RoadMap: display of surface properties of a macromolecule.

Michael Chapman*

24 September 1991
Revision: 15 May 1992
Printed October 19, 1993

Abstract

RoadMap can use a variety of projections to map the van der Waal's or solvent accessible surface of a macromolecule onto a plane. It creates a PostScript file showing the boundaries of each residue that can be printed or displayed on a workstation with a PostScript interpreter. Onto the map the user can overlay properties such as the surface topography, sequence conservation, hydrophobicity, atomic solvation or any other user–defined parameter, using a combination of contoring, black–and–white shading, and coloring. The map can be annotated.

*Institute of Molecular Biophysics, Florida State University, Tallahassee, FL 32306.

License Agreement for the Use of Roadmap

Roadmap will be installed at:

Department ______________________________

Name of Institution ______________________________

Street address ______________________________

City ______________________________

State, ZIP, Country ______________________________

Principal Investigator ______________________________

Contact person (e.g. system manager) ______________________________

E–mail address and Telephone ______________________________

Computer ______________________________

Operating System ______________________________

Tape format/density ______________________________

I understand that Purdue Reseach Foundation will retain the copyright for the software. Use is restricted to the site named above for the non–commercial purposes of non–profit research and the illustration of results. Separate written authorization must be obtained for:

- use for research by for–profit institutions.
- use as a service for those outside the group.
- inclusion in work for which payment or royalties can reasonably be expected.
- permission to copy or modify, source code or subprogram libraries.

I understand that copies of the program and the documentation may be made for internal use, but may not be distributed outside my research group, and I will take precautions with file protection to ensure that the copyright is not infringed. I understand that neither Purdue Research Foundation or Michael Chapman can accept liability in connection with the use of this program, and do not make any committment to install, service or maintain the software. I agree to update the program with new versions, and to delete all copies of the program and documentation upon the request of Michael Chapman or Purdue Research Foundation. I understand that such deletion may be requested if I do not agree to future changes in the license agreement.

I will acknowledge the use of this program in the publication of results.

Signature: ______________________ Date: ______________________
(Principal Investigator)

Please return to:

Prof. Michael S. Chapman,
Institute of Molecular Biophysics,
Florida State University,
Tallahassee,
FL 32306, USA.

FAX: (904) 561–1406

RoadMap

1 Related programs:

RoadMap extends and generalizes some of the most frequently used options of **VSurf**[1]. In particular, the "RoadMap" figures are produced automatically rather than by manual conversion of printable output.

Sequence conservation parameters can be calculated with **Similarity** [Chapman and Rossmann, *in preparation.*].

2 Citation and Related documents:

The capabilities of the program are illustrated in a separate paper, currently in preparation. This paper should be cited in publications where **Roadmap** is used. Its reference is "Mapping the Surface Properties of Macromolecules", Chapman, M.S., 1992, *in preparation.*

Program flow is controlled by an option–driver, and input is by tokens. Conventions and general explanations of use and syntax are given in the document *"Option Driver and Token Input/Output"*. The input of coordinates uses routines whose input it documented in *"ExpCoord"*. Generally, the defaults given in the documentation usually apply only to the first use of a variable. Thereafter the default will be the previous value. There are a few exceptions where this would obviously be inappropriate.

3 Description of options

Options are organized into a hierarchy of menus. Within each menu, each entry is either a command (option) or the name of another menu (in capitals). Various optional parameters may be specified after the option name.

Surface *must* be the first option called. New_page must be the second option called. Options that shade or color should be called before those that trace, contor or write text as they obliterate what is underneath.

SURFACE Calculates the projection of the surface, reads it from a file or saves it in a printable file.

COORDINATE_CONTROL [Add_Hydrogen [=] {logical}] [Add_Water [=] {logical}]

Coordinate_control reads input coordinates and expands them according to the molecular (and/or crystallographic) symmetry, and

[1]Rossmann and Palmenberg, 1988, Virology, <u>164</u>, 373–382

prepends a symmetry–equivalent designator to each residue name. All of the sub–options and their input are described in the document *ExpCoord*. (*FileOut* is not used.)

Coordinate_control also has two option–line parameters:

Add_Hydrogen: True – add 0.3 Å to van der Waal's distances to allow for hydrogens that have not been given explicitly in the coordinates. Default = .true..

Add_Water: True – add 1.4 Å to van der Waal's distances to convert from a van der Waal's surface to a solvent accessible surface. Default = .true..

Parallel_projection [Projection_vector [=] ({real})^3] [Origin [=] ({real})^3] [Horizontal_axis [=] ({real})^3] [Vertical_axis [=] ({real})^3] [Use_plane [=] {logical}] [Add_To_Distance [=] {real}] [Projection_direction [=] {logical}] [Distance_origin [=] ({real}^{3}] [Start_horizontal [=] {real}] [End_horizontal [=] {real}] [Space_horizontal [=] {real}] [Start_vertical [=] {real}] [End_vertical [=] {real}] [Space_vertical [=] {real}]

Projects the surface onto a plane using a set of vectors parallel to the direction of projection. This is the simplest projection, and is free from spacial distortion. It is most applicable when one wishes to examine a relatively small area, such as one asymmetric unit of a viral capsid.

The following vectors are all defined in coordinate space:

Projection_vector: a vector down which the surface is to be projected (normal to the plane). Default = (0.0,0.0,-1.0)

Origin: a 3–D positional vector specifying a point on a plane orthogonal to *Projection*. Over user–defined ranges, vectors parallel to *Projection* are translated and extended from this plane to their points of intersection with the molecule. The Origin should usually be chosen such that all of the plane would be outside the molecule. If option *Icosahedron* is to be used to outline an icosahedral asymmetric unit, then Origin needs to be on the z–axis. Default = (0.0,0.0,150.0)

Horizontal_axis: a vector whose projection onto the plane of projection will become the horizontal axis on the plot. Default = (0.0,1.0,0.0)
Vertical_axis: the vertical axis will be orthogonal to *Projection* and to *Horiontal_axis* with a direction such that its scalar product with

Vertical_axis is positive. Default = (1.0,0.0,0.0)

Use_plane: True – calculate the distance for topographical display as a height, namely the projection of a vector from *Origin* to each point of intersection upon *Projection*. False – calculate a radial distance from the point *Distance_origin*. Default = .false..

Add_To_Distance (relevant if use_plane = true): add this constant to all distances. It should normally be the absolute value of the projection upon *Projection* of a vector from *Origin* to some point of interest, such as the molecular center. The default is calculated in this way, assuming that the molecular center is at (0.0,0.0,0.0).

Projection_direction (relevant if use_plane = true): True – positive distances are in the same direction as *Projection*. Default = .false..

Distance_origin (relevant if use_plane = false): point from which radial distances are to be measured. Default = (0.0,0.0,0.0).

The following ranges are defined in the projected (Å) space to be plotted. If the projection is calculated by **RoadMap** then the horizontal and vertical axes map to user–defined vectors of arbitrary direction dependent on *Projection, Vertical_axis and Horizontal_axis.* (Only) when reading the projection from the output of **VSurf** does x neccessarily run from right to left and y from bottom to top. All limits are interpreted as defining the center of the relevant pixels.

Start_horizontal: left–most limit of surface projection. Default = -60.0.

End_horizontal: right–most limit of surface projection. Default = 60.0.

Space_horizontal: horizontal pixel spacing. Default = 2.0.

Start_vertical: left–most limit of surface projection. Default = -1.0.

End_vertical: left–most limit of surface projection. Default = 105.0.

Space_vertical: vertical pixel spacing. Default = 2.0.

Cylinder_projection [Vertical_axis [=] ({real})^3] [Center [=] ({real})^3] [Origin [=] ({real})^3] [Start_horizontal [=] {real}] [End_horizontal [=] {real}] [Space_horizontal [=] {real}] [Start_vertical [=] {real}] [End_vertical [=] {real}] [Space_vertical [=] {real}] [Spacing_radius

[=] {real}]

Projects the surface by mapping vectors that converge upon the origin upon a cylinder which is then developed (unfolded) into a flat plane. This projection is often used in atlases, is best for equatorial regions, and most appropriate to display an entire molecule.

Vertical_axis: this vector defines in atomic coordinates the "north pole" of the projection. Default = (1.0,0.0,0.0)

Center: defines a direction in atomic coordinates from which the ranges, StartH, EndV *etc.* are to be defined. Default = (0.0,1.0,0.0)

Origin - point to which all projection vectors converge. Default is calculated as the center of mass of atomic coordinates.

Spacing_radius: For plotting the Angstrom pixel–spacings are calculated from the angular spacings input and the average distance from the origin to the molecular surface. If one region is of particular interest, or if you want the same spacing as another plot, *spacing_radius* may be used instead of the average distance. To use, set *Spacing_radius* to a positive value. Default = -1.0.

All of the other parameters are as described as for *Parallel_projection*, with the exception that the ranges should be input as angles (degrees). Positive angles are defined according to the right–hand grip rule about *vertical_axis* and *center*. *There is an important difference between entering a range with a positive spacing and a negative one: if negtive then the mirror image about 0 of the original projection is returned.* This option is required, because with the freedom to choose *vertical_axis* and *center*, the projection have have to be converted back from a left–handed image. However, this option should be used with care. Defaults: StartH = -175.0; EndH = 175.0, SpaceH = 10.0, StartV = -85.0, EndV = 85.0, SpaceV = 10.0. These will be converted to Angstrom units after the projection has been calculated.

Pore_projection [Number_points [=] {integer}] [Guide_point [=] $((\{\text{real}\})^3)^n$] [Space_horizontal [=] {real}] [Vertical_axis [=] $(\{\text{real}\})^3$] [Right_hand_grip [=] {logical}] [Steps [=] {integer}] [Spacing_radius [=] {real}]

Maps surface residues and their distances by projecting radially from an axis that follows guide points. This is particularly applicable to inhibitor/coefactor pockets and pores.

Number_points: number of guide points to be entered. Must at least

2. No default.

Guide_point: These are 3–D position vectors that define the axis from which the molecular surface will be projected. In the simplest case, there will be just two points, and the surface will be projected onto a true cylinder. However, many pores and pockets are not straight. The user may enter a series of guide points for the axis to follow. This will result in a *distorted* projection. To minimize the distortion, ensure that the guide points follow a smooth curve. The guide points are entered $x_1, y_1, z_1, x_2, y_2, z_2 \ldots$.

Space_horizontal: The step–size along the projection axis, starting half a step from the first grid point. Default = 2.0.

Vertical_axis: The projection axis is always displayed horizontally. Vertical axis is a direction vector in atomic coordinate space that when projected onto the plane normal to the projection axis defines where the cylinder is to be opened up for flat projection. Default = (1.0,0.0,0.0).

Right_hand_grip: If true (default), then the projection vector is rotated about the cylindrical axis according to the right–hand grip rule, and the first regions to be projected are mapped to the top of the map near *EndH*. This can be reversed by setting Right_hand_grip to false.

Steps: The number of steps into which the 360 degrees of rotation is split. Default = 36.

Spacing_radius: For plotting, a vertical pixel–spacing is calculated from *Steps* and the average distance from the cylindrical axis to the moleular surface. If one region is of particular interest, or if you want the same spacing as another plot, *spacing_radius* may be used instead of the average distance. To use, set *Spacing_radius* to a positive value. Default = -1.0.

The other parameters are described under option *Parallel_projection*.

Printable_output [File [=] {string}] [Invert [=] {logical}]
Make an ASCII, printable file that could be edited and read in again with *Read_VSurf*.

File: the file name. Default = roadmap.printable.

Invert: True – write with the horizontal axis *decreasing* from left to right to be compatible with **VSurf** output.

Read_VSurf [File [=] {string}] [Start_horizontal [=] {real}] [End_horizontal [=] {real}] [Space_horizontal [=] {real}] [Start_vertical [=] {real}] [End_vertical [=] {real}] [Space_vertical [=] {real}]

File: the file name. This can either be the output of *Printable_output* or of **VSurf** (uncompressed). *Read_VSurf* will search through the file for the header to the projected surface table. Default = vsurf.output.

The other parameters are described under option *Parallel_projection*: they must match those used in the input file.

New_page [PostScript File [=] {string}] [Header [=] [']{string}[']] [Scale [=] {real}] [Origin [=] {real},{real}]
If this is the first call – open the output file; else issue a page break.

Header: placed on top left of the map. Default = 'Map of surface residues'.

Scale: inches/Å. The default is calculated to maximize the use of an 11 x 8.5 page, leaving a suitable margin. The defaults should suffice unless you want to use exactly the same scale/origin as another plot. The values used are echoed to standard output.

Origin: The position on the paper of (0,0) (horizontal, vertical) projected coordinates in inches from bottom left. The default is calculated to maximize the use of an 11 x 8.5 page, leaving a suitable margin.

Trace_residues [Width [=] {real}] [Dash [=] {real,real}]
Draw an outline around each residue, and label the central pixel with the residue type and name. This option will be required almost every run. However, it should be called after all coloring and shading, so that the latter do not obscure the text and lines. This option is currently optimised for three letter residue names and types, with longer strings having to be squeezed within one pixel area. Width is a multiplication factor to increase the program's default line–width of within this option (default 1.0). Dash controls the amount of black and white in an optionally dashed line. The input values are in inches with the defaults of 0.0 & 0.0 resulting in a solid line. Values of 0.005 and 0.02 generate a nice dotted line on our 300dpi printer, but lower resolution devices would need coarser dashes.

Pixel_properties

(See also *Residue_properties*.) Coloring should be done prior to any line–drawing (*eg. Trace_residues*) as the painting will obscure anything behind the filled area.

Property [Number [=] {integer}] [[File [=] '{string}'] [Default [=] {real}]] [Save [=] '{string}']] [Restore]

At most 3 residue/pixel properties may be *simultaneously* required for coloring, shading or contoring. This option loads them. Number should be between 1 and 3. File may either be the name of a file from which properties to be read, or a special command. If File is "height" (default), then the height of each pixel is calculated from the atomic coordinates and no further file is needed. If File is "atomic_property", then the value of each pixel is taken from the "weight" column (next after z–coordinate) from the co–ordinate file. In this way, atom–specific properties, such as atomic accessibility [2] can be plotted. Again no further file is needed. If File is anything, but "height" or "atomic_property", it is interpreted as a file–name giving the values to be plotted. Several sorts of file can be used:

1. Residue/atom type file: a property value is to be assigned according to the type of residue and type of atom. Each line of the file should contain residue–type, atom–type, value as:
   [']{string}[⋆][''] [']{string}[⋆][''] {real}
   Quotes are needed if white–spaces are included in the strings. Wild–cards can be used as in "xyz⋆" to mean any residue or atom beginning with the characters "xyz". "xyz" can be "" to indicate *any* residue or atom. Ambiguities resulting from the use of wild–cards are decided by selecting the atom within a residue with greatest number of matching characters, and the residue with greatest number of matching characters for which there is any atom that matches. The following example is distributed with the program in *atomic.solvations* and gives atomic solvation parameters adapted from Eisenberg and McLachlan[3]:

   ```
   ⋆ C⋆ 16 ! cal A-2 mol-1
   ⋆ N⋆ -6
   ⋆ O⋆ -6
   ⋆ S⋆ 21
   ARG NH⋆ -28 ! average of -6 and -50 for 2 NH's
   ASP OD⋆ -15 ! average of -6 and -24 for 2 OD's
   GLU OE⋆ -15 ! average of -6 and -24 for 2 OD's
   HIS NE⋆ -50
   LYS NZ⋆ -50
   ```

2. Residue–name file: If the first record of the file does not contain a residue name, an atom name and a value, then the file will be assumed to be of a second type which should contain on each line

[2] See program **Accessibility**: note that the residue accessibility can also be plotted with *Residue_properties*.

[3] Nature 317, 199–203, 1986

a residue name (its number) and its property (a real number):
['[']{string}[$\star$]['] {real}

Quotes are needed if white–spaces are included in the strings. Eg:
158 0.135455
159 0.467273
... etc.

It can take several minutes to associate each residue with all relevant pixels. Repetition of this, for the same property list can be avoided by adding *Save* = {*save_file*} to the input (default = roadmap.pixel_properties).

3. If "Restore" is on the input line, then the input file will be expected to be of the type saved previously with *Save*. It should contain a list of horizontal and vertical pixel coordinates followed by the property value. The parameter "Restore" takes precedence over "Save".

If a file is specified, the Default keyword may be used to assign values to residues that are not listed. The keyword, Default, with an empty token should be entered to use the default "Default" (0.0), as the absence of the keyword indicates that the values of unlisted residues are not to be changed. This is useful when overlaying mutually exclusive lists of residues, on the 2nd and subsequent calls to *Property*.

Set_color [Number [=] {integer}] [[Parameter [=] {string}] [Property_range [=] {real} [[,] {real}]] [Parameter_range [=] {real} [[,] {real}]] [Key_title [=] [']{string}[']]
In preparation for painting, this sets one of the color parameters to be a linear function of one of the three properties. Parameter should be either clear, hue, saturation or brightness. If "clear", then all color parameters are reset to their defaults. If "B&W", then the parameters are set for the blank–and–white shading of a single property. Otherwise the color parameter requested is set according to the value of the property for the pixel and according to the mapping of the parameter range to the property range. If the color parameter falls outside the range, then it is truncated to the limit. Although usually ranges will be specified with the 2nd value greater than the first, this is not necessary. "Hue" defines the color on a circle with 0 = red, 0.333 = green, 0.666 = blue and 1.0 = red. "Saturation" defines the amount of color *vs.* grey–shading from 0.0 = grey to 1.0 = full color. "Brightness" defines the overall intensity with 0.0 = black and 1.0 = full intensity. For full documentation consult the PostScript Language Reference Manual [4]. Key_title is the text used for a header to the key.

[4] Adobe Systems Inc., Addison–Wesley, Reading, Mass., 1986.

Defaults: Number = 1: Parameter = Saturation. Only a few of the range parameter defaults are of general use. For "Hue": Parameter_range = 0.0,0.6666,(red to blue through green), Property_range = -0.5,1.2 (suitable for similarity scores, showing most conserved in blue). For "Saturation": Parameter_range = 0.4,1.0, Property_range = 110.0,138.0 (together suitable for the color–shading of depressed areas on a viral surface, with depressed areas having less color). For "Brightness": Parameter_range = 1.0,0.4, Property_range = 110.0,138.0 (together suitable for the grey–shading of depressed areas on a viral surface, with depressed areas shaded darker). Key_title = 'Shade: Property:'.

Paint
Does the coloring.

RESIDUE_PROPERTIES (See also *Pixel_properties.*) Coloring should be done prior to any line–drawing (*eg. Trace_residues*) as the painting will obscure anything behind the filled area. *Residue_properties* has the same input and sub–options as *Pixel_properties*, except for a slight difference in sub–option *Property. Pixel_properties* is needed to represent any function changing pixel–by–pixel. If this is not required, *Residue_properties* generates a prettier image, because the coloring will match exactly the lines drawn by option *Trace_residues.*

Property [Number [=] {integer}] [[File [=] '{string}'] [Default [=] {real}]] [Reset]
Parameters *Number, File & Default* are described in *Pixel_properties – Property. Property* within *Residue_properties* differs slightly to allow the user to read different properties from different files. Thus for example, the user may highlight a few hydrophobic residues *and* a those of highest conservation at the same time. The program keeps track of which residues have been colored previously, and *Default* only affects residues that have not already been colored, unless a new parameter *Reset* is also given. As a translation from residues to pixels is not required, the commands *Save* and *Restore* are not required. Within *Residue_properties*, the atom–type is irrelevant, so if a residue/atom–type input file is used, all of the atoms should be set to ⋆. The following input file, "transfer.energies" is distributed with the program and enables the coloring of residues according to hydrophobicity (data modified from Eisenberg & McLachlan, 1986):

```
GLY ⋆ 0.00 ! del G_obs kcal mol-1
ALA ⋆ 0.42 ! adapted from Fauchere & Pliska.
VAL ⋆ 1.66
... etc.
```

Although translation need not be avoided, infrequently, with a large protein and many symmetry equivalents, processing of a long list of residues is slow (1000 residues with 12 symmetry equivalents takes more than 15 minutes). This can be shortened by inputting an abbreviated list, containing only surface residues. To do this, run the program once, reading residue values in the normal slow way, but saving the terminal output of the program. If the same surface is to be drawn again, select the last and last but 3 tokens of each record starting "Area named" of the saved terminal output as a list suitable for input in the future. A unix shell script
fetch_surface.only {*file_name*} > {*new_list*} is supplied for this purpose.

Set_color [Number [=] {integer}] [[Parameter [=] {string}] [Property_range [=] {real} [[,] {real}]] [Parameter_range [=] {real} [[,] {real}]] [Key_title [=] [']{string}[']]
Input parameters are the same as in *Pixel_properties – Set_color*.

Paint
Does the coloring.

CONTOR Contoring is an alternative or an addition to *Pixel_properties* or *Residue_properties* for displaying a variable over the mapped area. It is especially suited to slowly changing functions such as the height. It could also be used to outline a list of input residues. It should be called after any shading or coloring.

Property [[File [=] '{string}'] [Default [=] {real}]]
This option, and the parameters above operate as described for *Pixel_properties*.

Header [Text [=] '{string}']
As contors are drawn, a key is made on the right–hand side of the plot. This option allows the user to specify a header for the key (default = 'Contor levels').

Outline [Level [=] {real}] [Label [=] '{string}'] [Width [=] {real}] [Dash [=] {real,real}]
Draw an outline separating those pixels with property values above and below level (default = 0.5). The parameters *Width* and *Dash* control the line type and are described for option *Trace_residues*. Contors are more noticeable if bold, and I use Widths in between 1.0 and 2.2. Larger values sometimes make it difficult to read lables. Dash parameters of the order of $\frac{1}{10}$'s of inches are noticeable, but their sum should not greatly exceed a pixel width. A line of appropriate type is drawn in the key and labelled with *Label* (default = value entered for *Level*).

Label Text [=] '{string}' [Position [=] {real,real}]] [Point [=] {real,real}]]
Write a label at the specified position. Text is a *required* parameter. All co–ordinates for this option are in inches, measured from the bottom left corner. The default position for a label is to add it to the bottom of the key on the right of the plot. If *Point* is given, it draws a curved line from the label to the specified point and places a "bullet" mark at this point.

Icosahedron [Height [=] {real}]] [Width [=] {real}]] [Dash [=] {real,real}]]
(For viral surfaces:) Marks the diad, 5–fold and 3–folds of an icosahedron where the axes cross a sphere of radius, height. Connects the axes to form a triangle approximating the icosahedral asymmetric unit, using a line type specified by *Width* and *Dash* (see option *Trace_residues*; suggested that Dash = 0.1, 0.05; defaults = 0.0,0.0 for solid line).

4 Files *etc.*:

Directory(ies) – path	/usr/florida/msc
Sub–directory(ies)	path/Source/Roadmap
Source	roadmap.f
Executable	path/bin/MACHINE/roadmap
	MACHINE = IBM, Dec, ESV *etc.*
Documentation	roadmap.tex, token_io.tex, expcoord.tex
...To print	lpr -Phplj roadmap.ps
Compilation *etc.*	use make -f makedec or makeibm *etc.*
Object libraries	path/Lib/MACHINE coordinates, crystal,
	matrixvector, token_io
Object sources	path/Source/Lib/*.f: `grep DOC` $\leadsto$ documentation.

The package distributed to licensed authorized users without a license for the source code should include: `roadmap`, the executable, `roadmap.ps, token_io.ps, expcoord.ps, example*.in, r14954.pdb, icosahedral.symop, transfer.energies, & atomic.solvations`.

5 Examples:

The following input files and the files that **RoadMap** would need are within the Source directory. The input files can be used with the unix input redirect (<) when running **Roadmap**.

5.1 Example 1:

```
!Example 1: Outer surface topology of one triangular icosahedral unit of
!Human rhinovirus as complexed with WIN54954.  (Color output).
Surface
```

```
Coordinate_control
Files  Input = r14954.pdb /* HRV14 (WIN54954 conf.) w/ dist to WIN54954 as Wt */
Sphere Center =  35.00000     00.00000    140.00000  Radius = 75.00000
@icosahedral.symop /* Matrices for symmetry expansion in separate file */
Plane Icosahedron = 0.0,0.0 ; End
Parallel_projection Projection_vector = 0.0,0.0,-1.0 Origin = 0.0,0.0,200.0 \
  Horizontal_axis = 0.0,1.0,0.0  Vertical_axis = 1.0,0.0,0.0 \
  Use_plane = .false. /*Add_To_Distance=150.0*/ Projection_direction = .false. \
  Distance_origin = 0.0,0.0,0.0 \
  Start_horizontal = -55.0 End_horizontal = 55.0 Space_horizontal = 2.0 \
  Start_vertical = 0.0 End_vertical = 90.0 Space_vertical = 2.0 \
  Add_Hydrogen = .true. Add_Water = .true.
End
New_page PostScript file = example1.ps \
         Header = 'Roadmap Example 1: HRV14 (as w/ WIN54954) outer surface.'
Pixel_properties
  Property Number = 1
  Set_Color Number = 1 Parameter = hue  Property_range = 140.0,160.0 \
            Key_title = 'Distance from center (A)' Parameter_range = 0.60,0.00
  Paint ; End
Contour
  Property File = Height
  Header Text = 'Distance from center (A)'
  Outline Level = 145 Width = 2.2
  Outline Level = 155 Width = 1.5 ; End
Icosahedron Dash = 0.1,0.05
Trace_residues Dash = 0.005,0.02
Label Text = Canyon Position = 1.40,4.70 Point = 4.82 3.34
Label Text = 'NIm 1A' Position = 6.49,5.59 Point = 4.92 4.57
Label Text = 'NIm 1B' Position = 2.25,6.25 Point = 4.25 5.79
Label Text = 'NIm 2' Position = 0.39,2.75 Point = 3.63,2.51
Label Text = 'NIm 3' Position = 8.46,2.44 Point = 6.71,1.41
End
```

5.2 Example 2:

```
!Example 2: The inside surface of the drug binding pocket of HRV14 is projected
!outwards onto a bent cylinder.  In Black and White shading, this example
!shows the distance of the surface to the nearest drug atom.
Surface
Coordinate_control Add_Hydrogen = .true. Add_Water = .false.
  Files  Input = r14954.pdb
  Sphere Center =  42.00000     -1.00000   123.00000  Radius = 20.00000
  @icosahedral.symop
End
Pore_projection Number_points = 6 \
  Guide_point = \
  52.31  -6.62 122.01 /* 3A extrapolation @ toe end */ \
  49.70  -5.17 122.24 \
  43.99  -2.01 122.73 \
  40.86    .85 123.66 \
  36.41   6.06 125.81 \
  33.32   9.69 127.30 /* 5A extrapolation @ pore end */ \
  Vertical_axis = 1.0,0.0,0.0 Right_hand_grip = .true. Steps = 15 \
  Spacing_radius = 2.5 Space_horizontal = 1.00
End
New_page PostScript file = example2.ps \
         Header = 'Roadmap example 2: Drug-binding pocket surface of HRV14'
Pixel_properties
  Property Number = 1 File = 'atomic_property'
  Set_Color Number = 1 Parameter = b&w  Property_range = 3.5,6.5 \
            Key_title = 'Drug-protein distance' Parameter_range = 1.0,0.30
  Paint ; End
Trace_residues
End
```

5.3 Other partial examples:

The following are parts of input files.

5.3.1 Residue sequence similarity:

This shows how to color each residue according to a list of properties for each residue in the coordinate set.

```
Residue_properties
  Property Number = 1 File = rhino_gap_abbrev.similarity
  Set_Color Number = 1 Parameter = hue  Property_range = -0.1,1.5 \
            Key_title = 'Sequence conservation' Parameter_range = 0.00,0.55
  Paint ; End
```

5.3.2 Residue hydrophobicity:

This shows how to color accordining to properties that are dependent on the type of each residue.

```
Residue_properties
  Property Number = 1 File = transfer.energies
  Set_Color Number = 1 Parameter = hue \
            Key_title = 'Hydrophobicity' \
            Property_range = -1.3 2.5 \
            Parameter_range = 0.550 0.0
  Paint ; End
```

5.3.3 :

This shows how to color accordining to properties that are dependent on the type of each atom.

```
Pixel_properties
  Property Number = 1 File = atomic.solvations
  Set_Color Number = 1 Parameter = hue \
            Key_title = 'Atomic solvation' \
            Property_range = -13 20 \
            Parameter_range = 0.55 0.0
  Paint ; End
```