Contents

How to use Mountain Terrain

On startup, the program creates a Mountain Terrain data set, and then draws it in the main window, based on the size of that window. You can then play around with the set, to tweak it to your liking, or you can create a whole new data set to work with. You have several parameters you may change, as well as a near infinite variety of data sets.

Commands to change current data set:

Color
Applies a filter to the colors used to draw the mountains. This provides the most visceral change to the appearance of the mountain set. This can be set from the toolbar, options menu or settings dialog.

Relative Water Depth
Though the program does not draw mountains underwater, it does keep track of how deep they run. With Relative Water Depth on, the program colors water darker the deeper the mountains are beneath it; otherwise, all water has the same uniform blue. This can be set from the toolbar, options menu or settings dialog.

Size
If you resize the window, the program will redraw the mountains in the optimal way to fill the new area. This can produce some interesting effects. You can resize the window by using the directional arrows that appear on the edges of the window.

Commands to change the next data set:

All of the above parameters apply to new data sets, as well as:

Cragginess
Determines how sharply the mountain heights crease near each other. High Cragginess settings usually produce tall mountains and deep depressions, while low settings yield flat terrains. This can be set from the toolbar or settings dialog.

Low Detail
By default, the program creates data sets in High detail mode. By toggling Low detail, it creates a data set one quarter the size of a full set. Low detail mountains draw faster, and have larger faces, which is especially appealing when using the Sonar color setting. This can be set from the toolbar, options menu or settings dialog.

Commands to create and save data sets:

New
Creates and draws a new random set of mountains. This can be found on the toolbar or file menu.

Open
Loads a saved data set from disk. This can be found on the toolbar or file menu.

Save
Saves a data set to disk. This can be found on the toolbar or file menu.

Automatic Redraw
Creates a new data set shortly after the previous set finishes. This is used to make a constantly updating screen, for a hands-off approach to mountain generation. This can be found on the toolbar, options menu or settings dialog.

File Menu

New
Creates and draws a new mountain data set.

Open
Opens a previously saved data set.

Save
Saves the current data set.

Exit
Quits Mountain Terrain. But why would you want to do that?

Options Menu

Color
Chooses a new color for the current data set. The color will not update immediately, unless the program has not yet finished drawing the mountain set. To force a redraw, try resizing the screen.

Low Detail
Toggles on or off Low detail mode. In this mode, the data set will be one fourth the normal size, making individual elements appear larger, and the whole system to draw faster. This is the optimal setting for use with the Sonar color mode.

Relative Depth
Toggles on or off the Relative Water Depth mode. In this mode, the deeper the mountains are beneath the water, the darker the water gets. The colors fade pretty smoothly on most systems, and can lead to some nice effects.

Auto Redraw
Toggles the Automatic Redraw setting. When this is on, the program creates a new data set after the previous one has completed drawing. The program will continuously draw new mountain sets, with a short delay between each, like a slide show.

Settings...
Brings up the settings dialog. All options from the menu may also be changed in the dialog.

Settings Dialog

Color
Chooses a new color for the current data set. The color will not update immediately, unless the program has not yet finished drawing the mountain set. To force a redraw, try resizing the screen.

Cragginess
Determines the sharpness of the mountain peaks. This can be any integer value between 1 and 128. You may either use the scroll arrows to the right of the control to change it or type in a new value. Entering an illegal value will not affect the program.

Low Detail
Toggles on or off Low detail mode. In this mode, the data set will be one fourth the normal size, making individual elements appear larger, and the whole system to draw faster. This is the optimal setting for use with the Sonar color mode.

Relative Depth
Toggles on or off the Relative Water Depth mode. In this mode, the deeper the mountains are beneath the water, the darker the water gets. The colors fade pretty smoothly on most systems, and can lead to some nice effects.

Auto Redraw
Toggles the Automatic Redraw setting. When this is on, the program creates a new data set after the previous one has completed drawing. The program will continuously draw new mountain sets, with a short delay between each, like a slide show.

Use OK or Cancel to leave the dialog and return to the program. Choosing OK causes the program to redraw the mountains with the options that apply to the current set (Color, Relative Depth, Low Detail). Other changes will be used for the next data set.

Toolbar

The Toolbar sits above the main drawing area. There are buttons on it, to control the majority of the settings to control the way the mountains are created and drawn.

New
Creates and draws a new mountain data set.

Open
Opens a previously saved data set.

Save
Saves the current data set.

Low Detail
Toggles on or off Low detail mode. In this mode, the data set will be one fourth the normal size, making individual elements appear larger, and the whole system to draw faster. This is the optimal setting for use with the Sonar color mode.

Relative Depth
Toggles on or off the Relative Water Depth mode. In this mode, the deeper the mountains are beneath the water, the darker the water gets. The colors fade pretty smoothly on most systems, and can lead to some nice effects.

Auto Redraw
Toggles the Automatic Redraw setting. When this is on, the program creates a new data set after the previous one has completed drawing. The program will continuously draw new mountain sets, with a short delay between each, like a slide show.

Color
Chooses a new color for the current data set. The color will not update immediately, unless the program has not yet finished drawing the mountain set. To force a redraw, try resizing the screen.

Cragginess
Determines the sharpness of the mountain peaks. This can be any integer value between 1 and 128. You may select any value from the list. High values create sharper mountains.

Technical Information

This is the third version of the Mountain Terrain Program. Version 1.0 was very basic and never saw release. Version 2.0 is a Windows 95 screen saver, and is still available at my web site:

http://www.netcom.com/~baren/

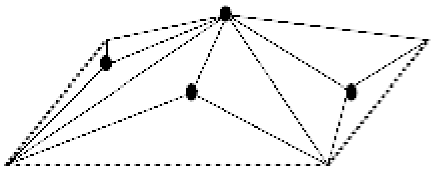
Version 3.0 has primarily superficial changes to the mountain drawing routines. The new elements of it are almost enitrely in the user interface. As this program is an experiment in Windows programming, I packed it with kitsch, like having a Toolbar, Settings dialog and Options menu for changing most of the settings. All of the user interface stuff I learned from reading Charles Petzold's book Programming Windows 95, and I highly recommend this book to anyone interested in Windows Programming.

One significant difference from version 2.0 is the lack of an Elevation setting. That setting never really did all that much to the appearance of the mountains, so I removed the user interface for it. The setting is still present in the program, but is now hard coded with a value of 100.

All of the really interesting stuff going on in the program takes place in the routines which create and draw the mountains, so if you're interested in how those work, read on.

To create a new data set, the program initializes a two dimensional array, currently hardcoded with a range of 64 by 64. It then examines the size of the client area of the window, and determines the boundaries of the parallelogram that will contain the zero elevation level of the mountains, as well as finding the aspect ratios along the theoretical X and Y axes of the parallelogram.

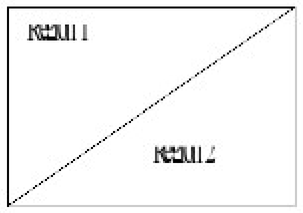
Then, the real work begins. It sets the corners of the field to random values from -(Elevation) / 2 to (Elevation) / 2. It then enters a recursive loop, passing the corner heights to a refinement function, which perturbs the four midpoints of the edges and the center of the rectangle by an amount equal to the average of the four elevations plus or minus half of Cragginess. The four rectangles thus created are then passed to the same function, but with Cragginess divided by two.



In the later iterations of the recursive loop, Cragginess approaches 1 as the rectangles it is perturbing get smaller. In this way, the gradient between discrete points gets smaller as the points get nearer. When the recursive function has finished, the data is set. The Field contains Z values for each point in the X and Y bounded rectangle. This is commonly known as a Z-buffer or height field.

Drawing the mountains should probably be accomplished cleaner than I'm doing it, but...

The main drawing loop iterates Y times, drawing a stripe each time on the X axis. For each X value, the program creates two triangles, as shown:



The corners of the triangles are set to Zero if below water level, and if the whole triangle is at height zero, it is marked as being Underwater. Otherwise, the program then examines the vector normal to the surface of the triangle, and colors it based on the direction of that vector. (More on color below). It draws the two triangles, and then reiterates the loop with an incremented X value.

If X is at the right edge of the Field array, it constructs a solid color rectangle from the far right edge to sea level, to give the appearance of having been sliced out of a map. The same occurs when the Y value is at the bottom edge, making the last pass the slowest one to draw.

When the sonar color is turned on, the program ignores both of these, as well as the color chosen by the vector normal. It instead draws each triangle twice, once in solid black, and once with a white border. This is my greatest heartache with this program. Windows does not have a way to draw a solid region with a border. That's why Sonar is so slow, and the borders for all interior regions are so thick (two borders back to back).

Choosing colors takes up the greatest volume in the program code. First it determines the system's output capability, be it 256 colors or more. This is the result of 256 colors, using my masking scheme, just look bad, so I had to come up with specific masking values for use with these systems. Also, vector colors for left side triangles do not work for right side triangles, so these have to handled separately. In total, the color matching algorithm is repeated FOUR times in the code. Sigh. What an eyesore.

Anyway, the way it works is this: The user chooses a color preferece. This preference is translated in a hexidecimal code, representing a solid color of that choice. Gray is 0x00FFFFFF, Purple is 0x00FF00FF, and so forth. Then, after the normal of a triangle is determined, it checks in a series of if-then-else statements to the theoretical possible directions it could face. Fortunately, the setup of the Field eliminates many of these choices -- for instance, the vector will never point downward, knocking out half of the options right there. In total, there are 9 possible facings I check. Some of them never appear, but have been left in anyway.

When it finds the proper facing, it creates a solid windows brush with a darkness value from the facing combined with the color preference using a logical AND. This is the brush used to fill the triangular regions created. Darkening water is simply a subtraction of the average height of the Field values at that location from Pure Blue, up to a reasonable maximum to prevent it from looping over into some other color (dark red -- wouldn't that be ominous?)

Still with me? Still care?

Most of this stuff is pretty basic, actually, and was gleaned from hints dropped on the net. At the time of this writing, landscape generation is a pretty hot topic on the programming groups, after some helicopter game made waves using Voxel technology for landscaping its terrains. As near as I can tell, though, the voxel stuff is not much different from what I'm doing here. The main trouble with this program is its speed, which is entirely hampered by Windows' GDI bottleneck.

Where to go in the future? Will there be a Version 4.0? Well, probably. My hope is to figure out the mechanics of Brownian motion, a physics concept which can be used to describe very accurately natural phenomena. Mountains generated this way are breathtaking. The topic has thus far proven to be extremely elusive, however, so don't hold your breath.

Credits

Written by A.C. Barentyne
Copyright 1997, Paradigms Lost

Original idea for this program came from XMountain, a Brownian Motion generator available as freeware for XWindows on Unix systems.

Many thanks are due to:

Crispen Scott
My boss, who loved the idea of a Mountain terrain program

Lucian Wischik
Whose Scrplus lib allowed me to write the Screen Saver version, my first Windows Program

Charles Petzold
Whose book, Programming Windows95, allowed me to give this program a modern-style interface