaces of the model with a lenticular inclusion (see Figure 1, left-hand picture).

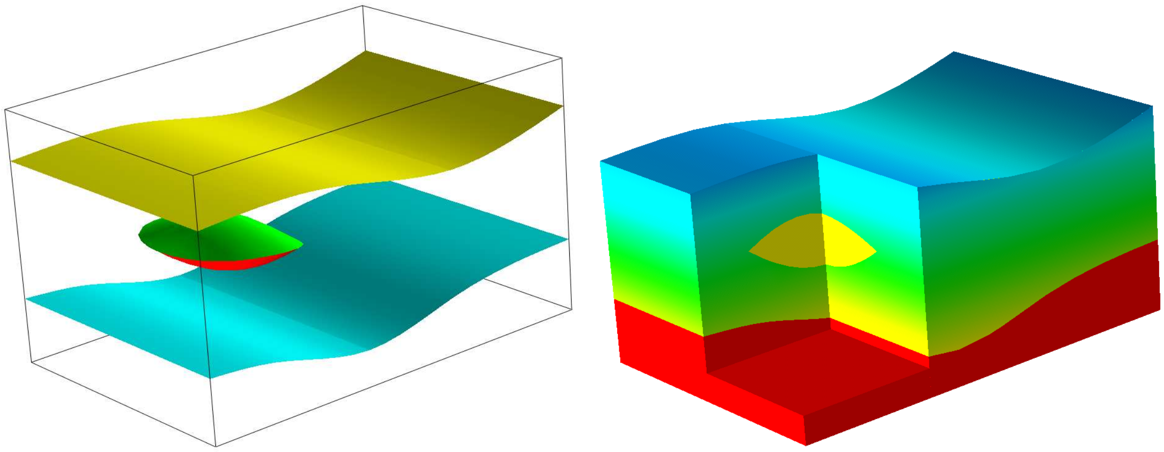

Figure 1. Interfaces coloured according to indices of surfaces (ISRF) (left-hand picture) and P-wave velocity sections (right-hand picture) in the model with a lenticular inclusion (history file `len-mod.h`).

```
# Structural interfaces in the model
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Name of output file
  WRL='len-srf.wrl'  # VRML
 #WRL='len-srf.mx'   # GOCAD
  iniwrl:

# Colours correspond to surface indices
  COLUMN07='ISRF'
  KOLSRF=7  VPER=6  VREF=1  CREF=0.
```

Parameter WRL contains the name of the VRML or GOCAD output file. In this example, there are two different extensions of the filename, one for VRML, `*.wrl` according to MIME (set of documents called Multipurpose Internet Mail Extensions), one for GOCAD, `*.mx`, that is recommended by GOCAD manuals for mixed types of GOCAD objects in one file. Program `iniwrl` uses all preceding parameters and initializes the output virtual reality description file.

Parameter COLUMNii is a string that specifies the quantity to be written to the column `ii` of the file with vertices of the polygons. First six numerical columns usually contain coordinates of the vertices and components of the normals. For details see `modsrf.for`. KOLSRF specifies the quantity to be scaled in colours. If zero, all surfaces will have the same colour given by parameters R,G,B. If positive, the quantity in KOLSRF-th numerical column of the input file with vertices of the polygons will be colour-coded at each vertex of each triangle or polygon of the surface (for details and other parameters see `srfwrl.for`). VPER, VREF and CREF are parameters specifying the colour scale. VPER is the period of values corresponding to one period in the colour map, VREF is the reference value and will be displayed in the reference colour, CREF selects the reference colour from the palette defined by parameter COLORS. Here CREF is the hue of the reference colour. For detailed description and other parameters see `colors.for`.

```
# Triangulating and plotting interfaces
  N1=25  N2=17  N3=15  D1=2.5  D2=2.5  D3=2.5  O1=0.  O2=-20.  O3=-25.
  VRTX='len-srfv.out' TRGL='len-srft.out'
  modsrf:
  trgl:
  KOLUMN=7
  VALUE=1  VRTXN='len-pts1.out'  TRGLN='len-trg1.out'  trglsort:
  VALUE=2  VRTXN='len-pts2.out'  TRGLN='len-trg2.out'  trglsort:
  VALUE=3  VRTXN='len-pts3.out'  TRGLN='len-trg3.out'  trglsort:
  VALUE=4  VRTXN='len-pts4.out'  TRGLN='len-trg4.out'  trglsort:
  PROPERTIES='N1 N2 N3 ISRF'                           # just for GOCAD
  NAME='Surface1'                                      # just for GOCAD
  VRTX='len-pts1.out'  TRGL='len-trg1.out' srfwrl:
  NAME='Surface2'                                      # just for GOCAD
  VRTX='len-pts2.out'  TRGL='len-trg2.out' srfwrl:
  NAME='Surface3'                                      # just for GOCAD
  VRTX='len-pts3.out'  TRGL='len-trg3.out' srfwrl:
```

```
  NAME='Surface4'                                   # just for GOCAD
  VRTX='len-pts4.out'  TRGL='len-trg4.out' srfwrl:
```

N1,N2,N3, D1,D2,D3, O1,O2,O3 are parameters defining the basic regular rectangular grid, number of gridpoints, origin of the grid, grid spacing. VRTX is the name of the output file with vertices of the polygons and TRGL is the name of the file describing the triangles. Program modsrf approximates structural interfaces by polygons. Program trgl divides the polygons into triangles. Program trglsort sorts triangles according to values of a selected quantity. Program trglsort reads triangles from files given by parameters VRTX and TRGL. All triangles which vertices have, in the seventh numerical column (KOLUMN=7), the value given by parameter VALUE are written to new files given by parameters VRTXN and TRGLN. Program trglsort is used to separate interfaces in the model (Surface1,Surface2...) for GOCAD object selection. Parameter PROPERTIES contains string with names of properties corresponding to quantities which may be used to control the colour of the surface, starting with the fourth numerical column of file VRTX. The names are separated by spaces. NAME contains a string with GOCAD name of the surface.

```
# Adding model box wireframe
  LIN='len-bnd.out'
  KOLLIN=0  R=1.00  G=1.00  B=1.00
  NAME='ModelBox'  PROPERTIES=' '  # just for GOCAD
  bndlin:
  linwrl:
```

In the above part of the history file, the edges of the model box are added. Program bndlin writes edges of the model box to the file specified by parameter LIN. Program linwrl converts the lines into the VRML or GOCAD representation, with colour specified by parameters R,G,B.

The history file continues with building P-wave velocity model composed of individual velocity sections (see Figure 1, right-hand picture).

```
# Solid P-wave velocity model
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Setting lights
  DLIGHT='len-dli2.dat'
  R=0.00  G=0.00  B=0.00  # Resetting black background

# Name of output file
  WRL='len-vp.wrl' # VRML
 #WRL='len-vp.mx'  # GOCAD
  iniwrl:

# Colours correspond to P-wave velocities
  COLUMN07='ISRF'  COLUMN08='+VP'
  KOLSRF=8  VPER=-6  VREF=7  CREF=0.
  PROPERTIES='N1 N2 N3 ISRF +VP'  # just for GOCAD

# Triangulating individual velocity sections
```

```
# Section perpendicular to X1, minimum X1 coordinate, part 1
  N1=1   N2=2   N3=16  D1=1.   D2=20.  D3=2.   O1=0.   O2=0.   O3=-25.
  NAME='Section1Min1'  # just for GOCAD
  modsrf:
  trgl:
  srfwrl:
```

This part of the history file contains parameters and programs already discussed in this paper. The colours of the surfaces correspond to P-wave velocities stored in the eighth numerical column (parameter `KOLSRF`). Model is divided into 12 sections. Each section is defined by the corresponding planar grid. Parameter `N1=1` means that the section is perpendicular to axis X1. Only the part of the history file corresponding to the first velocity section is shown here (others are similar).

Last part of the history file `len-mod.h` creates 3-D P-wave velocity cube (GOCAD Voxet) in the model with a lenticular inclusion and is prepared only for GOCAD (see Figure 2, left-hand picture) because VRML lacks this possibility. As the distributed version of the history file generates the VRML format of output files, the executable files in this part of history file are commented.

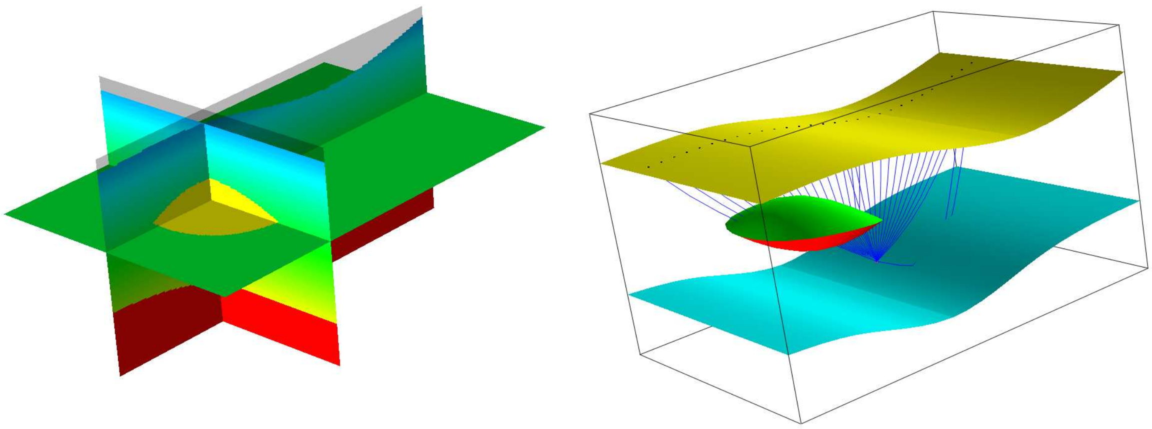

Figure 2. GOCAD Voxet P-wave velocity sections (left-hand picture) and interfaces with two-point rays (right-hand picture) in the model with a lenticular inclusion (history files `len-mod.h` and `len-crt.h`).

```
# 3-D P-wave velocity cube (GOCAD Voxet), only GOCAD
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Name of output file
  WRL='len-vpv.mx'  # GOCAD
 #iniwrl:

# Grid dimensions, initial position
  N1=151  N2=101  N3=76  D1=0.4  D2=0.4  D3=0.4  # sparser grid
# N1=301  N2=201  N3=151 D1=0.2  D2=0.2  D3=0.2  # minimum MRAM=9500000
  O1=0    O2=-20  O3=-25
```

```
 VEL='len-vpv.out'
#grid:  # Gridding velocities
 GRD='len-vpv.out' IN='len-vpv.bin'
#ascbin:
#swap:  # Gridded data for GOCAD must be big-endian (workstations)
        # Comment the above line on big-endian computers (workstations)
 NAME='VelocityCube'                 # GOCAD object (and property) name
 PROPERTY=                           # GOCAD property name equal to NAME
 NODATA=0. VPER=-6  VREF=7  CREF=0.  # Colour scale
#grdwrl:
```

There are two different lines with grid values. The line with sparser grid works with the default value of MRAM=4000000 in file ram.inc (suitable for 16MB computer) that determines approximately memory requirements of executable programs. The line with twice greater number of gridpoints in each direction needs increment of MRAM and recompilation of executable programs. Program grid discretizes P-wave velocities at gridpoints of a regular rectangular grid in the model. Program ascbin converts gridded data (data cubes) from formatted ASCII file len-vpv.out to direct-access binary file len-vpv.bin (this file is called from len-vpv.mx). Program swap converts binary gridded data (data cubes) between little-endian (PC's) and big-endian (VAX, Alpha, RISC workstations) hardware because GOCAD works only with big-endian data (we use PC). The command to run program swap should be commented on the big-endian computers! The last program of this part is grdwrl that converts gridded data into GOCAD representation. Parameter NODATA is used here to define grid points situated in the free space where the P wave velocity has value 0. ('property no data value' in GOCAD). This parameter should not be specified if there are undefined grid values. Figure 2, left-hand picture, uses partial transparency for the no data value. GOCAD visualization program has useful display tools for the Voxet object. It is possible to create many perpendicular sections in the model, run section animation, etc. The default positions of sections are set by program grdwrl.for for the minimal and maximal axis values of the model, i.e. you will see a model cube.

Figure 2, right-hand picture, shows rays, receivers and a source added to the model with a lenticular inclusion. The picture is created by the part of history file len-crt.h shown below.

```
# Adding two-point rays into the VRML or GOCAD file with surfaces
  VRML='VRML2'  WRL='len-srf.wrl'  WRLOUT='len-ray.wrl'  # VRML
 #VRML='GOCAD'  WRL='len-srf.mx'   WRLOUT='len-ray.mx'   # GOCAD
  LIN='len-ray.out'
# Setting blue colour for rays
  KOLLIN=0  R=0.00  G=0.00  B=1.00
  NAME='Rays'  PROPERTIES=' '  # just for GOCAD
# Example of parameters for colouring rays according to travel time
# KOLLIN=4  COLORS='hsv.dat'  VPER=6.0  VREF=0.0  CREF=0.166667
# NAME='Rays'  PROPERTIES='T'  # just for GOCAD
  linwrl:
```

```
# Adding source and receivers into the VRML or GOCAD file
  WRL='len-ray.wrl'  # VRML
 #WRL='len-ray.mx'   # GOCAD
  PTS='len-src.dat'
  KOLPTS=0  R=1.00  G=0.00  B=1.00
  NAME='Source'  PROPERTIES=' '  # just for GOCAD
  ptswrl:
  PTS='len-rec.out'
  KOLPTS=0  R=1.00  G=1.00  B=1.00  SHIFT3=0.001
  NAME='Receivers'  PROPERTIES=' '  # just for GOCAD
  ptswrl:
```

Parameter `WRL` specifies the name of the input VRML or GOCAD file with surfaces, and `WRLOUT` gives the name of the output file for the program `linwrl`. The next use of parameter `WRL` specifies the name of input and output file for program `ptswrl`. The colour of rays is blue. There is also commented possibility that rays are coloured according to travel time located in the 4th numerical column (`KOLLIN=4`) of the file given by parameter `LIN`. Parameter `SHIFT3` shifts all points situated at the surface to make them visible (increases value of coordinate X3).

5.2 Selected models

Figures in this part of the paper show snapshots of selected models, created by history files: `98-mod.h`, `98-ss.h`, `elf1-mod.h`, `l7-mod.h`, `sd1-mod.h`, `sd2-mod.h`, `wb2-mod.h`, `sal-wrl.h` (all are included on SW3D-CD-5). Computation and visualization were performed on a PC with RedHat Linux 7.0 and GOCAD 1.5.8. We hope that the history files are computer system independent and will run on other systems with the same output. The quality of snapshot bitmaps is limited by the resolution of the GOCAD window. Each figure caption includes the name of history file `*.h` that was used for computation. The positions of 3-D GOCAD camera (viewpoint) have been chosen manually. Only two default lights have been used.

Here is a short description of the models that have been used for various tests of SW3D software and are included in package DATA.

Model 98 is a sample synthetic model created during the technical session of the SW3D consortium meeting in June 1998. The model 98 is composed of three layers separated by two interfaces. The upper interface is curved, the lower one is formed by dipping plane. A salt body in the middle layer is bounded by a spherical interface (Figures 3,4,5).

Model ELF1 (MI) contains homogeneous layers separated by dipping plane interfaces (Figure 6).

Model L7, provided by the Institut Francais du Pétrole, is a model of real geological structure. Present version contains only two of seven structural interfaces (Figure 7).

Models SD1 and SD2 are sample synthetic models of a salt dome (Figures 8,9).

Model WB2 is Western Bohemia a priori model prepared as a test model resembling the conditions of refraction seismic measurements performed in Western Bohemia (Figure 10).

SAL is the 3-D SEG/EAGE Salt Model smoothed for ray tracing (Figure 11).

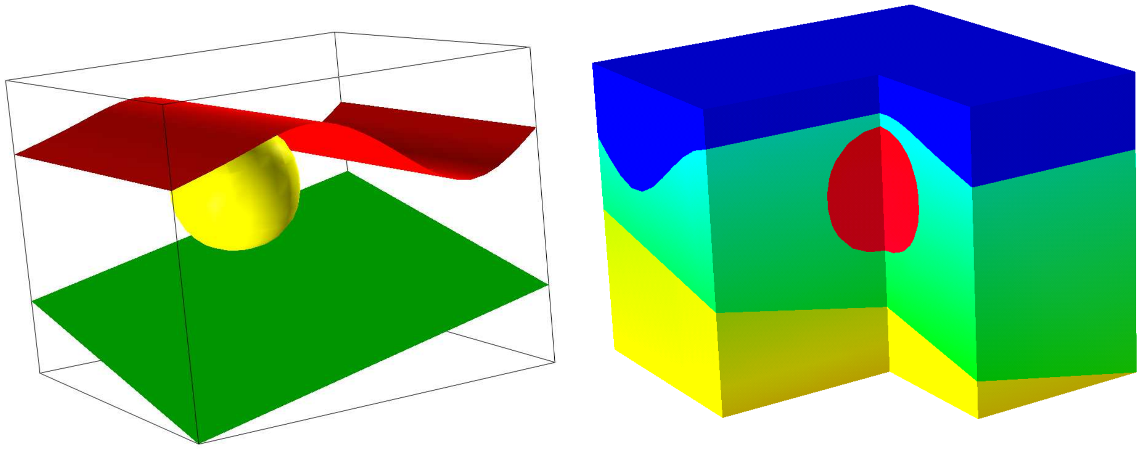

Figure 3. Interfaces coloured according to the indices of surfaces (left-hand picture) and P-wave velocity sections (right-hand picture) in the model 98 (`98-mod.h`).

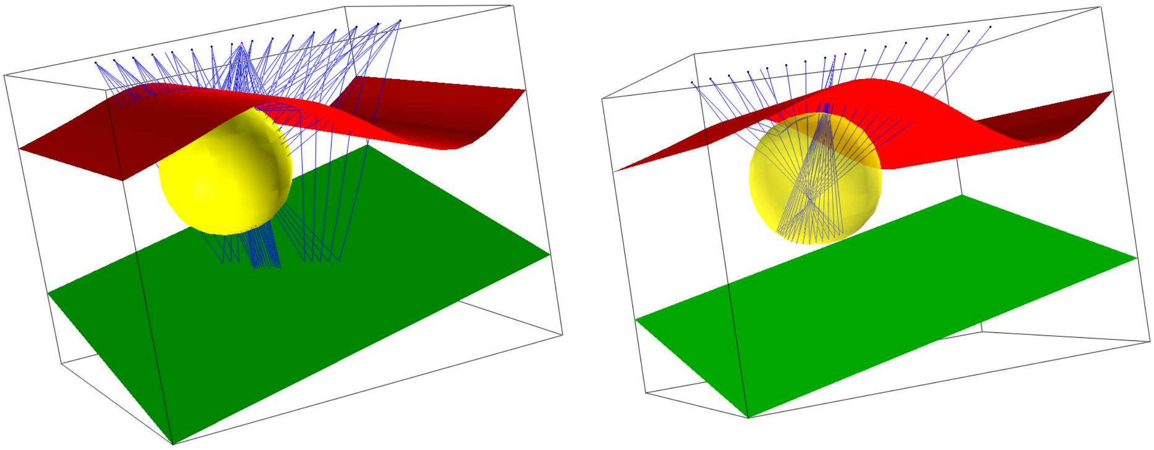

Figure 4. Two-point rays in the model 98 (`98-ss.h`). The spherical interface in right-hand picture is partially transparent.

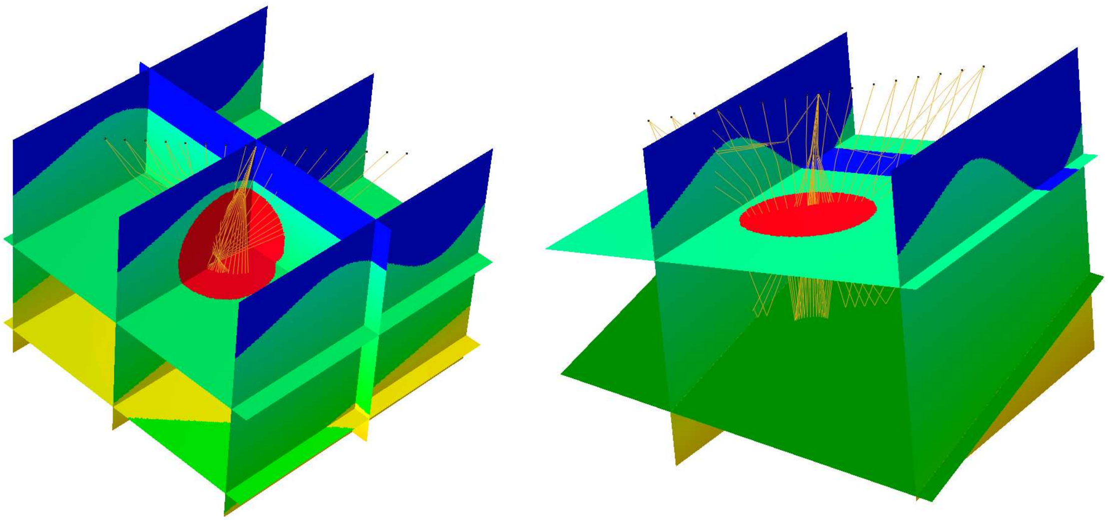

Figure 5. GOCAD Voxet P-wave velocity sections of the model 98 with the two-point rays (`98-mod.h` and `98-ss.h`). These pictures show the possibility of the GOCAD to combine objects from various files.

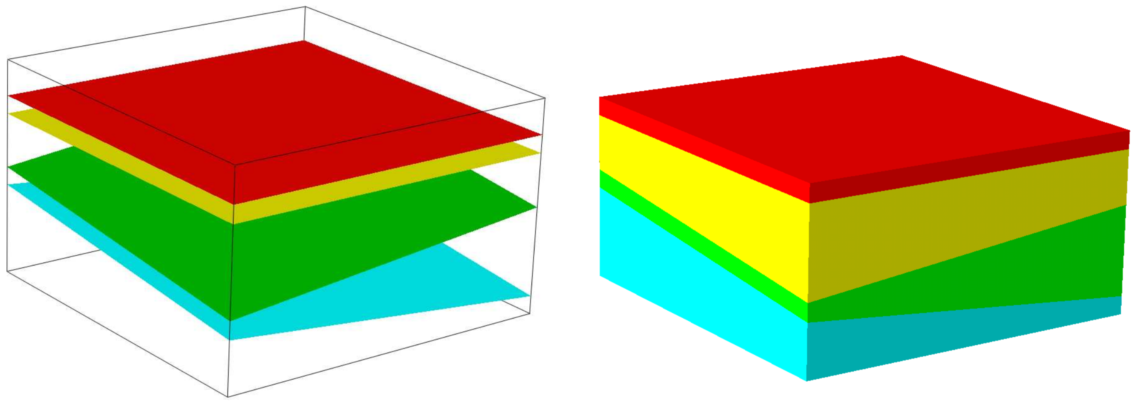

Figure 6. Interfaces coloured according to the indices of surfaces (left-hand picture), and sections coloured according to the indices of complex blocks (right-hand picture) in the model ELF1 (`elf1-mod.h`).

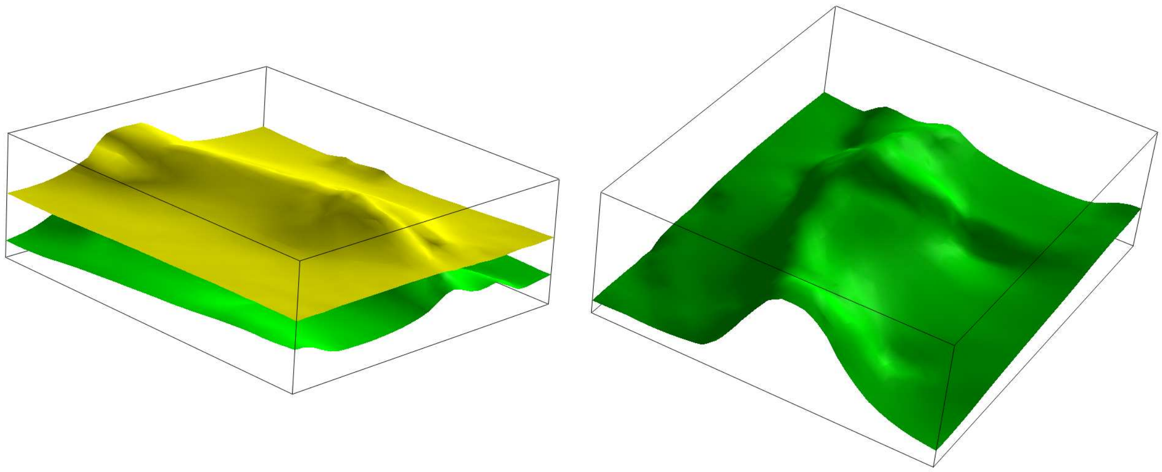

Figure 7. Two interfaces coloured according to the indices of surfaces in the model L7 (`l7-mod.h`).

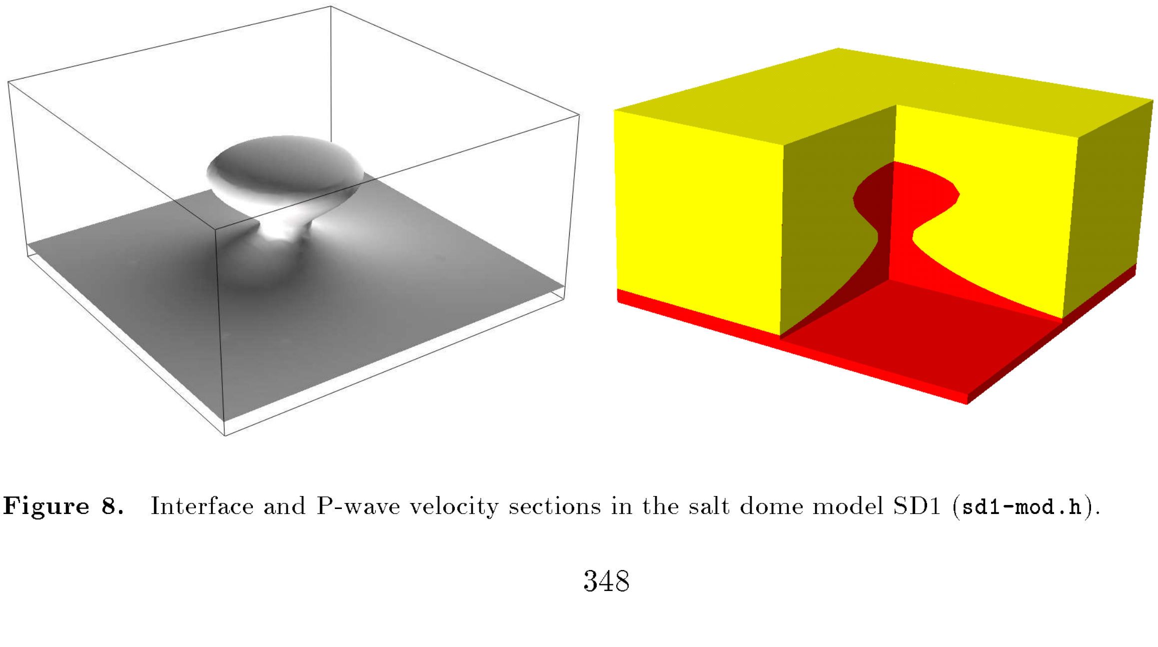

Figure 8. Interface and P-wave velocity sections in the salt dome model SD1 (`sd1-mod.h`).

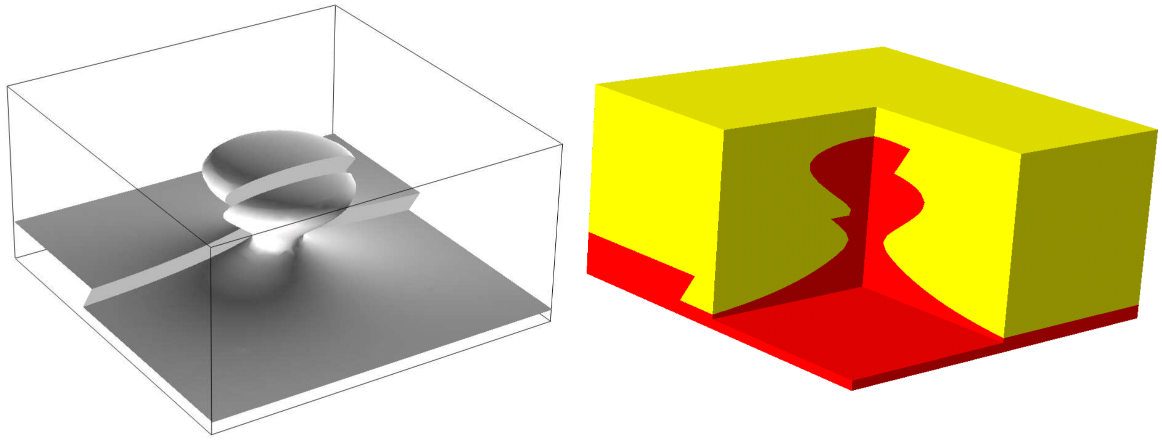

Figure 9. Interface and P-wave velocity sections in the dislocated salt dome model SD2 (`sd2-mod.h`).

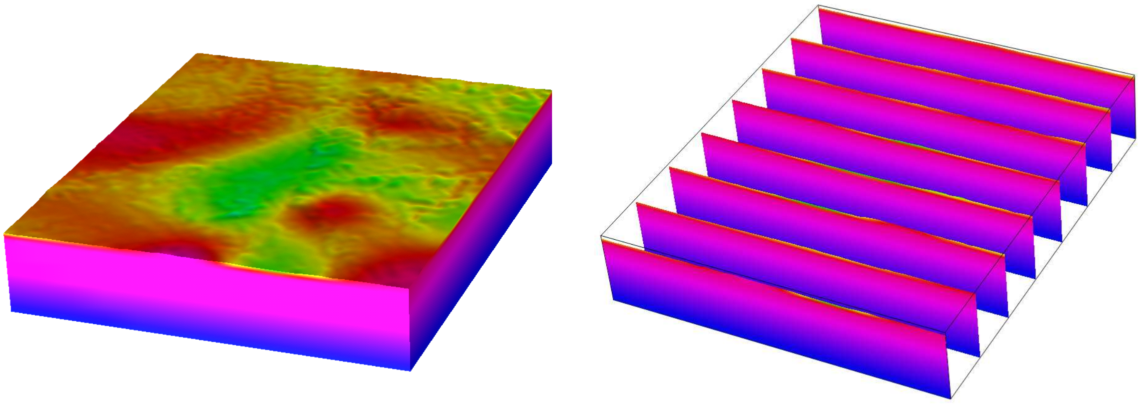

Figure 10. P-wave velocity sections in the Western Bohemia model WB2 (`wb2-mod.h`).

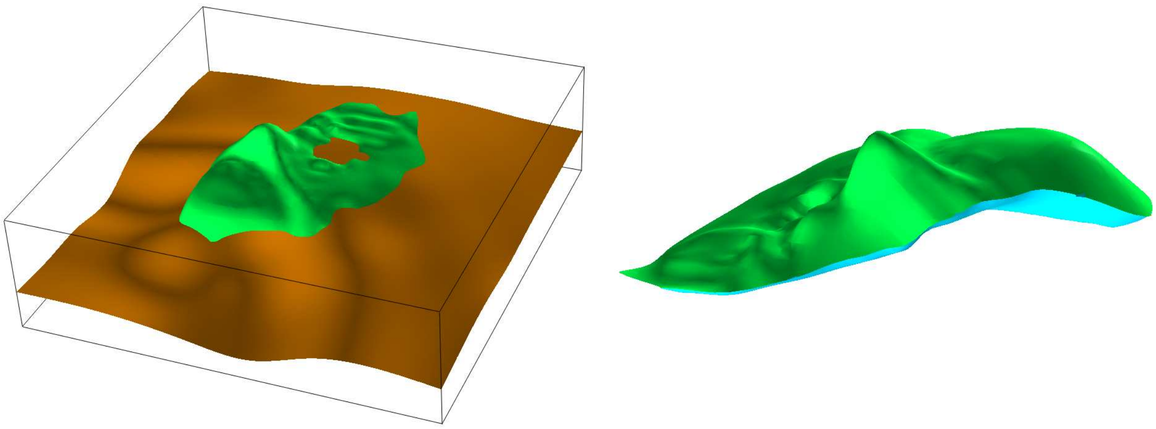

Figure 11. Interfaces in the SEG/EAGE Salt model described in the form suitable for the SW3D packages (`sal-wrl.h`).

A. Appendix

Text printed in *italics* contains simplified variables and constants used by SW3D Fortran77 programs. For details see source codes.

A1. VRML definitions and remarks

A1.1 Header, cameras, lights

All nodes in this part are set by program `iniwrl.for`.

Header

```
#VRML V2.0 utf8
```

Set cameras (viewpoints)

```
Viewpoint  }
  description    TEXT
  fieldOfView    AUX
  position       X1 X2 X3
  orientation    R1 R2 R3 R4
}
```

Turn off headlight

```
NavigationInfo {
  headlight   FALSE
}
```

Set ambient light

```
DirectionalLight {
  color              1.00 1.00 1.00
  intensity          0.00
  ambientIntensity   W
  on                 TRUE
}
```

Set directional light

```
DirectionalLight {
  direction   X1 X2 X3
}
```

Set point light

```
PointLight {
  location           X1 X2 X3
  radius             999999
  color              1.00 1.00 1.00
  intensity          ABS(W)
  ambientIntensity   0.00
  on                 TRUE
  on                 FALSE
}
```

Set background colour

```
Background {skyColor  RED GREEN BLUE}
```

A1.2 Points

VRML prototypes are set by `iniwrl.for`. Other nodes are set by `ptswrl.for`.

Prototype Point

```
PROTO Point [
  exposedField SFNode  appearance NULL
  field        MFVec3f point      [ ]
  exposedField SFNode  color      NULL
]{
  Transform {
    children [
      Shape {
        appearance IS appearance
        geometry PointSet {
          coord Coordinate {
            point    IS point
          }
          color      IS color
        }
      }
    ]
  }
}
```

Coordinates and colours of points

```
Point {
appearance USE PointAppearance
point [
X1(I) X2(I) X3(I)
]
color Color { color [
RED GREEN BLUE
] }
}
```

One colour for all points

```
Shape {
  appearance DEF
            PointAppearance Appearance {
    material Material {
      emissiveColor   RED GREEN BLUE
    }
  }
}
```

A1.3 Lines

VRML prototypes are set by `iniwrl.for`. Other nodes are set by `linwrl.for`.

Prototype Line

```
PROTO Line [
  exposedField SFNode  appearance NULL
  field        MFVec3f point      [ ]
  exposedField SFNode  color      NULL
  field        MFInt32 coordIndex [ ]
]{
  Transform {
    children [
      Shape {
        appearance IS appearance
        geometry IndexedLineSet {
          coord Coordinate {
            point    IS point
          }
          color      IS color
          coordIndex IS coordIndex
        }
      }
    ]
  }
}
```

Coordinates and colours of line points

```
Line {
appearance USE LineAppearance
point [
X1(I) X2(I) X3(I)
]
color Color { color [
RED GREEN BLUE
] }
coordIndex [
I1 ...
]
}
```

One colour for all line points

```
Shape {
  appearance DEF
            PointAppearance Appearance {
    material Material {
      emissiveColor    RED GREEN BLUE
    }
  }
}
```

A1.4 Surfaces

VRML prototypes are set by `iniwrl.for`. Other nodes are set by `srfwrl.for`.

Prototype Surface

```
PROTO Surface [
  exposedField SFNode  appearance NULL
  field        MFVec3f point      [ ]
  exposedField SFNode  normalPos  NULL
  exposedField SFNode  normalNeg  NULL
  exposedField SFNode  colorPos   NULL
  exposedField SFNode  colorNeg   NULL
  field        MFInt32 coordIndex [ ]
]{
  Transform {
    children [
      Shape {
        appearance IS appearance
        geometry IndexedFaceSet {
          ccw TRUE #positive surface side
          coord DEF
                SurfaceCoord Coordinate {
            point    IS point
          }
          normal     IS normalPos
          color      IS colorPos
          coordIndex IS coordIndex
        }
      }
      Shape {
        appearance IS appearance
        geometry IndexedFaceSet {
          ccw FALSE #negative surface side
          coord USE SurfaceCoord
          normal     IS normalNeg
          color      IS colorNeg
          coordIndex IS coordIndex
        }
      }
    ]
  }
}
```

Material properties of the surface

```
Shape {
  appearance DEF SurfaceAppearance Appearance {
    material Material {
      diffuseColor      RED  GREEN BLUE
      specularColor     SPEC SPEC  SPEC
      shininess         SHIN
      transparency      TRANSP
      ambientIntensity  1.00
      emissiveColor     0.00 0.00 0.00
    }
  }
}
```

Vertices, normals, colours and polygons of the surface

```
Surface {
appearance USE SurfaceAppearance
point [
X1(I) X2(I) X3(I)
]
normalPos Normal { vector [
# right-handed normals (positive surface side)
N1(I) N2(I) N3(I)
] }
normalNeg Normal { vector [
# left-handed normals (negative surface side)
-N1(I) -N2(I) -N3(I)
] }
colorPos DEF SurfaceColor Color { color [
# colours of the positive surface side
RED GREEN BLUE
] }
colorNeg USE SurfaceColor
# colours of the negative surface side
colorNeg Color { color [
RED GREEN BLUE
] }
coordIndex [
# polygons (usually triangles)
I1 I2 I3 ...
]
}
```

A1.5 Vrmlview browser problems

We found following serious problems when running this browser:
- ambient light makes all objects white, so it cannot be used,
- transparency does not work for overlapping transparent objects,
- text string rotates like a separate letters not one string,
- limited user interaction and animation support,
- transparency is not supported for lines or points,
- it is impossible to switch on all lights defined in VRML file at once.

A2. GOCAD definitions and remarks

A2.1 File type, points

File type is written by `iniwrl.for`. Object `VSet` and its attributes are written by `ptswrl.for`.

File type
```
#GOCAD format (generated by SW3D package
FORMS)
```

Points
```
GOCAD VSet
HDR name:                NAME
HDR *visible:            true
HDR *atoms*symbol:       point
HDR *atoms*size:         3
HDR *atoms*color:        RED GREEN BLUE
HDR *painted:            true
HDR *painted*variable:      TEXT
PROPERTIES                  TEXT
PROPERTY_CLASSES            TEXT
PROPERTY_CLASS_HEADER       TEXT  {
   *colormap:      NAME
   *low_clip:      OUTMIN
   *high_clip:     OUTMAX
   *colormap*NAME*colors:   \
     I R G B  \
     I R G B
}
VRTX     I0 ...
PVRTX    I0 ...
END
```

A2.2 Lines

Object `PLine` and its attributes are written by `linwrl.for`.

Lines
```
GOCAD PLine
HDR name:                NAME
HDR *atoms:              false
HDR *visible:            true
HDR *line*color:         RED GREEN BLUE
HDR *painted:            true
HDR *painted*variable:      TEXT
PROPERTIES                  TEXT
PROPERTY_CLASSES            TEXT
PROPERTY_CLASS_HEADER       TEXT  {
   *colormap:      NAME
   *low_clip:      OUTMIN
   *high_clip:     OUTMAX
   *colormap*NAME*colors:   \
     I R G B  \
     I R G B
}
```

Lines - continues
```
ILINE
VRTX    IVRTX ...
PVRTX   IVRTX ...
SEG     I1 ...
END
```

A2.3 Surfaces

Object `TSurf` and its attributes are written by program `srfwrl.for`.

Surfaces
```
GOCAD TSurf
HDR name:                    NAME
HDR *visible:                true
HDR color:                   RED GREEN BLUE
HDR *solid*transparency:       TRANSP
HDR *painted:                  true
HDR *shaded:                   true
HDR *precise_painted:          true
HDR *painted*variable:         PROP
PROPERTIES                     PROP
PROPERTY_CLASSES               TEXT
PROPERTY_CLASS_HEADER          NAME PROP {
   *colormap:      NAME
   *low_clip:      OUTMIN
   *high_clip:     OUTMAX
   *colormap*NAME*colors:   \
     I R G B  \
     I R G B
   *colormap*alphas:  \
     I TRANSP  \
     I TRANSP
}
 VRTX    I0 ...
 PVRTX   I0 ...
 TRGL    I1 ...
END
```

A2.4 Voxets

Object `Voxet` and its attributes are written by program `grdwrl.for`.

Voxets
```
GOCAD Voxet
HDR name:                  NAME
HDR *TEXT*cage:            false
HDR *TEXT*shaded:          true
HDR *TEXT*precise:         true
HDR *TEXT*smoothed:        true
HDR *TEXT*sections:        6  1 1 0  1 1 N1-1
                              2 1 0  2 1 N2-1
                              3 1 0  3 1 N3-1
```

Voxet - continues

```
AXIS_O        O1   O2   O3
AXIS_U        D1   0.   0.
AXIS_V        0.   D2   0.
AXIS_W        0.   0.   D3
AXIS_MIN      0    0    0
AXIS_MAX      N1-1 N2-1 N3-1
AXIS_N        N1   N2   N3
PROPERTY 1              TEXT
PROPERTY_CLASS 1        TEXT
PROPERTY_CLASS_HEADER   TEXT {
   *colormap:           NAME
   *colormap*nodata:    true
   *colormap*ndtransparency:  0.80
   *low_clip:     OUTMIN
   *high_clip:    OUTMAX
   *colormap*NAME*colors:  \
     I R G B  \
     I R G B
   *colormap*alphas:  \
     I TRANSP  \
     I TRANSP
}
PROP_FORMAT 1 RAW
PROP_ETYPE 1 IEEE
PROP_ESIZE 1 4
PROP_NO_DATA_VALUE 1  AUX
PROP_FILE 1  FBIN
END
```

A2.5 GOCAD 1.5.8 problems

A2.5.1 Linux version

We found following serious problems when running the program on Linux PC:
- precise painted switched on (true) for surfaces causes a strange colors and textures on the surface,
- smoothed switched on (true) for voxet causes bad perspective and colors of the displayed voxet.

The GOCAD support team explained both problems as OpenGL bugs. OpenGL is not yet very well supported on Linux (we use Mesa 3-D graphics library v.3.0).

A2.5.2 General

- there is no complete description of GOCAD object attributes,
- it is impossible to define cameras and lights directly in a GOCAD input file.

Acknowledgments

The author thanks to Luděk Klimeš and Petr Bulant for valuable comments and recommendations concerning this paper. The research has been supported by the Consortium "Seismic Waves in Complex 3-D Structures".

References

ASGA (Association Scientifique pour la Geologie et ses Applications) (1997): GOCAD 1.4 User's Manual. Nancy, France.

ASGA (Association Scientifique pour la Geologie et ses Applications) (2000): GOCAD++ Developer's Guide version 1.5. Nancy, France.

Bucha, V., Bulant, P. & Klimeš, L. (eds.) (1999): SW3D-CD-3. In: Seismic Waves in Complex 3-D Structures, Report 8, pp. 193, Dep. Geophys., Charles Univ., Prague.

Bucha, V., Bulant, P. & Klimeš, L. (eds.) (2001): SW3D-CD-5. In: Seismic Waves in Complex 3-D Structures, Report 11, pp. 357, Dep. Geophys., Charles Univ., Prague.

Bulant, P. (1999): Triangulation of structural interfaces. In: Seismic Waves in Complex 3-D Structures, Report 8, pp. 17–26, Dep. Geophys., Charles Univ., Prague.

ISO/IEC 14772-1 (1997): VRML97. The Virtual Reality Modeling Language. The VRML Consortium, Inc., "http://www.vrml.org/Specifications/VRML97/index.html".