MineSight® in the Foreground

Volume 24, Number 3, March 2008

Current Affairs

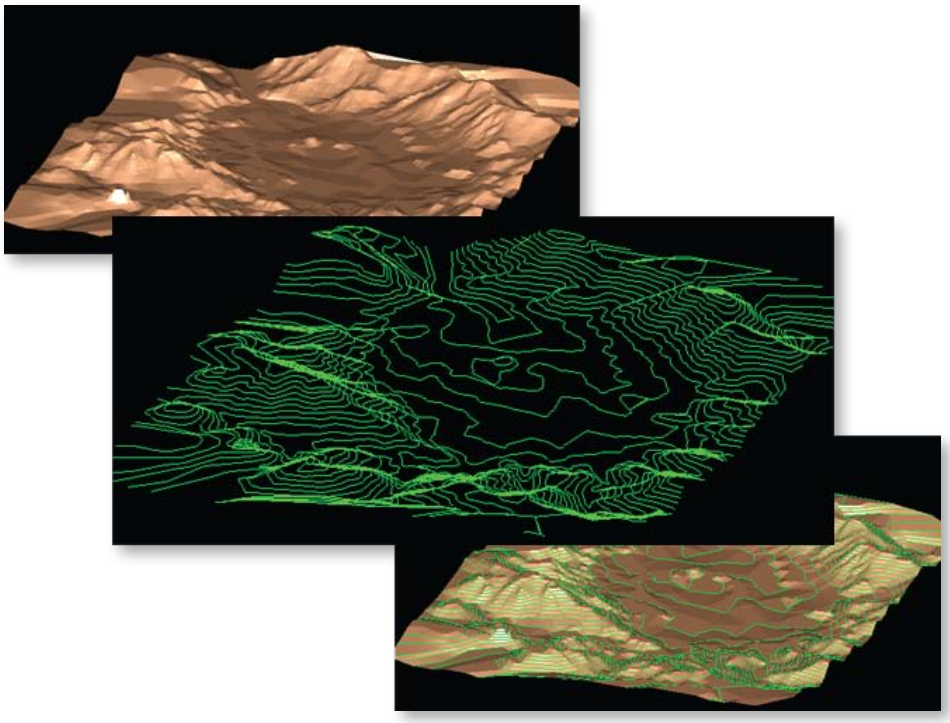
Three New Standalone Engines: contour, mscode, and SolidExpand

Three tools in MineSight® 3-D (MS3D) v.4.00 were also released on the 2007 Update CD as new standalone engine programs: **`contour.exe`**, **`mscode.exe`**, and **`SolidExpand.exe`**.

Like the other standalone engines (**`pclip.exe`**, **`clip.exe`**, **`union.exe`**, **`voxel.exe`**, etc), these engines are run outside of MS3D, and are executed from the command line using a specific syntax. For help with the syntax, use the **`-h`** option on the command line. Or you can refer to the standalone engines help file, **`StandaloneEngines.chm`**.

Since the standalone engines are executed from the command line, they can be used in MineSight® Compass™ Multi-Run packages (via procedure, **`runcmd.dat`**) or they can be run in batch mode.

Contour



The standalone engine, **`contour.exe`** is derived from the **Contour Tool** inside MS3D (found on either the **Polyline** or **Surface** menus). While the **Contour Tool** allows you to contour a surface with elevations, or contour drillhole or model items on a level or on a surface, the

(continued on page 2)

Inside This Issue:

(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 1)

standalone engine works only with shell (**.shl**) files to contour elevations.

To contour an existing surface (**.msr**) outside of MS3D from the command line, the surface must first be exported to shell (**.shl**) ASCII file format. From the **Surface** menu inside MS3D, go to **Export | Shell file**. Then select the surface to be exported, give it a name in the file chooser dialog and click on **Save**.

The shell file is an ASCII file that consists of **Point Count**, **Face Count**, **Point List**, and **Face List** information.

The required command line syntax to use for standalone engine, contour.exe is:

```
contour -ishl <input filename> -osrv
<output filename> -l [elevations to
contour]
```

➢ **Required switches:**

-ish **Input shell file** - this command must be followed by the name of the existing input **.shl** file.

-osrv **Output survey file** - this command is followed by the name output file that **contour.exe** will create and contain the resulting contour data.

-l **Levels** - this option specifies the elevation levels to be contoured. A range of levels can be specified using a starting elevation, a step size and the number of levels to contour, or you can specify the individual levels to contour with comma separation. You can also use a combination of both methods.

For example:

-l [100,10,5] means that contouring starts on level 100, step by 10, and contour the next 5 levels so that levels 100, 110, 120, 130, and 140 would be contoured.

-l 10,20,30,40,80,100 will contour only those particular levels that are specified. Note: do not use spaces in the levels format.*

-1 150,[10,2,5] this option uses a combination of the above syntax. In this case, levels 150, 10, 12, 14, 16, 18, and 20 would be contoured.

**If spaces must be used, use double quotes, i.e.,* "1,2, 3 [4, 1, 10]".

➢ **Optional switches:**

-idx **Index option** – is used to output the contour index level in the output survey file as the point code (output in columns 1-4 in the output file). If this option is not used, then a point code of "none" is used.

For example, if the contours are on the 40th level from the specified base elevation, then the point code will be 0040 (as shown in the examples below).

Output not using the **-idx** option

```
* * * Top of File * * *
none          4000.000    5946.898    2600.000
none          3987.000    5941.298    2600.000
none          3973.901    5935.799    2600.000
none          3973.901    5935.798    2600.000
none          3955.501    5927.899    2600.000
none          3955.501    5927.898    2600.000
none          3937.101    5920.099    2600.000
none          3937.101    5920.098    2600.000
none          3937.099    5920.098    2600.000
none          3918.700    5912.298    2600.000
none          3900.301    5904.399    2600.000
none          3900.301    5904.398    2600.000
```

Output using the **-idx** option

```
* * * Top of File * * *
0040          4000.000    5946.898    2600.000
0040          3987.000    5941.298    2600.000
0040          3973.901    5935.799    2600.000
0040          3973.901    5935.798    2600.000
0040          3955.501    5927.899    2600.000
0040          3955.501    5927.898    2600.000
0040          3937.101    5920.099    2600.000
0040          3937.101    5920.098    2600.000
0040          3937.099    5920.098    2600.000
0040          3918.700    5912.298    2600.000
0040          3900.301    5904.399    2600.000
0040          3900.301    5904.398    2600.000
```

Output ASCII files generated by **contour.exe** *with and without using the* **-idx** *option.*

-e **Exact option** – is used to force the program to use exact contours and not "bump up" the values. This option is the same as the Contour coincident faces and edges option in the **Contour Tool** inside MS3D. It is used when contour edges fall exactly on one of the specified contour increments.

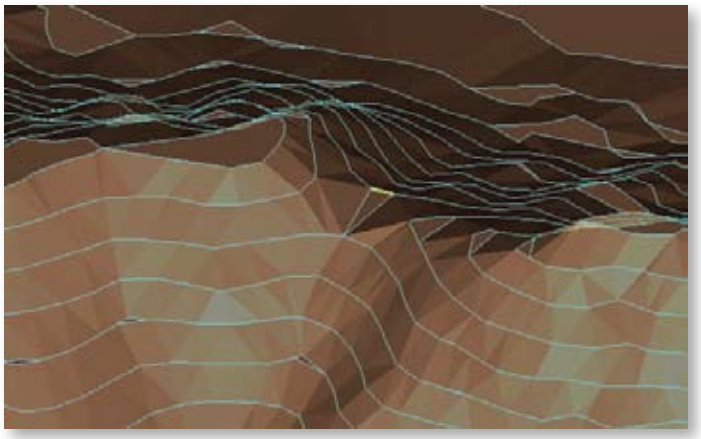
The following illustrations show the results from contouring the same surface (shell file) with and without using the **-e** option. (The results were imported into MS3D to illustrate the differences in the two methods for purposes of this article).

Notice in this case, when using the **-e** option, numerous polyline segments are generated and all are along the exact contours. Without using the **-e**

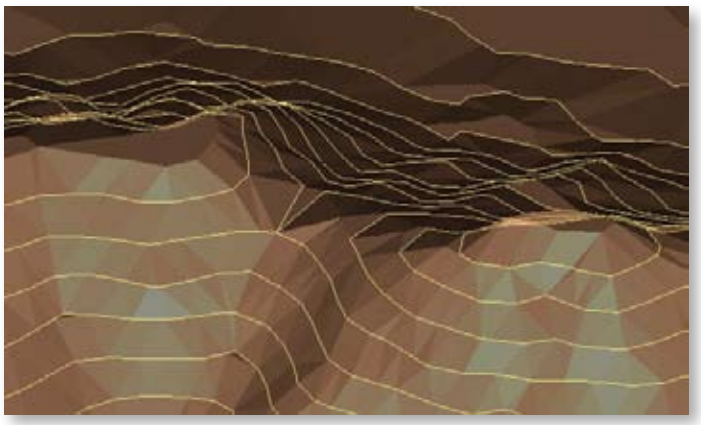
(continued on page 3)

(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 2)

option, if points are detected exactly on a level, then contours are bumped up by a very small amount (an amount equal to float epsilon, a very small number).



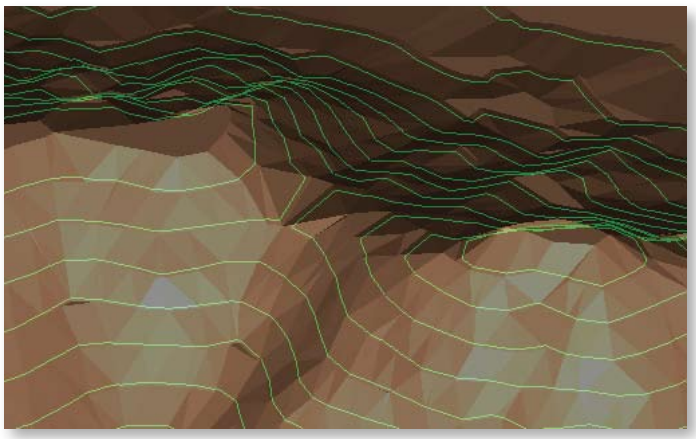
Contours generated by the standalone engine, **`contour.exe`** *using the* **-e** *option. Notice the polylines in the center of the picture (in the saddle), in particular the highlighted segment (in yellow).*



Contour results from **`contour.exe`** *without using any of the optional switches (without the* **-e** *option).*

-t **Tolerance option** - is used in conjunction with a given tolerance value to detect coincident points. It should be used when the surface contains many areas with numerous minute changes within the tolerance range.

-b Used with a specified value, this option is used for bumping up the elevation to remove coincident faces and edges. It can only be used when the -e option is not specified.

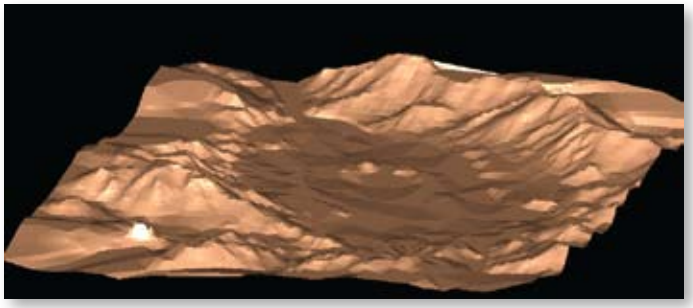


Contour results from **`contour.exe`** *using the* **-b** *option with a specified value of 2.*

-q **Quiet option** – except for any error messages, this option is used to suppress all screen output.

Example

Using an existing surface (**`.msr`**) from MS3D, export it to **`.shl`** file format (**Surface menu | Export**) to create the input file necessary for **`contour.exe`**.



The surface msr object in MS3D used in this example.

Then use the following syntax to contour from a base elevation of 2,000 ft and use a step size of 15 ft with a step count of 64. In this example, the surface will be contoured using this particular project's minimum elevation for the base elevation, the project's cell size (dz) as the step size, and a step count equal to the number of benches in the project.

```
contour –ishl topo.shl –osrv
topocontours.srv –l [2000,15,64]
```

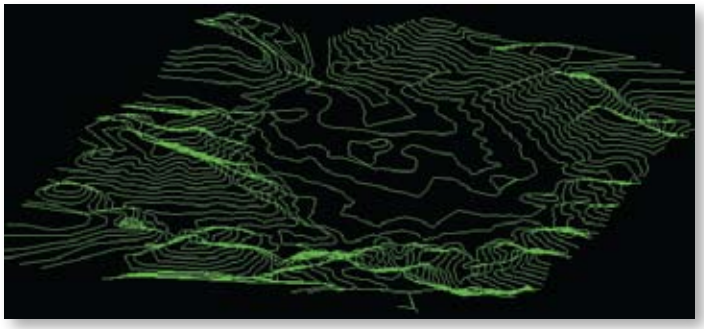
This creates an output file called, **`topocontours.srv`** in a column delimited **`.srv`** format ASCII file (an example of which is shown above). The columns in the output file (left → right) consist of the point code, Easting, Northing, and elevation. If the **-idx** option is used, the point code is equal to the

(continued on page 4)

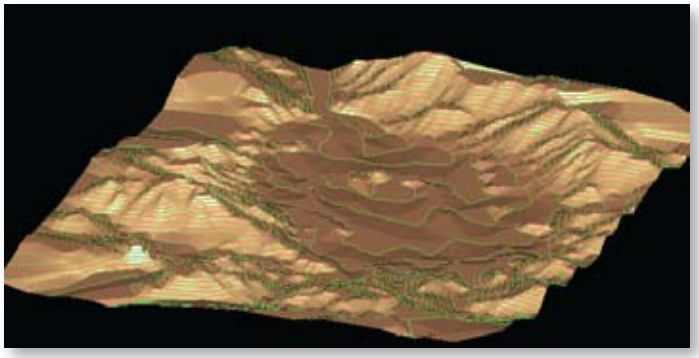
(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 3)

contour index, or level number in the step count. Otherwise, "none" is used at the default index value.

Once the resulting output file has been created, the data can be imported into MS3D for you to view it in 3-D.



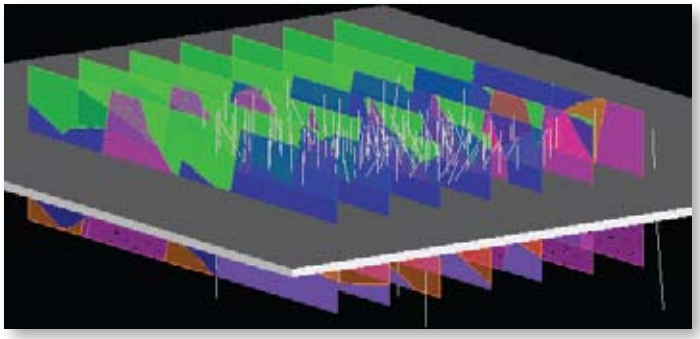
Result from contouring using **`contour.exe`**, *viewed inside MS3D.*



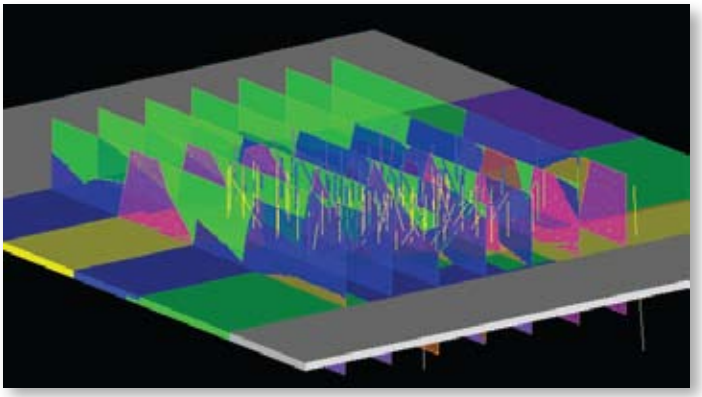
Original topographic surface + contours.

mscode

The **`mscode.exe`** standalone engine is used to code drillhole intervals or model blocks from polygons or solids. It can also be used to generate a partials file and not code the model.

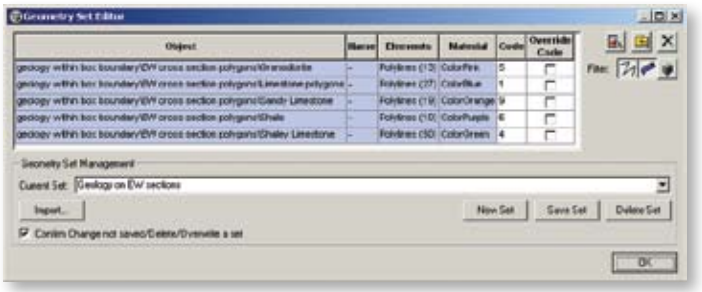


Drillholes and a model view slab shown with a series of cross sections that contain geology polygons before coding.

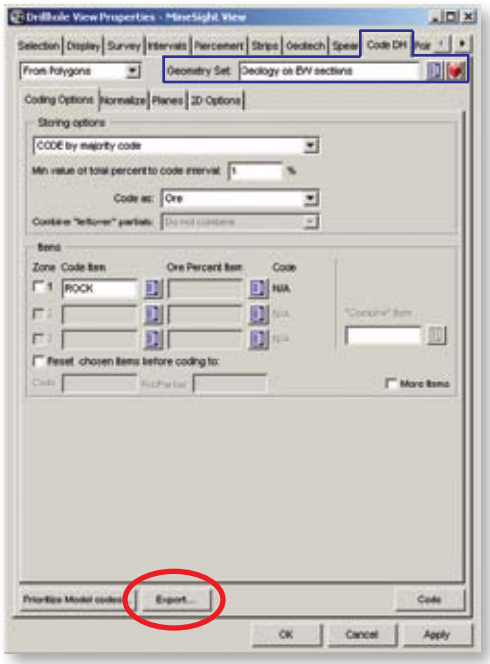


Drillhole intervals and model view slab after coding from polygons using **`mscode.exe`**.

The geometry to be used for coding must exist in an MS3D project and is accessed by **`mscode.exe`** via a template file. Template files are created in MS3D using the **Geometry Set Editor** followed by the **Export** option in **Model View Coding** or **Drillhole View Coding**. Template files are saved with the extension, **`.mscode`**. Different template files should be created for model view coding and for drillhole coding.

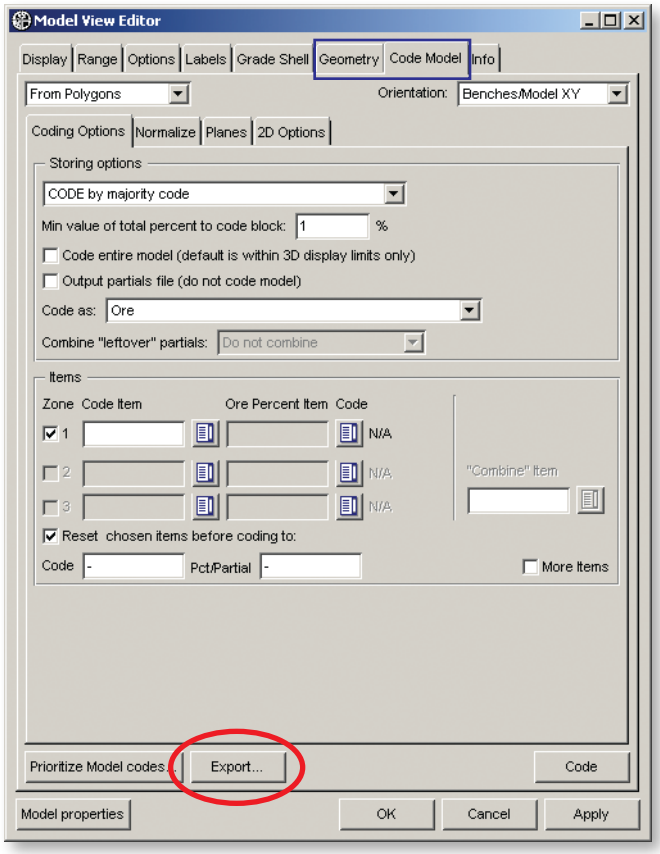


Geometry sets are created in the **Geometry Set Editor**.



(continued on page 5)

(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 4)



Drillhole View Properites | Code DH *tab dialog (on the previous page) and the* **Model View Editor | Code Model** *tab dialog (above). The* **Export** *button on either dialog is shown circled in red. As indicated by the blue outline on either dialog, in* **Drillhole Views**, **Geometry Sets** *are created and accessed via the* **Code DH** *tab dialog and in* **Model Views**, **Geometry Sets** *are created and accessed via the* **Geometry** *tab dialog.*

The template file is an XML file. Although modifying this file is not recommended, it can be modified with a text editor. It contains information about the **Geometry Set** (location, **`.msr`** file names, etc.) as well as the **Drillhole** or **Model View** information. Comments inside the file document identify the various values it contains.

The Template file created in MS3D contains information about the selected **Geometry Set** *and about the MineSight® drillhole and/or model file.*

The required command line syntax to use for standalone engine, `mscode.exe` is:

```
mscode -t <template filename> -o <output partials filename>
```

➢ **Required switches:**

-t **Template file** - this command must be followed by the name of the existing template file (**`*.mscode`**).

➢ **Optional switches:**

-o **Output partials file** - this command must be followed by the name of the output partials file to be created. When this option is used, a partials file will be created and the model will not be coded.

This option is only available when using template files that code models. Multiple sessions will overwrite, rather than append, values to the specified file. The output file

(continued on page 6)

(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 5)

is written in MineSight® program, M659V1 format:

```
Bench, Row, Column1, Partial1,
Column2, Partial2, Bench,
Elevation
```

All blocks with column numbers between Column1 and Column2 have partial 100%.

[Note: Documentation for program M659V1 is found in %medexe%\doc\M600.PDF].

-r **Real (floating point)** - use this option to output percent values in the partials file as real (floating point) values. This option invokes the **Free format (-f)** switch.

-f **Free format** – this option is used to output the partials file in free format. Free format option guarantees spaces between entries.

-np **No progress** - this switch is used to suppress the progress display. By default, while **`mscode.exe`** is running, progress meter percentiles are output to the screen. The **-np** option will avoid printing that list.

-q **Quiet** – except for error messages, this option suppresses all output.

Note: When coding from polygons, where there are unresolved model blocks or drillhole intervals, the plane name or plane location as well as the number of unresolved blocks or intervals will be reported to the message window. Unresolved blocks or intervals occur when there is an overlap of polygons with different codes.

Example

To output the following partials file (**`output.prt`**) in free format, the following syntax was used:

```
mscode -t newsletter.mscode -o output.
prt -f
```

```
* * * Top of File * * *
 15   0   0     0     0     0   15  2735.0
 15   1  43    50    66    72   15  2735.0
 15   1  67   100    74    50   15  2735.0
 15   1  75   100   101    92   15  2735.0
 15   1 102   100   128    50   15  2735.0
 15   2  43    50    66    72   15  2735.0
 15   2  67   100    74    50   15  2735.0
 15   2  75   100   101    92   15  2735.0
 15   2 102   100   128    50   15  2735.0
 15   3  43    50    66    72   15  2735.0
 15   3  67   100    74    50   15  2735.0
 15   3  75   100   101    92   15  2735.0
```

*Shown is the top of the partials files (**`output.prt`**) created by **`mscode.exe`**. This partials file is in program M659V1 column-delimited format (L→R): Bench, Row, Column, Partial Percent, Column, Partial Percent, Bench, Elevation.*

SolidExpand

*Solids displayed in 3-D are shown both before and after expansion (the before solid is shown enclosed within the after). This solid expansion was accomplished using the **`SolidExpand.exe`** standalone engine and the result was then imported into MS3D for 3-D visualization.*

The standalone engine, **`SolidExpand.exe`** is used to expand MineSight® shells isotropically by a user-specified amount. This program is the standalone version of the **Solid Expansion Tool** found on the **Surface** menu inside MS3D.

The **SolidExpand** standalone engine is executed from the command line and works only with shell (**`.shl`**) files.

To use an MS3D solid, first export the geometry **`.msr`** file to **`.shl`** output format (**Surface | Export**).

The required command line syntax to use for standalone engine, `SolidExpand.exe` is:

```
SolidExpand –I <input .shl file> -O
<output filename> -offset <amount>
```

➢ **Required switches:**

-I **Input** - this option must be followed by the name of the existing input shell filename.

-O **Output** - this option must be followed by the output shell filename to be created by this standalone engine.

(continued on page 7)

(Three New Standalone Engines: contour, mscode, and SolidExpand continued from page 6)

-offset Offset value – this option must be followed by the value to expand the solid(s). Use a positive value to expand and a negative value to shrink a solid(s).

➢ **Optional Switches:**

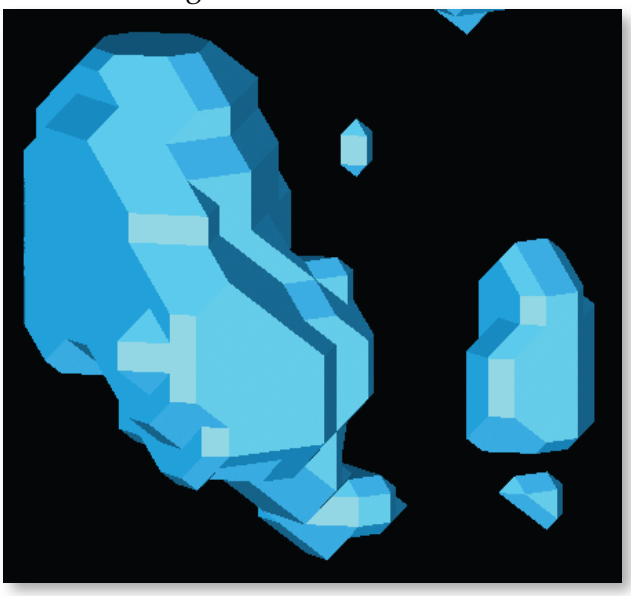
-a **Accuracy level** - enter 0 or 1; 0 for lower accuracy, 1 for higher accuracy. Accuracy is proportional to the absolute value of the expansion width. Expansion done using the **high accuracy** will perform the expansion in a sequence of smaller expansions, thereby increasing the accuracy. However, this method will use more of your computer's resources. If your input shell data contains many faces and points, you may want a higher level of accuracy in the result.

-dat **Data directory** - use this option together with the complete path to the directory where the input shell file is located.

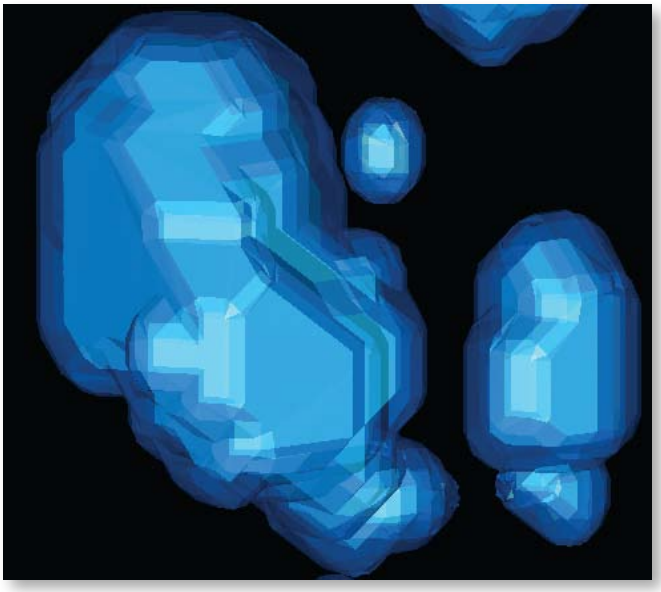
Future versions of the **`SolidExpand.exe`** standalone engine will include the ability to expand in different directions, find the center of mass, compute the solid's area, etc.

Example

In the pictures shown below, grade shell solids are expanded by 15 ft using the **`SolidExpand.exe`** standalone engine.



Grade shell solids before expansion.



Grade shell solids after expansion (shown with transparency on) surrounding the original grade shells.

The grade shells were first exported from MS3D to a shell file (via **Surface | Export**) and put in the directory, **`c:\inputshells`**. This shell file is the input for **`SolidExpand.exe`**. The shells were then expanded using the following syntax and the result was saved to another shell file called, **`gradeshells_expanded.shl`**. The pictures show the resulting solids shell file after it was imported into MS3D to show the results in the 3-D viewer.

```
SolidExpand -dat c:\inputshells -I gradeshells.shl -O gradeshells_expanded.shl
-offset 15
```

These new standalone engines are installed as part of the 2007 Update CD. If you have any questions about the standalone engines, please contact Mintec Technical Support.