Creating Terrain for Simulations

MultiGen® Creator
Version 2.6 for Windows and IRIX
April 2003

MultiGen-Paradigm
VISUALIZE REALITY

Creating Terrain for Simulations, Version 2.6 for Windows and IRIX
April 2003

Printed in the U.S.A.
Part Number: DBMC04 08/01
DRAFT - April 2003

Contents

Chapter 1 Overview 1-1

Chapter 2 Planning the Database 2-1

Scenario Requirements 2-1
Hardware and Software Requirements 2-1
Estimating Your Polygon Budget 2-2
Organizing Your Data 2-3

Chapter 3 Creating a Test Terrain 3-1

Converting Terrain Source Data 3-1
Importing and Viewing the Terrain Data 3-3
Choosing Single File or Batch Process 3-5
Selecting a Terrain Area 3-5
Creating Levels of Detail 3-7
- Number of LODs 3-7
- Switching Distances 3-8

Setting Contour Properties 3-9
Choosing a Map Projection 3-10
Choosing a Processing Algorithm 3-14
Testing the Terrain 3-18
- Partial Groups 3-18
- Edge Slivers 3-19
- Walls and Gaps 3-21

Chapter 4 Using Textures with Terrain 4-1

Geospecific Texture 4-2
Clip Textures and Mipmaps 4-3
High Resolution Insets 4-5
Detail Textures 4-6
Indirect Texture 4-7

Chapter 5 Projecting Features 5-1
Deciding What to Project 5-2
Importing and Correlating Feature Data 5-2
Using the GeoFeature Menu 5-3
Feature Only Projection 5-6
Modifying Feature Data 5-8
Creating New Features 5-10
Feature Projection Methods 5-11
Manual Post-Projection 5-11
Automatic Post-Projection 5-12
Automatic Pre-Projection 5-12
Setting Up Feature and Projection Files 5-15
Feature Preferences 5-15
Projection Preferences 5-16
Library Substitution/External References 5-16
Rules and Actions 5-17
Palette Files 5-18
Adding Light Points and Strings 5-18
Projecting the Features 5-19

Chapter 6 Adding Special Features 6-1
Building Roads 6-1
Road Construction 6-2
Road Tessellation 6-5
Scenario Data 6-8
Roadway Centerline Files 6-8
Scenario Lane Files 6-10
Adding Special Models 6-11
Importing Models From Other Applications 6-12

Chapter 7 Refining Your Database 7-1
Balancing Culling and Drawing 7-1
Reducing Polygon Count 7-1

Chapter 8 Finishing Your Simulation 8-1

Index Index-1

1 Overview

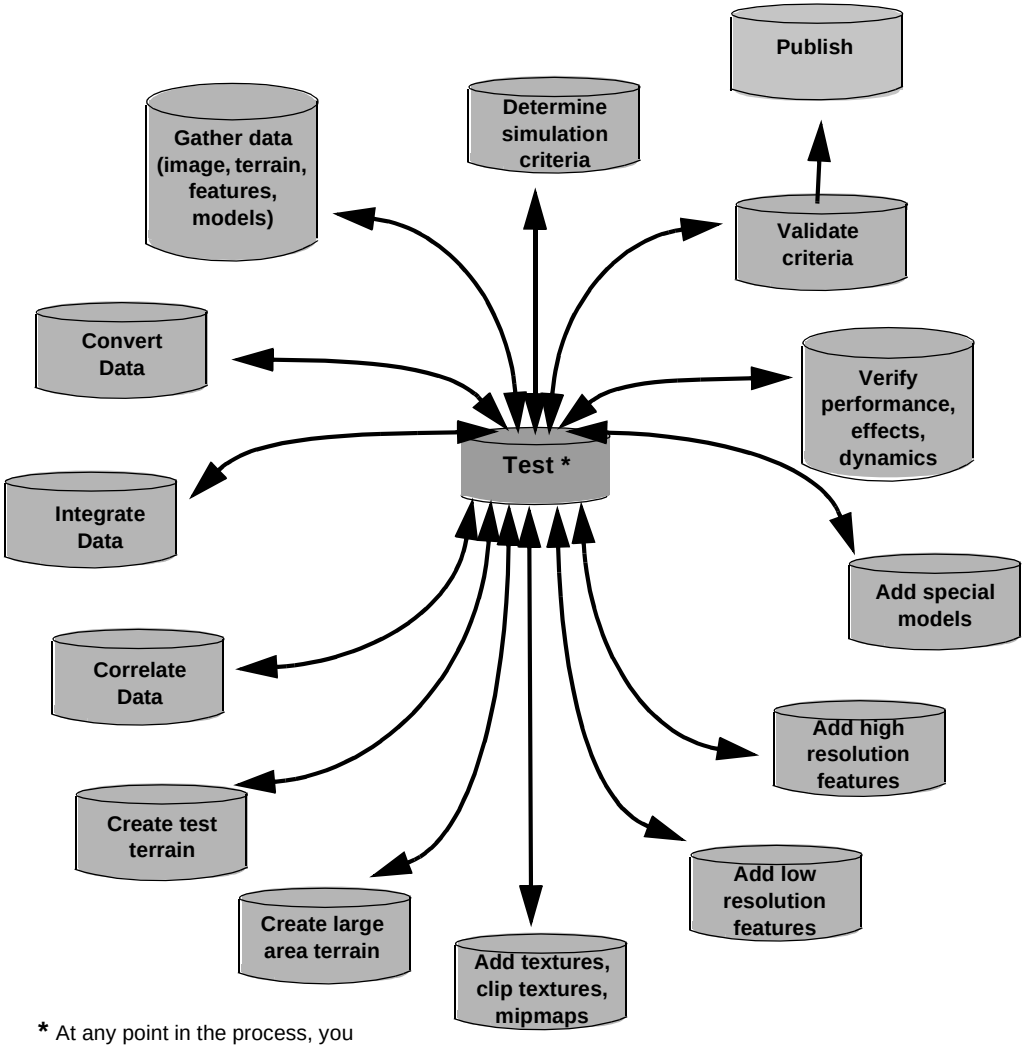
You have a requirement to build a terrain database for a realtime simulation using Creator. What tasks are required to accomplish this? To a certain extent, creating a well organized, efficient, and useful database for simulation is an iterative process. However, you can considerably reduce the number of iterations required by doing some preliminary planning and preparation, and a lot of testing.

The chapters in this book describe a general sequence of tasks that are required to perform to create terrain databases for a realtime simulation, and some of the things you need to consider when doing each step. The tasks include the following:

- **Planning** - Set some goals for what your simulation will do, and determine whether your hardware and realtime software can meet those goals.
- **Organization and preparation** - Gather together the data you will be using to create your simulation. Some format conversion and correlation may be necessary.
- **Building and testing** - Build a test terrain database using a small sample of your elevation data. The generated test terrain must be checked for errors at each level of detail (LOD). It is also a good idea to run your sample database in your realtime system to see if it runs smoothly and efficiently, with no anomalies or holes.
- **Adding textures and general features** - Once your test terrain looks good, you can select the area your database will cover and then consider adding textures and feature data. You must decide what features your simulation requires. You need to verify your feature data for any errors or overlaps, and reduce the number of features or vertices if they are too dense.
- **Checking and refining** - When your terrain and feature data is generated, projected, and checked in Creator, you will need to run the database in your realtime system. This is where you will check its performance. You may need to reduce polygons or rearrange your database hierarchy to obtain better performance. All levels of detail must be checked to make sure they switch smoothly, and that features don't suddenly pop in and out of view. You may also decide to change some of the properties (such as texture or material) assigned to your polygons to make your database look more realistic.

- **Adding additional specific models** - Now you can start adding geospecific or target models to your database if you wish. The models may have been built using Creator, or they may have been modeled in another application. If they were modeled in another application they will probably need to be optimized and restructured for realtime performance.

Generating a terrain simulation is an iterative process



* At any point in the process, you may have to repeat earlier steps

2 Planning the Database

Before you create your terrain database you need to take time to plan it out properly. Several factors need to be considered:

- Scenario / project requirements
- Hardware capabilities
- Realtime / rendering software
- Polygon budget

Scenario Requirements

Your first goal is to determine what type of simulation your database will be supporting. Will it be high-flying (flying at high altitude over terrain)? Low-flying (a low altitude flight)? A ground-based driving simulation? A combination of different types? An airplane flying at 40,000 feet has a large viewing area, but the terrain can be less accurate than if it were viewed closer to the ground. A vehicle on the ground has a more limited viewing area, but the terrain and culture that is visible from a vehicle must be very detailed, with a high degree of realism.

Scenario requirements also impact the number of levels of detail you will need. If your scenario is a flight simulation, you will probably need at least two levels of detail, possibly more. Conversely, if your simulation is an architectural walk-through, you may only need one level of detail.

Hardware and Software Requirements

You must determine the number of polygons, textures, light sources, and so forth, that can be supported by the target hardware platform running the realtime simulation.

How much time, in milliseconds, is required for applications processing, collision detection, culling, and rendering? *Applications processing* includes calculating the x, y, z position, and yaw, pitch, roll of the eyepoint, and the flight / driving model acceleration. Many realtime systems approximate the surface of a model by wrapping

it in a simplified shape, called a *bounding volume*. *Collision detection* calculates when the bounding volumes of models intersect, indicating that a moving model has collided with a stationary object or terrain polygon, or with another moving model. *Culling* is the process of determining what needs to be drawn, based on the eyepoint and viewing frustum. *Rendering* involves displaying the results to the screen. You must determine whether your realtime system can render the visual database at the required frame rate. If not, you may need to simplify your terrain.

Estimating Your Polygon Budget

One of the most important steps in planning your terrain database (or any database) is estimating your polygon budget. This directly affects your simulation's performance as well as the amount of realism you can achieve. A polygon budget represents the number of polygons that can be drawn in one frame at one time. It is related to the frame rate. Typically, only a portion of the entire database is drawn at any given time.

The main factors affecting polygon budget are hardware limitations, and the complexity (density) of the database you want to create. Remember, polygon budget applies to the entire database, including the terrain, any features you want to project onto it, and any other models that will be a part of your virtual environment, such as moving models.

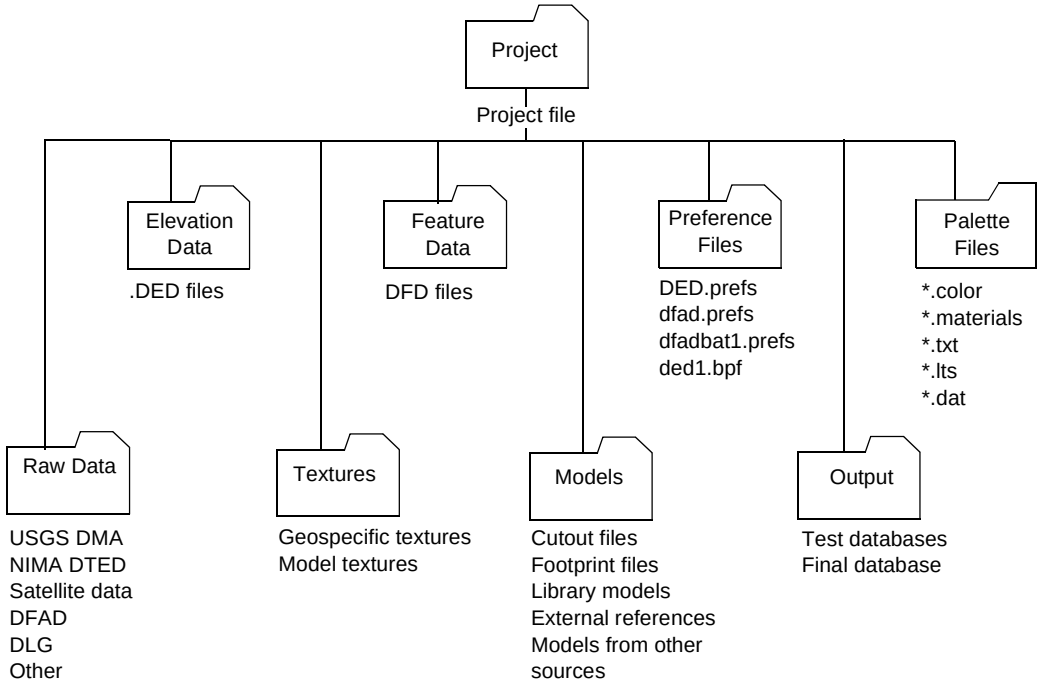
The simplest way to estimate your polygon budget is to use your hardware vendor's published polygon performance rate (polygons per second) and divide it by the number of frames-per-second at which you want to run the simulation (usually between 15 and 60). Because published hardware performance rates are typically measured under optimum conditions, you should divide these rates by a factor of 3 or 4 to obtain a more realistic polygon budget for a reasonably complex database.

There are more elaborate and accurate ways to determine polygon budget that take into account pixel fill rate, depth complexity, and the types of features and moving models that will be part of the database. MultiGen-Paradigm offers training classes that demonstrate more accurate methods, but the previous example will give you a quick approximation.

You also need to decide how you will allocate your polygon budget. A general rule is to allocate one third of the budget for terrain; one third for static features such as light points, buildings, and trees; and one third for moving models such as cars and planes.

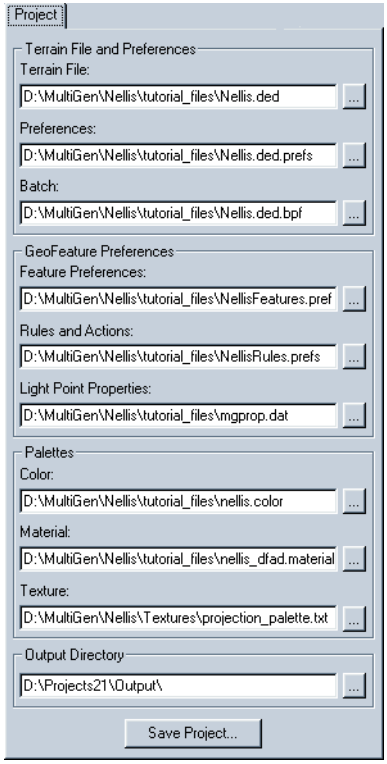
Organizing Your Data

It's a good idea to keep all the files associated with a project organized into subdirectories within a main project directory. This prevents data from getting mixed up between different projects and inadvertently changed or deleted. The following figure shows an example of how a project directory could be set up.



Creator helps you to organize the various files necessary for a project in the *Terrain* window's Project panel. In the Project panel you tell Creator which elevation data, preferences, settings, custom palettes and output directory to use. You can save your panel settings in a Project file (*.prj). The Project file will save this information and load

the necessary files when the project is loaded. Project files store settings for both batch and individual terrain generation files.



Several files external to Creator are associated with terrain database generation. They contain settings made in the Preferences panels, the *Import Terrain* window, and feature projection preferences. References to these files are saved as part of the project file. You can also create custom Color, Material, and Texture palettes that contain the properties you need for your database, and save them to palette files. Each of the following files has a default file name and directory path. However, you can rename the files and move them to another location, such as a project directory. The new file names and paths can then be saved in a project file.

- **<ded>.prefs**

  Stores the *Terrain* window settings for a DED file. The default file name is the name of the digital elevation data file (DED) with a `.prefs` extension. The information contained in this file includes the **Gaming Area** selection, contour

band sizes and colors, the **Face Color From** setting, LOD settings, texture, map and terrain preference settings. The file is stored in the same directory as the DED file.

- **<ded>1.bpf**

  Stores the *Terrain* window Batch panel settings. These settings include area block size, file naming scheme, and information for each tile in the area, such as whether it was previously processed or selected. The default file name is `<DED file name>1.bpf`. By default, it is saved to the same directory as the DED file.

- **dfad.prefs**

  Contains settings defined in the *Feature Preferences* window that control the geometry and texturing of features as they are projected. The default location for this file is the directory where Creator is installed.

- **dfadbat1.prefs**

  Stores rules and actions settings for projecting features, and also contains the list of `.dfd` files associated with the project. These preferences are set up in the *Action List* and *LOD Rule List* windows. The default name of the file is `dfadbat1.prefs`, and its default location is in the directory where Creator is installed.

- ***.color**

  Loads the Color palette to be used for the project. You specify a directory path and file name (must have a `.color` suffix) when you save the file.

- ***.materials**

  Loads the Material palette to be used for the project. You specify a directory path and file name (must have a `.materials` suffix) when you save the file.

- ***.txt**

  Loads the Texture palette to be used for the project. You specify a directory path and file name (must have a `.txt` suffix) when you save the file.

- ***.lts**

  Loads the Light Source palette to be used for the project. You specify a directory path and file name (must have a `.lts` suffix) when you save the file.

- **mgprop.dat**

  Loads the Light String preset file to be used for the project. The default name is `mgprop.dat`. You specify a directory path and file name (must have a `.dat` suffix) when you save the file.

3 Creating a Test Terrain

Before you can process your elevation data it must be converted to MultiGen-Paradigm Digital Elevation Data (DED) format. Next, you import the .ded file into Creator, select the area or areas you want to process, and set the parameters that control the way Creator converts the elevation data to terrain polygons.

Note: Use the Batch tutorial lesson in the Desktop Tutor or consult the Creator online help to guide you through setting your conversion preferences.

Depending on the settings you choose in the *Terrain* window, the same DED file can produce many different terrain databases. When setting terrain processing parameters, you must consider the following factors:

- How many levels of detail does my simulation need?
- Which processing method should I use to convert elevation data to polygons?
- What is my polygon budget?
- Does my hardware support z-buffering? Fixed listing?
- Which map projection is best for my terrain?

Because processing large terrain databases can be somewhat time-consuming, building a small test terrain from the elevation data you plan to use gives you a sample that you can use to refine your settings.

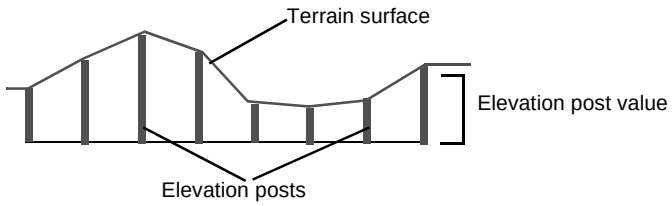
The generated test terrain must be checked for errors at all LODs. You generally get the best results if you generate a series of trial databases, and then test them on the realtime engine. You can make adjustments and corrections to your settings and, in some cases, to your data, and then re-generate the database. These steps may have to be repeated several times to get acceptable results.

Converting Terrain Source Data

Creator creates an OpenFlight terrain database from elevation data. Elevation data is available in different resolutions and formats. For example, DTED Level 1 data represents a one-degree by one-degree cell containing a grid of elevation post values that define the terrain's contours. There are roughly 1201 posts in a cell's latitude. For most latitudes, posts are evenly spaced every three arc seconds (92.43 meters) apart.

The post spacing can be higher, but this is the most commonly available. The number of posts in a cell's longitude varies to maintain consistent post spacing.

Elevation Data Side View



Terrain elevation data must be converted to DED format before the data can be imported and processed in Creator. The most commonly used source data is USGS DEM or NIMA DTED data, but you can also convert other formats and even bitmapped images (for example, satellite data in TIF format) using Creator offline converters, the GView utility (for IRIX systems only), or third party converters such as Okino PolyTrans™.

The Creator offline converters include:

- readdma - converts NIMA DTED data to .ded format
- readusgs - converts USGS DEM data to .ded format
- float2ded - converts data in binary float format to a one-degree by one-degree cell .ded format
- image2ded - converts RGB or intensity images to .ded format

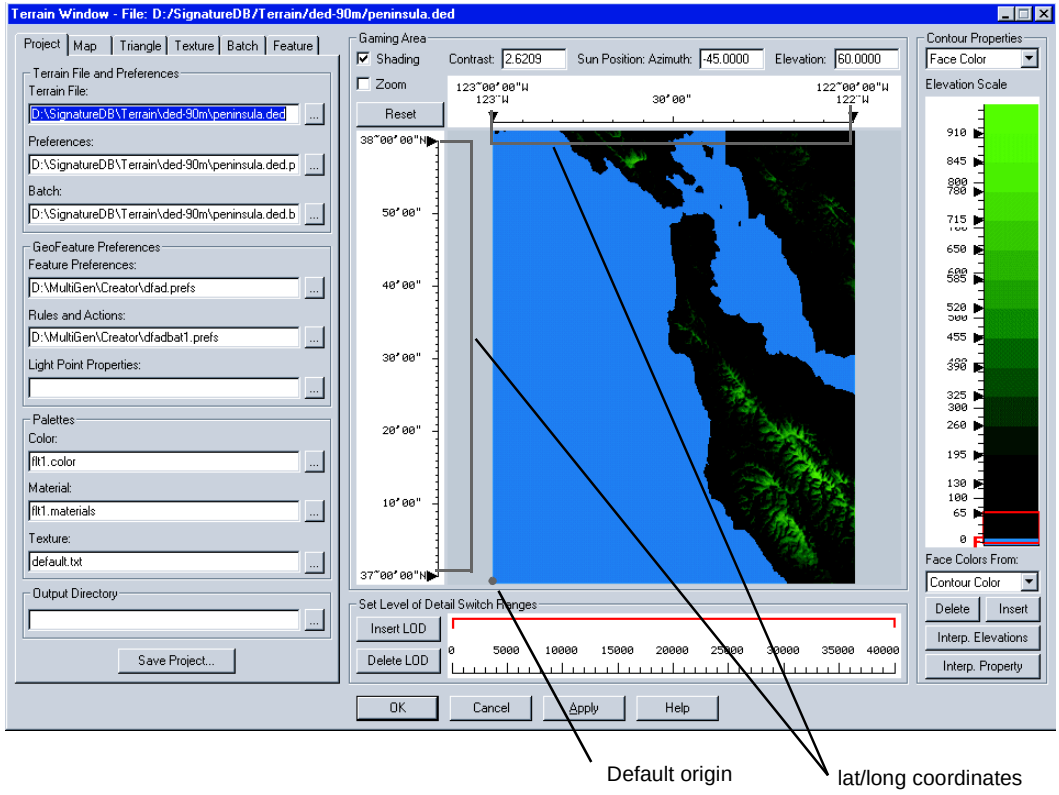
In addition to performing the above conversions, the GView utility also converts DLG data and TIF images to .ded format.

You can use the Creator DED Builder tool to convert one or more elevation files of different file types to a single DED file.

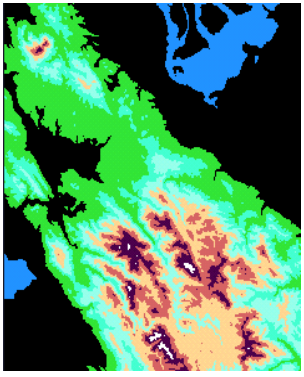
Importing and Viewing the Terrain Data

To import your terrain data in Creator you create a new project. After you select the .ded file to import, Creator loads the .ded file and default preference and palette files. The names of these files are displayed in the Project panel of the *Terrain* window. As the previous chapter mentioned, Creator lets you save project files that load all the preferences and palettes associated with a particular project. This helps to keep your project data organized.

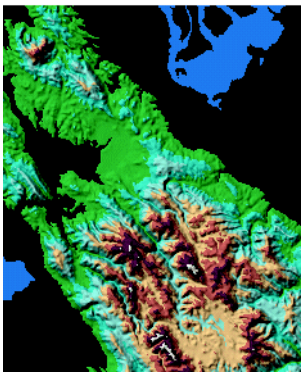
The *Terrain* window displays your imported elevation data and its latitude and longitude boundary coordinates, or meter distances for DED files in UTM format, in a **Gaming Area**. By default, the origin is the lower left corner of the data (the southwest corner). The *z* value of each elevation data post is its real world elevation.



Using the controls at the top of the Gaming Area, you can view your terrain data as a flat shaded image or as a shaded relief image, and you can set the light source elevation, angle, and the amount of contrast and light that is applied to the image to simulate the sun shining on your terrain.

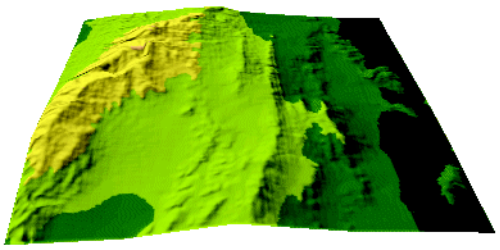


Flat shaded image

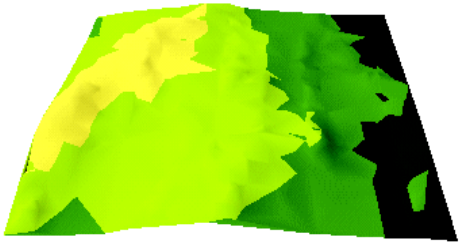


Shaded relief

The shaded image, with its geocoordinate information, can be saved to an image file and then applied to the terrain database as a *geospecific texture* for a more realistic appearance. Alternatively, the contour colors can be applied to the terrain polygons, or used together with geospecific or non-geospecific textures. For more information about applying texture to terrain, see “Using Textures with Terrain” on page 4-1.



Geospeicific texture created from shaded relief and applied to terrain polygons



Terrain polygons with contour color and no geospecific texture applied

Choosing Single File or Batch Process

You can create your terrain with either a single file process or a Batch process. When you use the single file process, you create one single OpenFlight file. This method is generally used for small areas of terrain.

The Batch process divides the terrain into a grid of evenly spaced area blocks. You can select as many or as few area blocks for processing as you want. Creator processes each area block as a separate OpenFlight file, and creates a single master file that contains external references to each of the single files. Batch processing can be used for small areas of terrain, but is particularly useful for large areas of terrain for the following reasons:

- The system doesn't have to process all the data at once - each area block has its own LOD structure and can be paged in differently.
- Each file created is the same size because the area blocks are the same size.
- Edge matching is preserved with adjacent files.
- Each area block can have its own local origin rather than having an origin that is offset from the southwest corner of the elevation data. This prevents the flickering that can occur in the realtime system when offsets become so large that significant digits are truncated.

Selecting a Terrain Area

You can select an area for processing using several methods:

- Drag a box around an area of terrain in the Gaming Area
- Drag the boundary coordinate markers in the Gaming Area until the desired area is highlighted
- Highlight a boundary coordinate marker and enter exact coordinates
- Set **Enable Area Block Processing** in the Batch panel and choose the area blocks you want to process

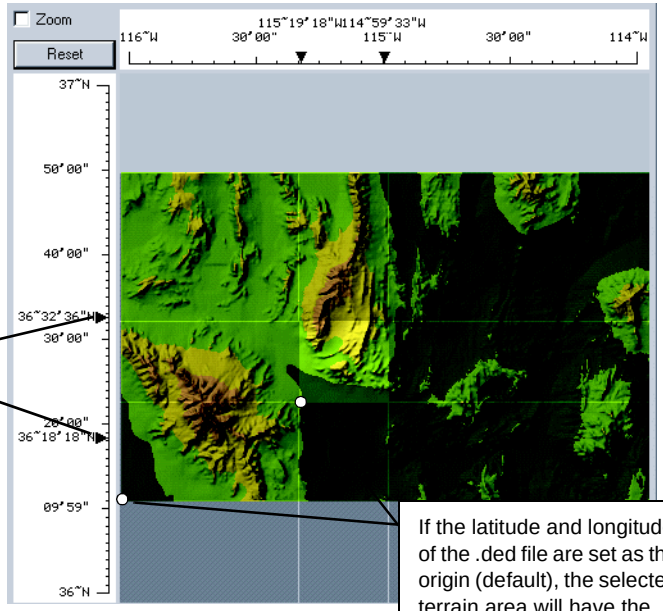
By default, the lower left corner of the entire terrain corresponds to 0, 0 in a Creator database. When processed, your selected terrain area will have the same offset from 0, 0 as from the larger terrain's lower left corner.

Terrain selection:

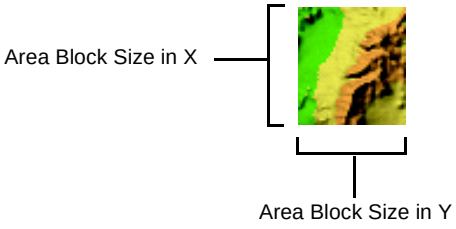
Drag a box around the area

Drag the coordinate markers

Highlight a coordinate marker and enter a coordinate

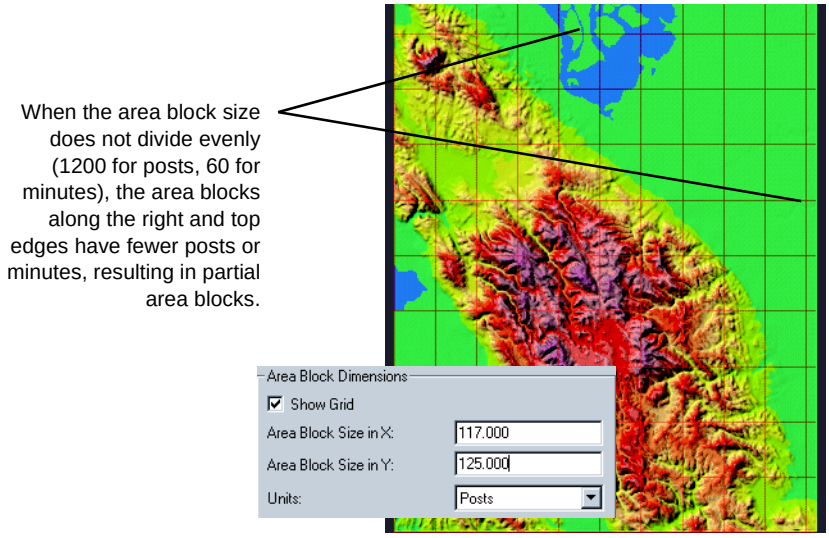


Batch processing breaks the elevation data into tiles based on the number of posts or minutes you specify for the area block size.



You should set the area block *x* and *y* values to a number that will divide into 1200 for posts (roughly the total number of posts in *x* and *y*), or 60, for minutes. This ensures that the area blocks along the top and right edges have the same post or minute count

as the rest of the area blocks. Partial area blocks can result in edge splinters and partial groups which can degrade the performance of your simulation.



Creating Levels of Detail

Creating effective levels of detail (LODs) in terrain processing requires some thought and planning. To make your scene realistic when moving through it, accurate switching distances are important. You must also consider the amount of terrain that will be switched in as the eyepoint moves, whether there are features on the terrain, and how much detail your users need to see at each level of detail. Total polygon count is also an important factor when planning LODs.

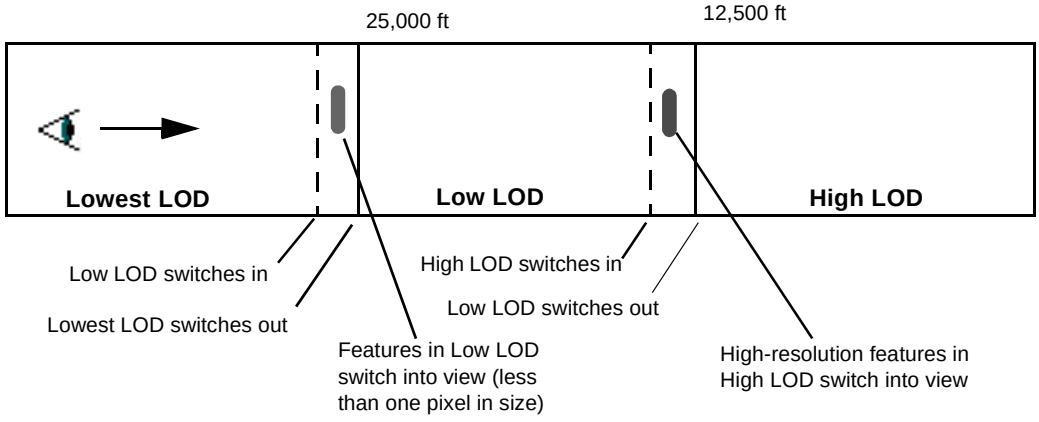
Number of LODs

The number of LODs your database needs will depend in part on the type of simulation you are creating. If your simulation is ground-based driving or a walk-through, you may only need a single LOD, or possibly two. But if your simulation is low-flying with a ground target, you will probably need several LODs.

Another factor that will influence the number of LODs you create is your total polygon budget. The total polygon count between adjacent LODs should not differ by more than approximately 25%. When you allocate too few polygons to lower LODs, Creator must concentrate the vertices needed for edge matching along the edge of the LOD. This results in long, slivered polygons. Large differences in the polygon count between LODs can result in poor edge matching and edge slivers that may be visible when the LODs switch.

Switching Distances

A general rule for setting initial switching distances is to make the next highest LOD switch in at half the distance of the lowest LOD, and so on. For example, the lowest LOD might switch out at 25 000 feet, the next lowest LOD might switch in at 25,000 feet and switch out at 12,500 feet, and the highest LOD might switch in at 12,500 feet. You can also set the transition distance to overlap a bit in order to decrease the appearance of features popping in and out of the scene. For example, the next lowest LOD might switch in at 25,200 feet, 200 feet before the lowest LOD switches out.

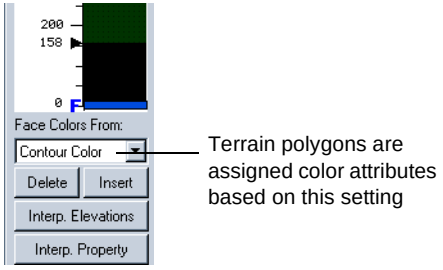


There are more accurate ways to calculate switching distances, but this method will give you acceptable results with a minimum amount of calculation.

Be sure your LOD switch range covers at least as much area as the size of your area block. If the switching distance is smaller, your LOD may switch out too early or may not switch in at all (see “Edge Slivers” on page 3-19).

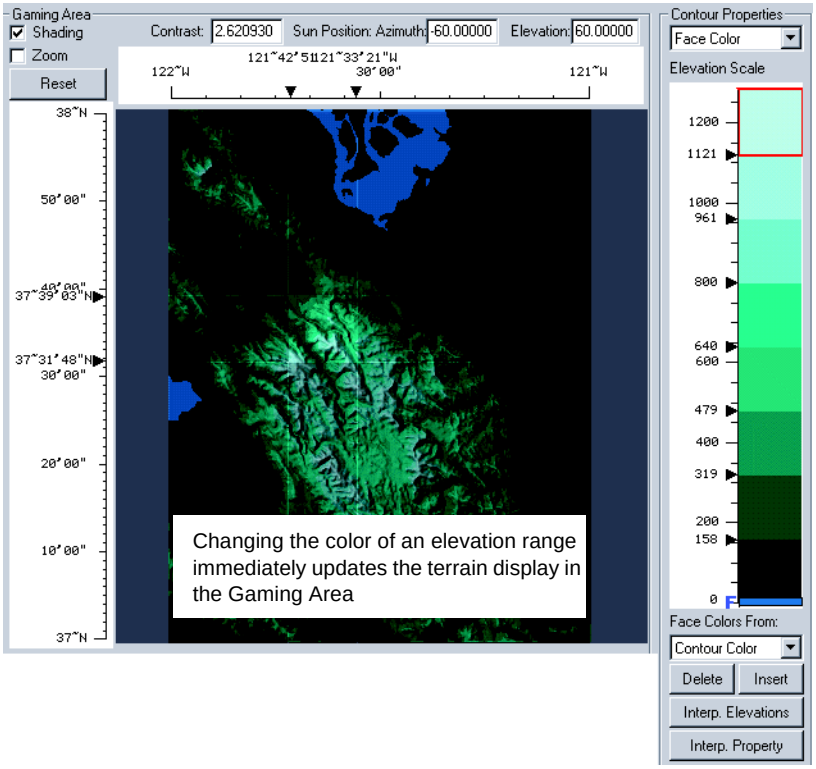
Setting Contour Properties

Using Contour Properties settings in the *Terrain* window, you can assign colors, texture patterns, and material attributes that will be applied to the terrain polygons when they are processed. Any properties that are applied to a single polygon will be blended with each other. For example, if you have specified red for one of the color contours and applied a grass texture to the same contour, the grass color will be blended with the red polygon color. You can apply the face colors in the contour palette to the terrain polygons, or use the colors strictly for viewing the terrain in the *Terrain* window. If you do not want to apply face colors from the contour palette to your terrain, set **Face Colors From** to **Texture Color** rather than **Contour Color**.



You can increase or decrease the elevation ranges covered by a single color by changing the Elevation Scale (see the following figure). When you change the color of an

elevation range, you can see the results on the terrain map displayed in the Gaming Area.



To change the elevation ranges denoted by each color, click Insert or Delete, or drag the markers to a new setting

To change the color applied to a range of elevations, double-click a color band to open the Color palette, and choose a new color

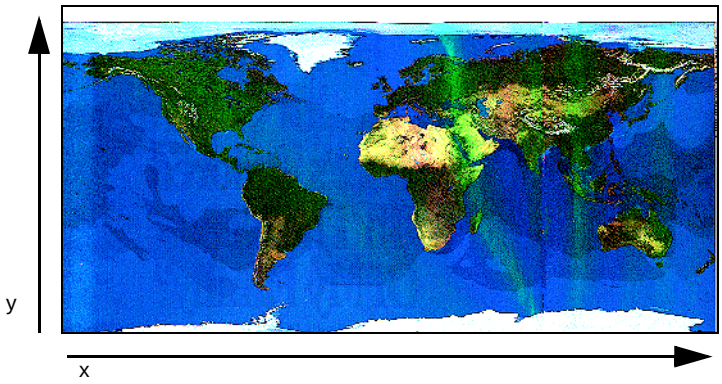
Choosing a Map Projection

The **Map** panel in the *Terrain* window has a number of choices. What do these mean, and how do you know which is the right one for your purpose?

Map projections can represent the earth as a flat surface, or can closely represent the curvature of the earth. *Ellipsoids* are a set of control points that the map projection method uses to define the 3-dimensional shape of the earth. Because the Earth is not a perfectly round ball, different projection methods and ellipsoids have been developed to compensate for the Earth's irregularities at certain places. Terrain near the equator needs a different projection than terrain near either pole.

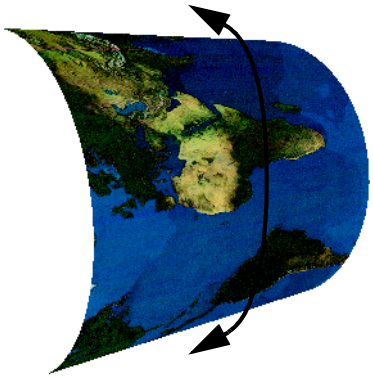
The map projection and ellipsoid method you choose will depend on your terrain's location in the world, the size of the area being covered, and its shape. The goal of choosing a map projection and ellipsoid setting is to introduce the least amount of distortion into your generated terrain. Creator provides five map projections and four ellipsoids to from which to choose.

- **Flat Earth** treats latitude and longitude coordinates as *x* and *y* coordinates. This results in a rectangular database. Flat Earth projections are generally used for databases which run more east-to-west and are not too close to either pole.



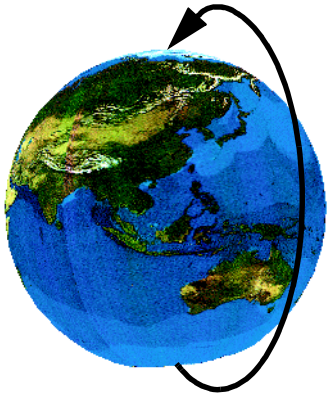
- **Universal Transverse Mercator** (UTM) works well for databases that run more north-to-south, or along polar routes. UTM divides the earth into six degree longitudinal zones with a central meridian. This map projection is most accurate along

the meridian, with distortion increasing as you move farther east or west of the meridian.



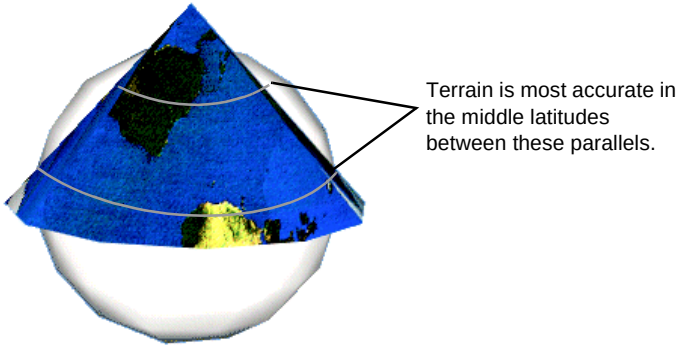
Most accurate along the center meridian, becoming less accurate as you move east or west

- **Geocentric** is a round earth projection where z radiates from the center of the Earth through the North Pole. This projection is used for large area databases, or for simulating the earth's curvature.

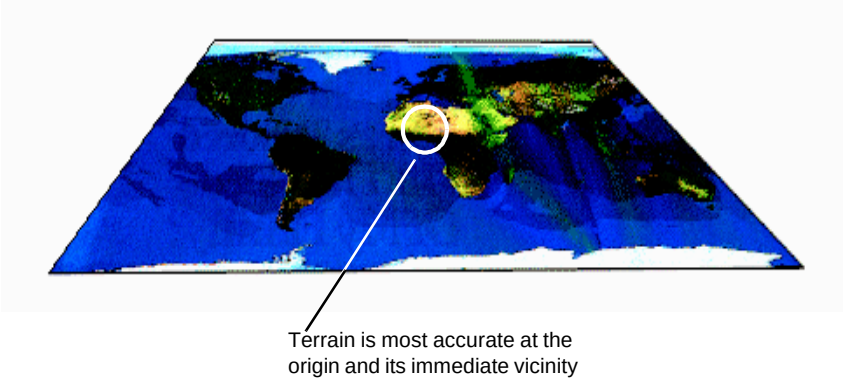


- **Lambert Conic Conformal** uses two standard parallels as boundaries. Between the boundaries, the latitude lines are perpendicular to the longitude lines. This projection is most accurate for high latitude flights over areas located in the middle lati-

tudes between 84 degrees North and 80 degrees South. Distortion increases closer to the poles.



- **Trapezoidal** is an azimuthal projection that is most accurate at its central point, with distortion increasing as the distance increases from this point. This projection method is best for small (one degree by one degree or less) databases.



Ellipsoids are mathematical formulas that define the shape of the Earth, which bulges in the middle and is flattened at the poles. Cartographic data uses several different ellipsoids. For example, the ellipsoids used primarily for mapping data in the United States are WGS 1972 and WGS 1984. Ellipsoids are used in conjunction with projections to convert real world locations to places on a map. All maps specify the earth ellipsoid best suited for the area.

Note: When you apply a geospecific texture to your terrain, you must make sure it has the same projection and ellipsoid as your terrain or it may not map correctly.

Choosing a Processing Algorithm

Creator terrain algorithms provide different methods for converting elevation data to the polygons that form your terrain database. During conversion, only a very small percentage of the elevation posts in your data become the vertices of the polygons that form your final terrain.You choose the algorithm and settings that will give you the best terrain representation that will run efficiently on your target realtime system.

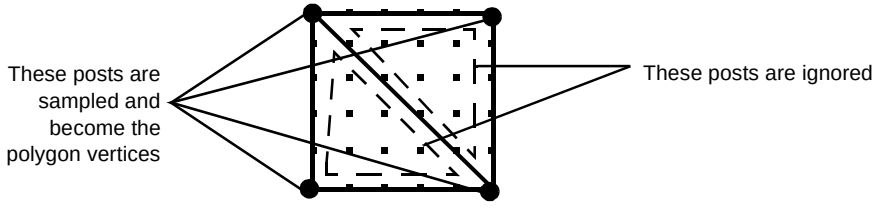
The *Terrain* window's Triangle panel offers several algorithms for converting elevation data to terrain.The conversion algorithm you choose will depend upon the type of hardware on which your database will be running. Factors to be considered include processing speed, type of edge matching supported, tri-stripping ability, polygon limitations, and BSP or Z-buffer compatibility. The following table describes each algorithm's advantages and limitations and the type of edge matching the algorithm uses. You can use this table as a general guideline for choosing the best conversion algorithm for your needs.

Algorithm/Edge Matching Type	Advantages	Limitations
Polymesh / No walls	• Triangle-strips well • No inherent limitation on the number of LODs created • Exactly predictable - no variation in terrain skin, resulting in efficient paging • Creates terrain quickly • Processing is not vertex limited • BSP-compatible	• May overtessellate flat areas • Can produce edge gaps between LODs • No error tolerance testing

Algorithm/Edge Matching Type	Advantages	Limitations
Polymesh / Walls	• Exactly predictable - no variation in terrain skin resulting in efficient paging • Creates terrain quickly • No inherent limitation on the number of LODs • Triangle-strips well • Processing is not vertex-limited • BSP-compatible	• May overtessellate flat areas • Can produce artifacts (discontinuity) along edges between LODs • No error tolerance testing • Walls do not tri-strip • Terrain with extreme variations in elevation are not represented as well as with other algorithms
Polymesh / Irregular Mesh, no walls	• No inherent limitation on the number of LODs created • Processing is not vertex limited • Error tolerance testing • Triangle-strips well • More accurate terrain description for the same number of polygons • Exactly predictable	• Slow processing speed • Not BSP-compatible
Delaunay / Exact Edge Delaunay / No Matching	• Error tolerance testing • Best terrain fit for the least amount of polygons • Edge matching between LODs	• Polygon limitation • Unpredictable number of polygons • Creates short triangle-strips • LOD edge matching can result in large polygons and slivers

Algorithm/Edge Matching Type	Advantages	Limitations
Terrain Culture Triangulation (TCT)	• Error tolerance testing • Feature pre-projection • Pathfinder roads are drivable • Best terrain fit for the least amount of polygons	• Polygon limitation • Unpredictable number of polygons • Creates short triangle-strips • Single LOD
CAT / Morphing	• Error tolerance testing • Best terrain fit for the least amount of polygons • Edge matching between LODs	• Only runs on systems running SGI Performer • Very CPU-intensive • Features must be cut into terrain with adaptive attributes added for morphing between LODs • Geospecific textures must be at least as large as the terrain database because cut-in is not possible

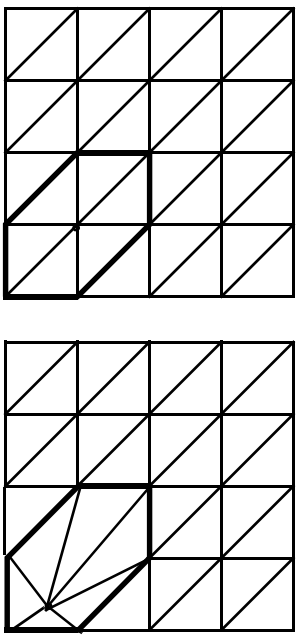
Polymesh samples every *n*th post specified by the **Post Sampling Rate**. For example, if you choose the Polymesh algorithm and define a **Post Sampling Rate** of 6, Creator creates a polygon based on the value of every 6th post in x and every 6th post in y and discards the post values in between. The sampled posts become the polygon vertices.



Terrain polygon pair

Note: To avoid partial terrain polygon pairs, set the **Post Sampling Rate** to a value that will divide evenly into the number of posts that each area block covers. For example, in a DED file of 1201 posts with an area block size of 120 units, there are 10 posts per area block (1200 / 120=10, so you could use a Post Sampling Rate of 10, 5, 2, or 1.

Irregular Mesh uses the **Post Sampling Rate** a little differently. It creates a non-rectangular polygon, samples each post lying within the polygon for the maximum deviation, and then moves the polygon's midpoint to the new location. The result is irregularly shaped triangles that describe the terrain more accurately than regular polymesh for the same number of polygons.



Note: Keep in mind that as the **Post Sampling Rate** increases, more posts are tested, and as more posts are tested, the time required to generate the terrain increases.

Delaunay samples each post in the selection. Additional posts are added to the triangulation when **Ridge and Valley Detection** or **Preserve Coastline** is set, in order to more accurately define the terrain's contours. Delaunay starts with the lowest LOD and incorporates its vertices into the next highest LOD, ensuring that vertices match when the LODs switch.

Terrain Culture Triangulation (TCT), a constrained Delaunay algorithm, allows pre-projection of features – features are projected first, and then the terrain is built around them. You can also pre-project only the features, without the terrain. TCT can only be used as a batch process.

Continuous Adaptive Terrain (CAT) provides smooth LOD transitions without the popping that can occur with Polymesh or Delaunay. CAT databases can only be run on systems that are running SGI Performer. CAT uses a modified version of the Delaunay algorithm for tessellation.

Testing the Terrain

Once you create your test database, you should open it in Creator and carefully check it at all levels of detail and from all sides. Incorrect terrain settings can cause anomalies such as the following:

- Walls or gaps in a Polymesh terrain can appear along the boundary between adjacent area blocks when switching LODs.
- Sliver polygons – long, thin, triangular polygons – along the edges of an area block that has been generated with Delaunay can occur when there are insufficient vertices along the edges for proper edge matching.
- Partial groups can be formed along the right and top edges of a Polymesh terrain

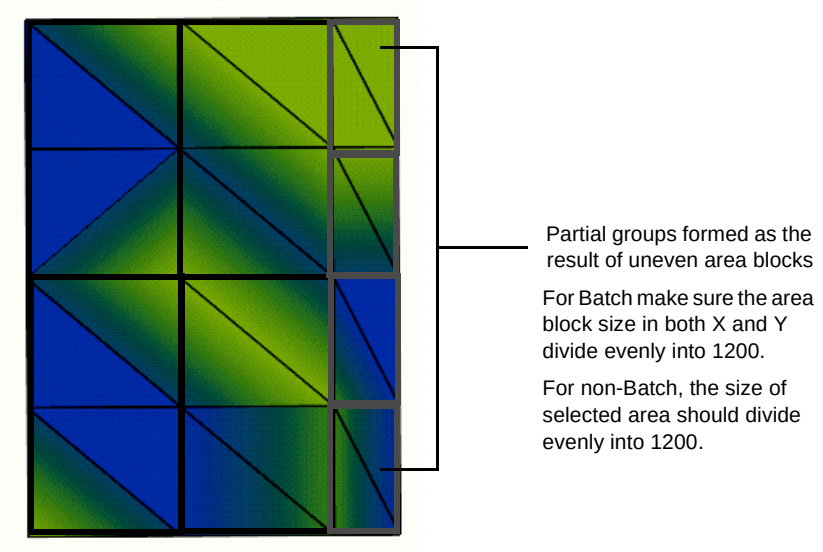
Anomalies such as these can negatively affect your simulation's performance in the realtime system. They are explained in the following sections.

You should also run your test database in the realtime system. If your polygon count is too high or your system spends too much time culling and rendering each frame, your performance will suffer. You may need to adjust the settings and regenerate the terrain. Adjustments include increasing or decreasing your polygon count, reselecting the area you want to process, or changing other parameter settings.

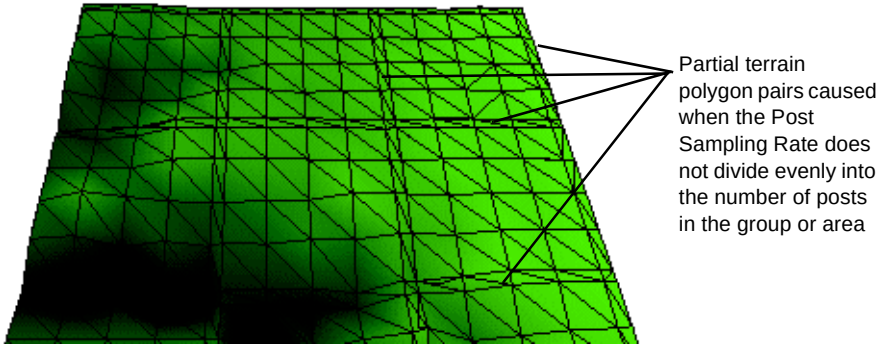
Partial Groups

When processing terrain the area selected for processing must be divisible by the number of posts or minutes. This ensures that whole groups are processed. When this

is not the case, partial groups may be generated along the north and east edges of your terrain skin.



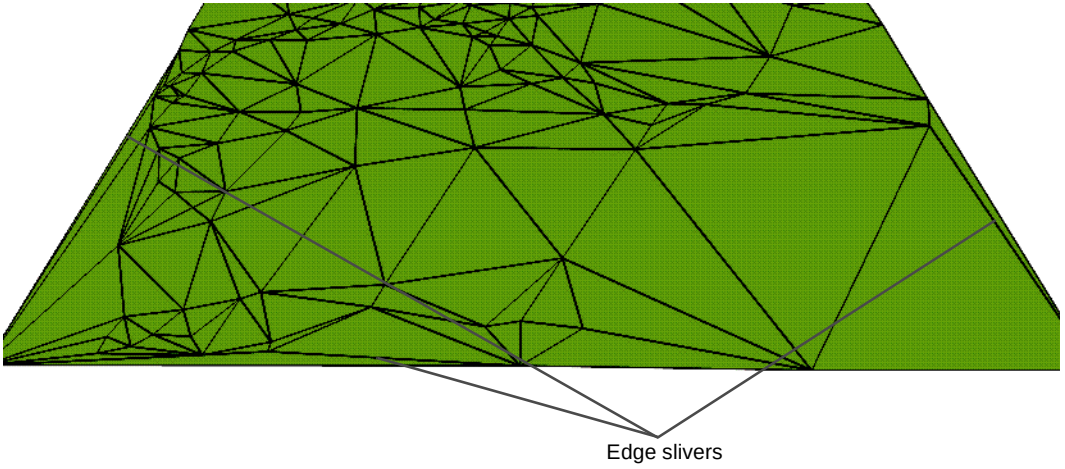
When using Polymesh, partial terrain polygon pairs can result in slower performance.



Edge Slivers

Edge slivers are very thin polygons that sometimes occur along the edges of a Delaunay terrain skin. Delaunay forces error matching by assigning edge vertices in

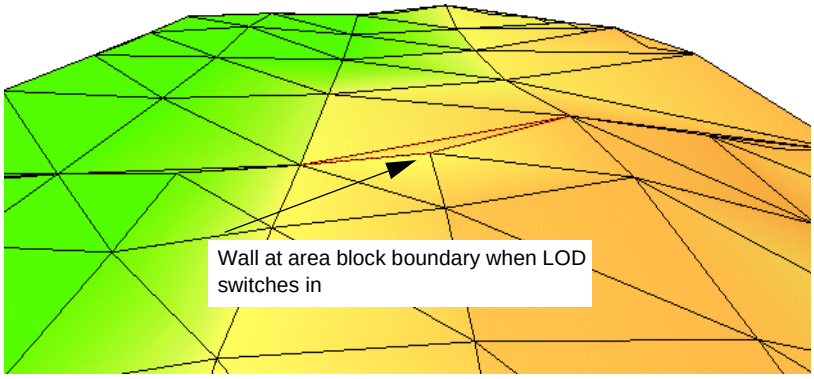
the lowest LOD, and then retaining them in higher LODs. Therefore, no vertices are added to the edges as higher LODs are built. Because higher LODs are allocated more vertices, some of them may fall very close to the edges. When two edge vertices are very far apart and the third vertex that forms the triangle is close to the edge, you get sliver polygons, as show in the figure below.



To solve this problem, allocate more polygons to the lowest LOD to get more edge vertices. This reduces the number of polygons you can assign to higher LODs, but it produces a more representative terrain overall, with smoother transitions.

Walls and Gaps

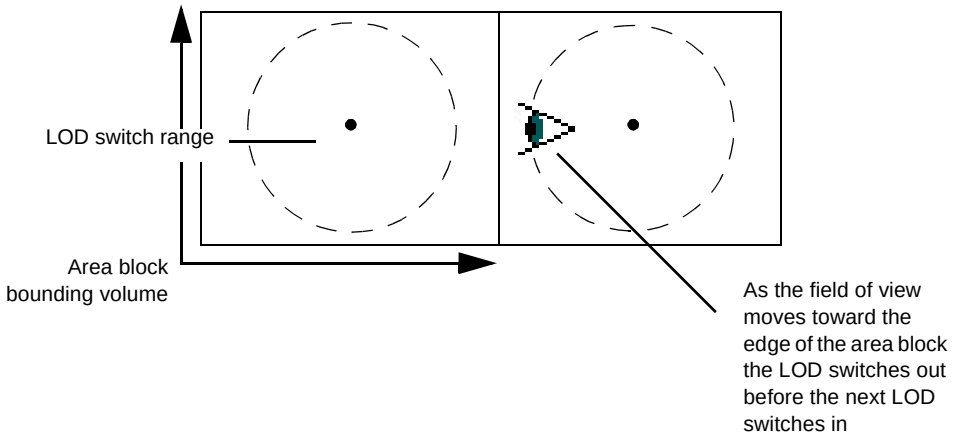
Because Polymesh doesn't force edge matching, you have the option of enabling walls along area block boundaries. Walls can occur along the boundaries during LOD switching.

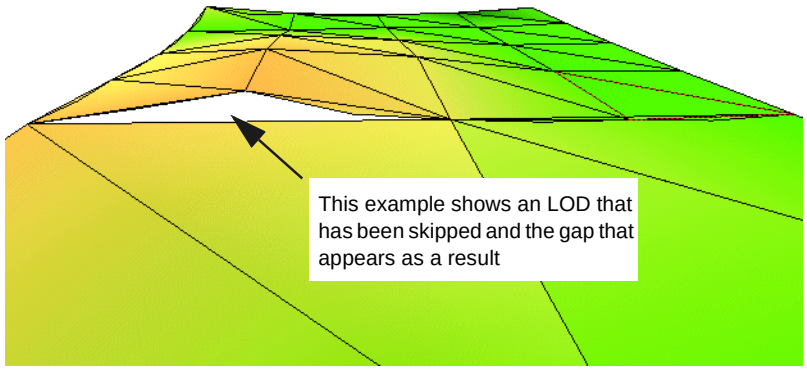


If your simulation is high-flying, the appearance of walls may not present a visual problem. If the walls are visible, however, set your switching distance so that the LOD switches in when the eyepoint is far enough away so that the wall is not apparent.

Gaps can occur when switching LODs if the area blocks (in Batch Polymesh) are larger than the area covered by the switching distance. This can cause an LOD to switch out before the next LOD switches in, or can cause a level of detail to be skipped entirely.

The feature toward which you are moving may disappear from view if the LOD switches out!

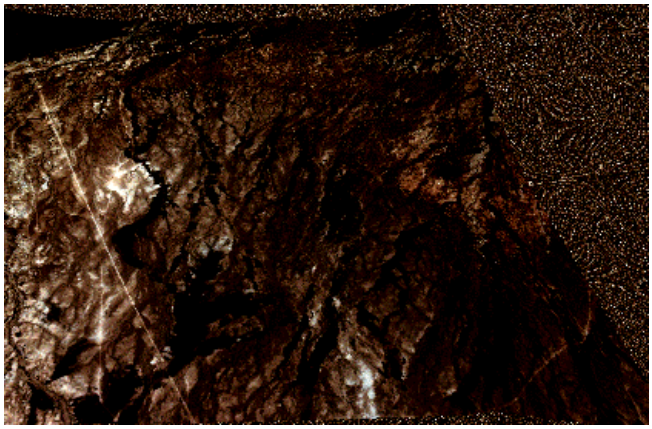




To solve this problem, you must increase the switching distance or regenerate the terrain with smaller area blocks.

4 Using Textures with Terrain

Textures add visual realism to your terrain database. A detailed texture pattern can be used in place of actual features in lower levels of detail when the database scenario doesn't require interaction (because collision detection can't be programmed into textures). This can save polygons that can be used elsewhere in your simulation.

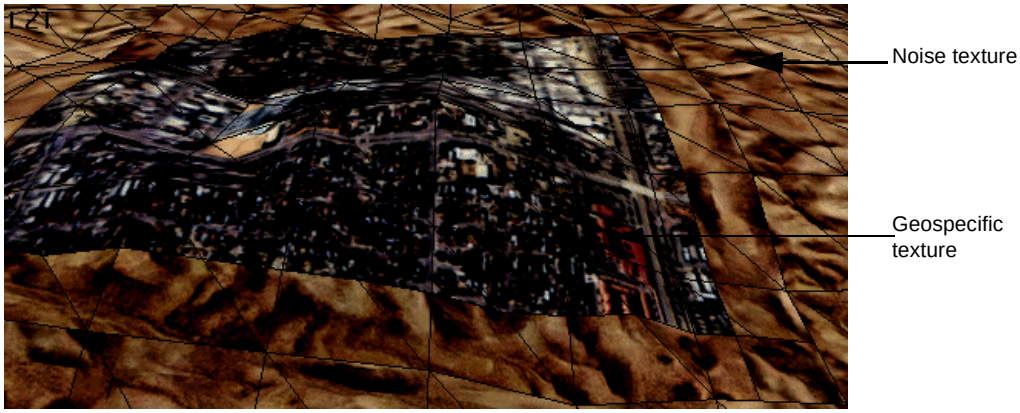


Textures can be applied to terrain in the following ways:

- Specify texture patterns in the *Terrain* window's Contour palette. With this method, you can choose different texture patterns for each elevation level.
- Choose a geospecific texture pattern in the Texture panel of the *Terrain* window. Geospecific textures are texture patterns that have attributes which match selected texels in the image to specific places in the world. The texture's geocoordinates must match or fall within the terrain's latitude and longitude in order to be applied. Only one geospecific texture can be applied to a terrain.

  You can also apply texture in the Contour palette together with a geospecific texture. For example, you can apply a geospecific texture mapped to a city area that will be the main focus of a flyover, and apply a "noise" texture (a general

pattern applied to areas outside the simulation's area of interest) to the rest of the terrain.



- Use an indirect texture to map color, texture, and material properties to specific colors in a texture pattern. You can apply an indirect texture to USGS material class usage maps or satellite photographs, and create richly textured landscapes from relatively low-resolution images. You can also use indirect texture to paint imaginary worlds for a gaming environment (See "Indirect Texture" on page 4-7 for more information about applying indirect textures).

If you apply color or material properties to the terrain faces, their properties will be blended with the texture.

Geospecific Texture

A *geospecific texture* contains geographical attribute information (the image's latitude and longitude) that indicates where the texture pattern belongs. Typically, geospecific textures are used to apply satellite photographs to large terrain areas. Because Creator correlates geospecific attributes with database coordinates, a geospecific texture can be applied automatically during terrain conversion, or you can apply it to an existing terrain database.

Real World Size, the size of the texture pattern in pixels, is used to scale the image in some systems. Check that the texture's real world size is not set to 0. If it is, it will not be mapped.

Geospecific textures can be applied in three ways:

- Specify the geospecific texture in the Texture panel of the *Terrain* window. The texture will be applied at the time the terrain is generated
- Use the **GeoPut Texture** tool

  This tool, which is located in the **Texture** toolbox, applies a geospecific texture pattern to an existing terrain database that has the same geospecific coordinates. The texture can be applied to a single level of detail, or to all levels of detail, depending on what has been selected. To apply geospecific textures to all levels of detail, select the hidden LOD nodes in the Hierarchy view, and choose Display in the Hierarchy tools to display all the levels of detail. Then select all the faces to which you want to apply the texture, and click the **GeoPut Texture** tool.
- Use **Batch GeoPut**

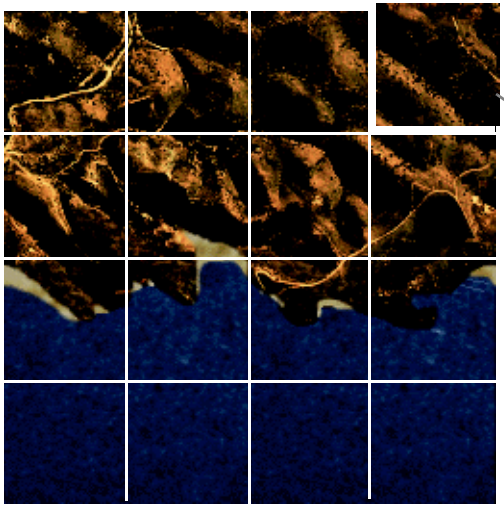
  This **Terrain** menu utility applies one or more geospecific textures to a specified set of OpenFlight files. The textures are applied to all levels of detail. The OpenFlight files do not have to be open, and the textures do not have to be loaded in the Texture palette in order to be applied.

Clip Textures and Mipmaps

If your realtime system supports clip textures and mipmaps, you can create them with the Mgmosaic utility and then apply them by loading the original texture into the Texture palette and applying it.

Clip textures provide an efficient way to manage texture overhead. Clip textures are a set of small, square images created from a large geospecific texture. Because only a

portion of the large texture is visible at any one time, the realtime system pages a clip texture into memory only when it is needed, thus saving system resources.



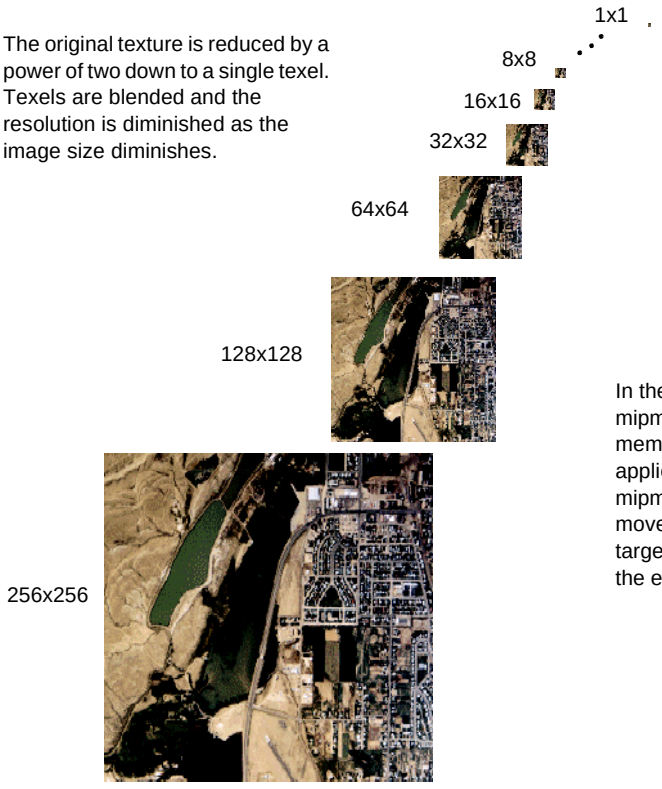
Clip textures are created by dividing a geospecific texture into even-sized tiles.

Each tile, or clip texture, is paged in and out of memory as needed.

Mipmaps are a series of low-resolution versions of the base image, which the realtime system swaps in when the polygon's size is smaller than the original texture size.

Mipmap Textures

The original texture is reduced by a power of two down to a single texel. Texels are blended and the resolution is diminished as the image size diminishes.



In the realtime system the entire mipmap stack is loaded into memory. The smallest mipmap is applied and then successively larger mipmaps are applied as the eyepoint moves closer to the ground or the target object. The reverse is true as the eyepoint moves away.

High Resolution Insets

High resolution insets are applied to areas of interest. These insets have a higher resolution than the base image. When you zoom in on an area where a high resolution inset has been applied, the area appears in detail, as opposed to the areas covered by the base image, which become less distinct. High-resolution insets are stored and processed as a partially populated level in the clip stack.

You create high resolution insets with the Mgmosaic utility. Insets are assigned using Mgmosaic's **Det** data control files, which are created by re-selecting **New Data Control File** for each higher resolution needed. High resolution insets are only supported on SGI InfiniteReality® systems running Performer® 2.1 or higher.

Detail Textures

Detail textures let you create successively more detailed texture images from a single image. These textures, when applied to a database with several levels of detail, give the appearance of greater and greater detail as the eyepoint moves through higher levels of detail. At a low level of detail, a low resolution mipmap of the texture image is applied. As the levels of detail increase, the realtime system blends in higher and higher resolution versions of the image. Detail textures appear to increase the resolution of a texture without loading redundant information into texture memory.

You create detail textures by editing your texture pattern's attributes (See the Creator online help for instructions). Although you can view the effects of a detail texture in realtime, you cannot see them in Creator.

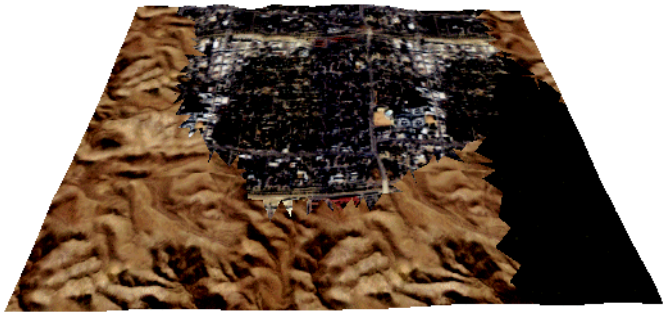
Note: Detail textures are supported only on SGI systems and require an extension to OpenGL®.

Indirect Texture

Indirect texture lets you assign texture, color, and material properties to terrain using a bitmapped image and an ASCII file that keys each color in the image to a specific group of properties.

Indirect Texture maps color, texture and material properties to your terrain by the use of a text file that assigns the properties to each color in the texture.

When two or more vertices of a polygon fall on the same color, that color is used to assign properties to the polygon.



;Indirect texel color				Texture	Texel	Texel	Color	Matl.	LOD
;R	G	B	A	index	sizex	sizey	index	index	
0	255	255	255	0	20.0	20.0	127	-1	0
220	96	0	255	2	20.0	20.0	127	-1	0
0	135	64	255	3	20.0	20.0	127	-1	0
0	0	255	255	1	20.0	20.0	762	-1	-1
220	96	0	255	-1	20.0	20.0	3327	-1	-1
0	135	64	255	-1	20.0	20.0	62175	-1	-1

Colors in Indirect Texture

Properties to be assigned

= The Texel size x and Texel size y values override the Texture attributes' Real World values.

= A -1 in the Texture, Color, or Material index column indicates the absence of that property.

= A -1 in the LOD column indicates that properties listed on that line are to be used in all LODs that are not otherwise specified.

= A semicolon indicates that the line of text that follows is a comment. A space or tab separates columns, and a carriage return separates lines.

You must create two files before applying an indirect texture:

- An indirect texture image file that defines where specific colors, textures, and materials are to be applied.
- An ASCII text file that defines how the colors, textures and materials will be applied to the polygons in the terrain database you are about to create.

  Each color in the indirect texture refers to a line in the ASCII file that calls out a specific texture, color, material, and LOD to be assigned to that area of the terrain.

Both the image file and the user-defined ASCII file must be located in the same directory. The image file must be in rgb or rgba format. The name of the ASCII file must be the same as the image file, and must have a `.indirect` suffix.

Once you create these files, you apply them by selecting the **Indirect Texture** checkbox in the Texture panel in the *Terrain* window. When this checkbox is set, the current texture file in the Texture palette is used as the indirect mapping image.

The indirect texture is mapped to the exact extents of the elevation data file that is being imported and is projected onto the newly-created terrain polygons. When two or more vertices of a polygon fall on the same color, that color is used to assign properties to the polygon.

5 Projecting Features

Once you successfully create a test terrain, correct any anomalies, and are ready to generate your terrain, you also need to gather your feature or vector data and decide what features your simulation needs. Feature data can be obtained from many sources and is generally in DFAD, DLG, or Vector format. Before you import this data into a Creator database, you must first convert the data to the Creator DFD format using one of the Creator offline converters, Gview (for SGI IRIX), or a third party converter such as Okino PolyTrans™.

Features are classified as *points* such as signs, individual buildings, trees, cities, and airports; *linears* such as rivers, roads, and fences; or *areals* such as lakes, cities, and forests.

The general sequence of tasks that are performed when adding features to terrain are as follows.

1. Gather raw feature data and convert it to DFD format.
2. Decide whether you want to pre-project any features and prepare the necessary cutout or footprint files.
3. Import the feature data into your terrain database using the GeoFeature menu, or process your terrain as TCT and choose Project Culture Only (no terrain).
4. Correct overlapping and intersecting features using Creator editing tools.
5. Add or reduce features, if necessary.
6. Save the edited feature data to a new cutout or footprint file, or export the data to a new DFD file to be used for projecting the features.
7. Set up feature preferences, which define how each of the features will appear when then are projected.
8. Set up feature projection preferences, including library substitutions.
9. Set up Rules and Actions if you will be post-projecting features using the Batch terrain processing option, or set up Apply Rules to Files to batch-project features on existing terrain.
10. Project the features.

These tasks are explained in detail in the rest of this chapter.

Deciding What to Project

Deciding which features to import and project depends upon the type of simulation you are creating. If you are flying over terrain, you may be able to use a texture for most of the high detail and only project features in certain areas of interest or interaction. Conversely, if you are doing a ground-based simulation where the eyepoint moves through the scenery, you need to consider which features will be coming into view that must be viewed in detail.

You can eliminate the features you do not want or need in your simulation by performing an attribute search from the *Import DFD* dialog box and importing only the features that meet certain criteria. You can also reduce the number of linears and/or vertices by setting parameters in this dialog box, or you can choose to do this on a case-by-case basis once the features have been imported. See the Creator online help for complete instructions on importing feature data.

Because you project features in layers, you must give some thought to the features that you import in each layer. Project your layers from bottom to top. For example, project streams, forests, and roads before you project buildings. This ensures that roads and rivers don't run through buildings. If you are using Batch terrain processing, Rules and Actions determine the order in which features are projected.

Importing and Correlating Feature Data

Before you can import feature data into Creator and project it, your data must be in Creator DFD format. Creator provides utilities and plugins to convert data in DFAD and VPF formats to DFD.

The geographical location of the feature data you want to project must fall within, or partially overlap, the area covered by the terrain.

You should import your feature data and check it before projecting it. Feature files generally have more data than you will use, and the data can have errors. For example, two areal features may overlap each other. Perhaps your simulation is high-flying and these anomalies will not be noticed, but if your simulation is ground-based driving, you will have to modify your data. Differences in feature placement and detail between a geospecific texture and the feature data also occurs. For example, a river or road feature may not line up with the geospecific texture applied to your terrain database.

You must decide which representation provides the greatest degree of accuracy for your simulation and correct the data accordingly.

There are two ways to check feature data:

- Use the **GeoFeature** menu to import the feature or vector data that will be post-projected, edit the data, and then export the data to a new DFD file.
- Using TCT, select **Project Features Only (no terrain)** in the *Terrain* window's Batch panel.

When you finish correlating the feature data, you can manually project it or, for Batch processing, export the data to a DFD file and project it in the *Terrain* window or use the **Apply Rules to Files** command in the **GeoFeature** menu. See "Feature Projection Methods" on page 5-11 for more information about types of feature projection.

Using the GeoFeature Menu

For Polymesh, Delaunay and CAT terrains, you use the **Import/DFD** command in the **GeoFeature** menu to import the feature data into an existing terrain database. When you import data, the *Import DFD File* dialog box allows you to define attribute search conditions to import only the features that meet specific criteria. You can also eliminate data that does not fall within specified latitude and longitude locations, and set a reduction tolerance for linears and areals to eliminate vertices that are not necessary for retaining the general shape of the feature.

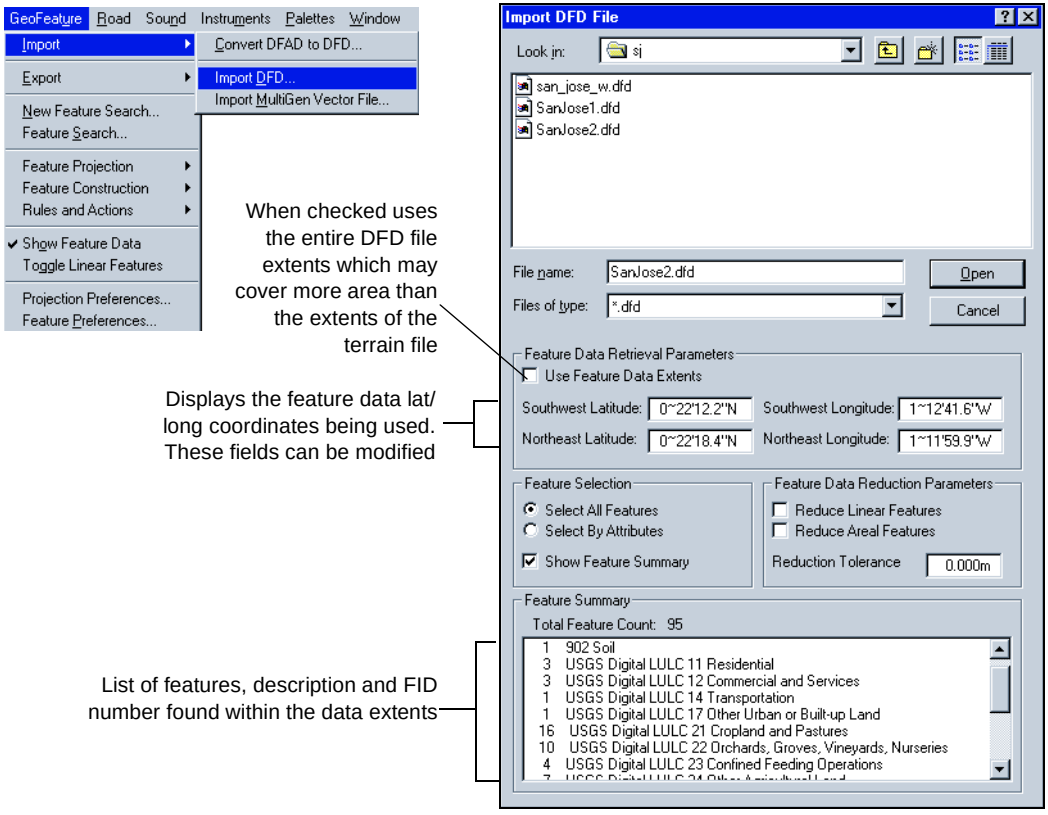
When you import features using the **Import DFD** command, the feature layer "floats" over the terrain. This gives you an opportunity to reduce the number of features you want to project, check for overlapping or conflicting features, correct any errors, and

reduce the number of feature vertices. When you finish editing your feature layer, you can project it, or export it to another DFD file and project it at a later time.

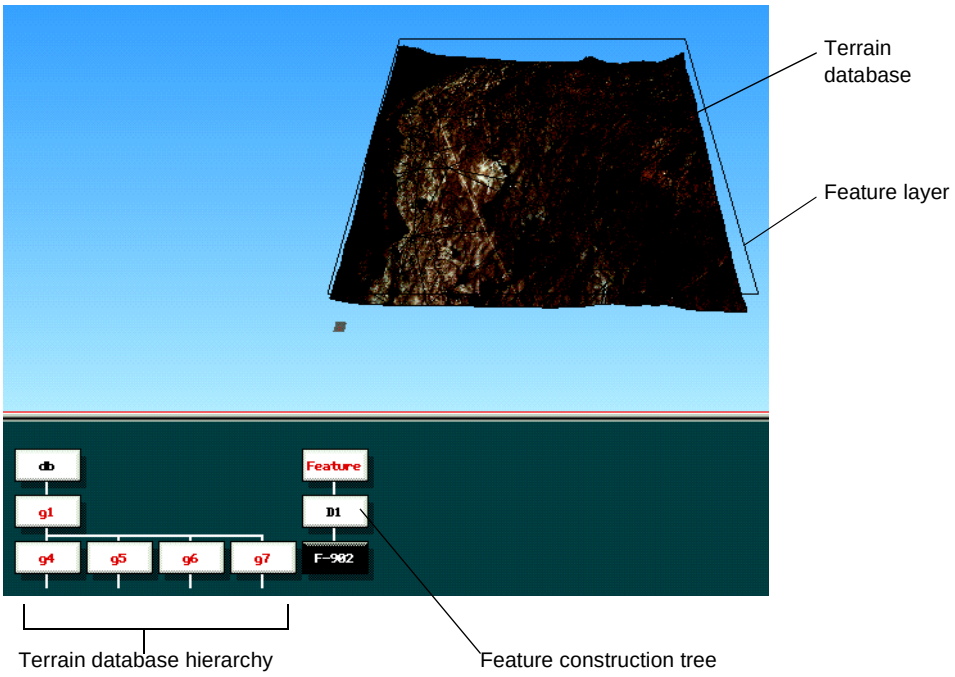
Feature Attribute Searches

When you import feature data using **Import/DFD** in the **GeoFeature** menu, you choose which features you want to import. The *Import DFD File* dialog box displays a list of the features that are located within the latitude and longitude coordinates of your terrain (this area is also called the “extents” of your terrain database). The Feature Summary

lists the feature identification (FID) numbers, a description of the feature denoted by the FID, and the number of features that correspond to each FID.



The imported feature data appears as a layer (the *Feature layer*) that floats above the terrain in the Graphics view, and as a separate single-layer construction tree in the Hierarchy view.

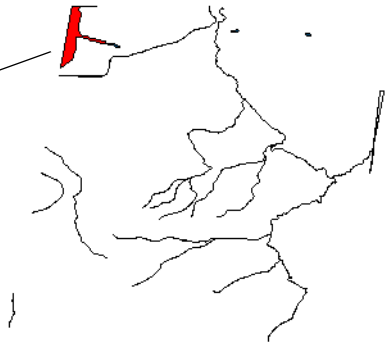


Feature Only Projection

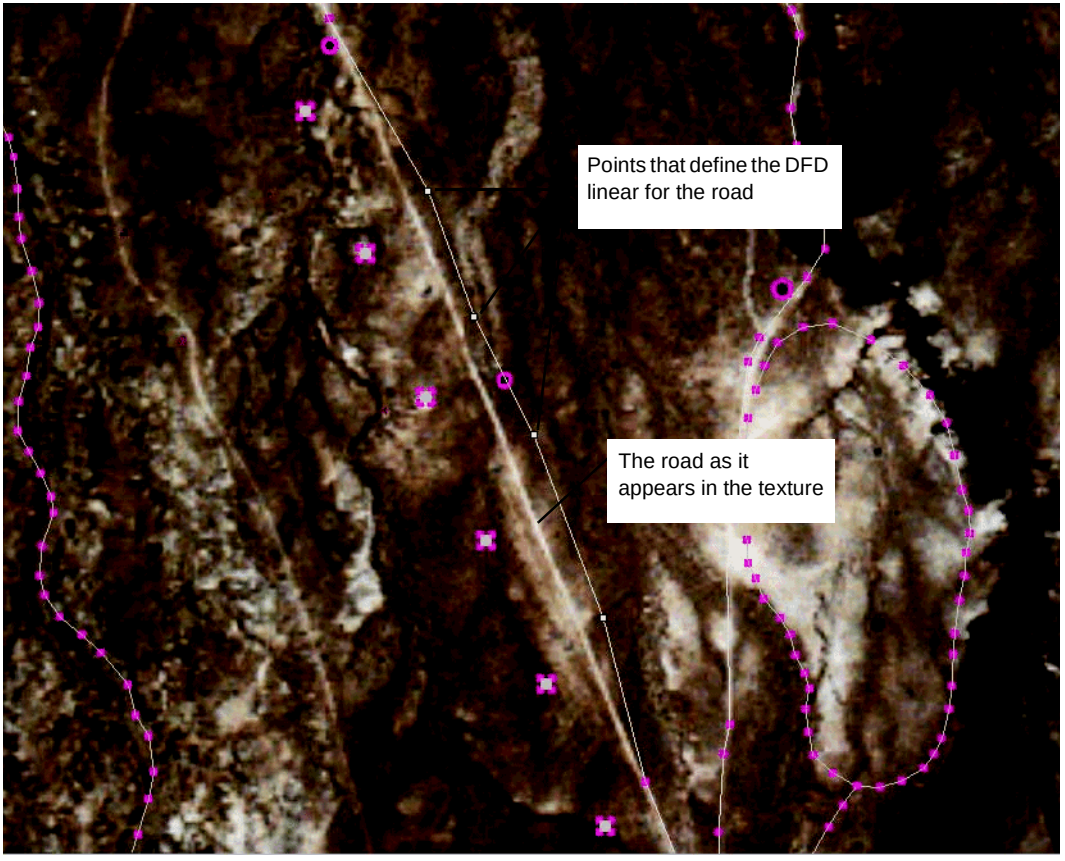
The TCT **Project Features Only (no terrain)** option pre-projects cutout and footprint files but strips out the terrain polygons. You can then edit the feature data and save the modified data to a new file.

Conflicts in the feature data, such as overlapping features or features that are cut off by the terrain tile's boundaries, are flagged in red. You can move or modify the feature in the feature data layer and then export the corrected data to a new DFD file.

Features pre-projected without terrain. The area shown in red indicates a problem with the data; in this case the feature overlaps the boundary of the terrain tile.



For example, in the following figure, the feature data for a road does not match its location in the satellite photo texture of the area. To correct this you must use the **Translate** tool on the road's vertices to align the road correctly with the texture.



Modifying Feature Data

You can use some of Creator's modeling tools to change a feature's size, shape, or position before you project it. You can also change settings in the feature's attribute window, or use the **Reduce Feature Vertices** tool to eliminate some of the vertices and thus reduce the polygon count when the feature is projected. Once you import the feature data, you can use the Creator modeling tools to modify the features in the Feature layer, or change their attributes, before you project them.

There are several ways for you to reduce the number of features:

- When you import data, you define attribute search conditions to import only the features that meet specific criteria. You can eliminate data that does not fall within specified latitude and longitude locations, and set a reduction tolerance for linears and areals to eliminate vertices that are not necessary for retaining the general shape of the feature.
- Once the data is imported, you can use modeling tools to eliminate unnecessary vertices and edges, and reduce the overall polygon count when the features are projected.
- If you project features manually, you can perform attribute searches to project specific subsets of the feature data.
- If you project features in a batch operation, you create Rules and Actions that control exactly which features are projected, and in what order.

Note: Features are not saved with OpenFlight databases until they have been projected. If you have edited features but have not projected them, you must save the feature data separately using the **GeoFeature/Export** menu option.

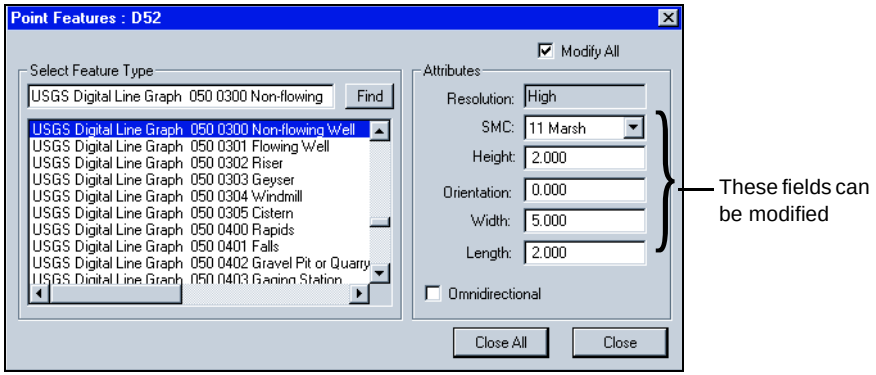
Reducing Vertices

To make feature data more efficient to display, you can reduce the number of vertices in some of the features before you project them. **Reduce Feature Vertices** in the **GeoFeature/Feature Construction** menu applies a dynamic vertex reduction algorithm to objects in the Feature layer. This decreases the number of polygons that are constructed when the feature is projected.

Modifying Feature Attributes

You can modify a feature in the Feature layer by selecting it and making changes in its attribute window. The feature *Attributes* window lists each feature's current attributes.

When you change the value in any field, the results will be visible when the feature is projected.



Using Modeling Tools to Edit Features

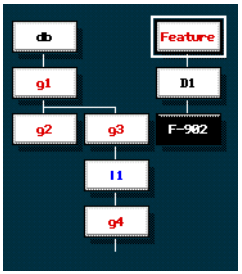
You can resize and reposition objects in the Feature layer with the Creator face and vertex modeling tools. This lets you correct the feature data manually before projecting it into the database or exporting it to a DFD file.

- To select an areal or linear feature in the Feature layer and change its size, shape, and orientation use the **Translate**, **Scale**, **Rotate About Point**, and **Modify Vertex** tools.
- To move point features use the **Translate** tool.
- To change a feature's dimensions and orientation, modify its attributes. Do not use the **Scale** and **Rotate About Point** tools because the polygons in the Feature layer are temporary construction representations; true polygons are not built until you project the feature.

Creating New Features

Feature data does not have to be imported; you can create new features to be projected onto your terrain. To do this you must create a construction layer by choosing **GeoFeature/New Feature Layer** or by selecting a layer as the parent. The Hierarchy view will show a Feature construction tree containing three nodes: the top node is the parent

of the construction group, its child is an Object node that represents the extent of the database soil layer, and the third node is the Face node that represents the soil layer.



To create a new feature, select the Feature node and set it as the parent, then use the **GeoFeature/Feature Construction** menu options to create a new point, linear or areal feature.

Note: Features are not saved with OpenFlight databases until they have been projected. If you have not projected your features you must save the feature data separately using the **GeoFeature/Export** menu option.

Feature Projection Methods

You can project features in the following ways:

- Manually post-project features onto existing terrain using **GeoFeature** menu commands
- Automatically post-project features when generating the terrain as a Batch process using any conversion algorithm
- Automatically pre-project features when generating the terrain as TCT

Manual Post-Projection

You can manually project features onto existing terrain using commands in the **GeoFeature** menu. When features are manually post-projected, you must apply the features to each level of detail individually. In addition to the file or files containing your feature data, you must also set up feature preferences and projection preferences. These preference files tell Creator how you want each feature to appear when it is

projected. If library models are being substituted for certain features, you must also have OpenFlight models available to be substituted. You can manually post-project features as a batch process when you use the **Apply Rules to Files** GeoFeature option.

Automatic Post-Projection

You can automatically post-project features after the terrain has been generated. This is a batch processing option. Creator populates terrain databases as they are created from area blocks by extracting features from DFD files and then projecting the features into each database. When using automatic post-projection, features can be applied to all levels of detail or only to the levels of detail that you specify.

To post-project features automatically, you must still set up feature and projection preferences and have OpenFlight models for substitution, as for manual post-projection. In addition, you must also set up Rules and Actions that tell Creator which features to extract and project at each level of detail, and in which order.

- *Actions* are projections and / or library substitutions that extract features according to attributes.
- *Rules* are lists of actions to be performed for each LOD during block processing. Rules can contain more than one action and are needed because only one rule can be applied to each LOD.

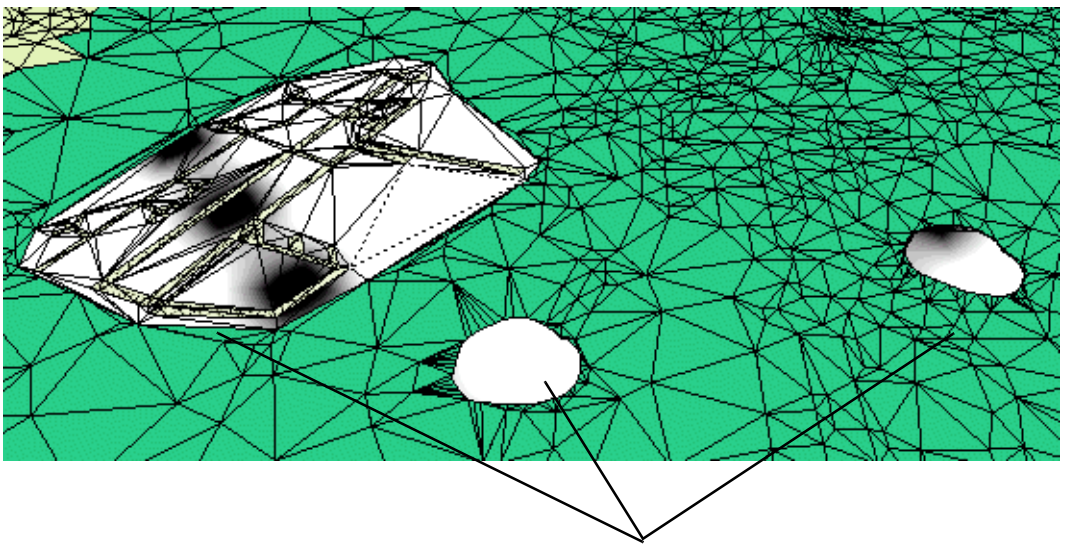
Automatic Pre-Projection

You can pre-project feature data at the time your terrain is being generated if you are using TCT. The TCT option **Project to Terrain Data** in the GeoFeature *Action Definition* window causes the specified feature to be *cut in*; that is, the feature is projected first and the terrain is generated around it, as opposed to post-projection which generates the terrain first and then projects the feature on top of the terrain polygons.

Library substitution files that are pre-projected are called *footprint files*. Footprint files are OpenFlight files that have a bottom face with a footprint attribute set. When Creator detects the footprint attribute during TCT terrain generation, it projects the bottom face as part of the terrain. Footprint files do not have geocoordinates, but take their location from the point or areal features in the DFD data. For best results, the footprint polygon should be at least the same size as the feature's shape and should contain no more than four vertices. Polygons with many vertices use up too much of

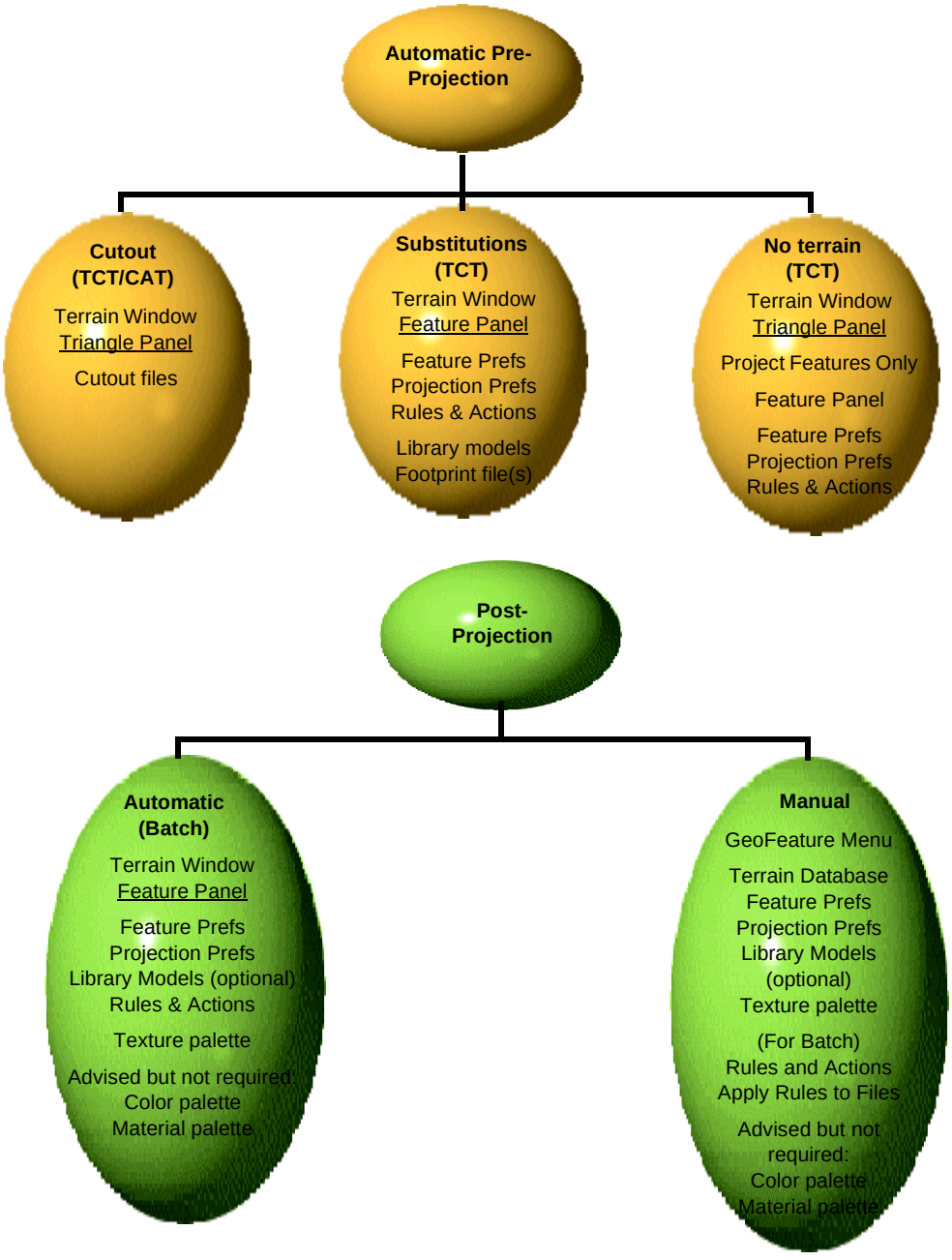
the polygon budget, limiting the number of terrain polygons that can be generated around them.

You can also pre-project features through the use of *cutout files*. Cutout files are OpenFlight files whose geocoordinates fall within the terrain area being processed. They are useful for adding important features, such as airports and military installations to your database and are frequently used to preserve a terrain's elevation. For example, an airfield located at an exact geospecific point must be placed at that particular location and elevation, and the terrain in that area will be leveled, if necessary, to accommodate the feature. You specify the cutout files you want to add to your database in the *Terrain* window's Triangle panel.



Pre-projected features have priority over post-projected features; linear or areal features will not be post-projected onto a pre-projected feature. A post-projected feature will be cut around a pre-projected feature if there is a conflict. This ensures that roads run through cities and rivers run though forests.

The following figure illustrates the types of feature projection, where each type is enabled, and the files that are used.

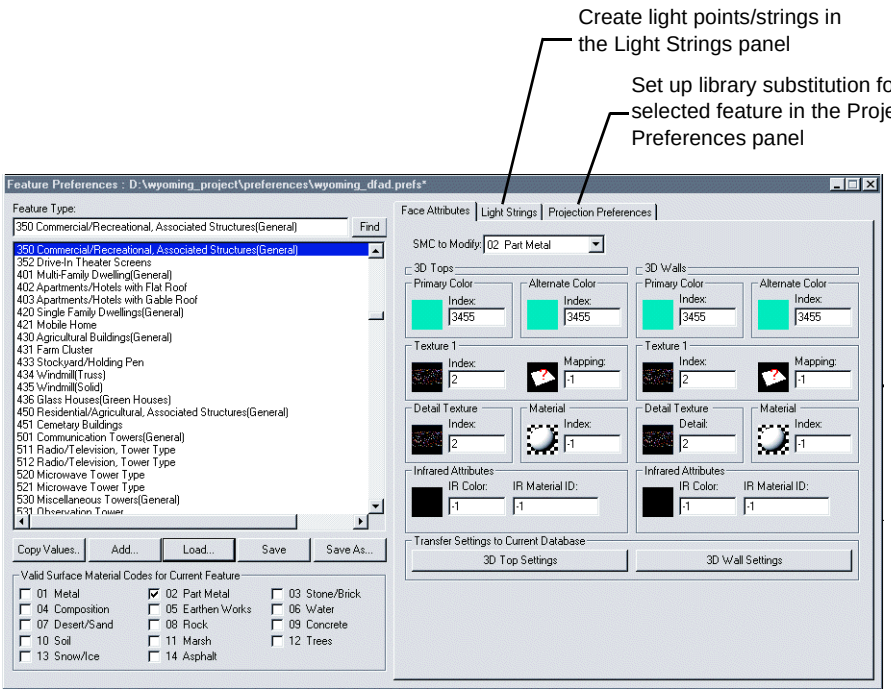


Setting Up Feature and Projection Files

Whether you are projecting features manually or automatically, Creator needs to know how you want the features to appear. To do this, you must set up preferences for the features being projected. If you want to substitute models for any projected features, these files must be available. In addition, if you are planning to project your features automatically, you must set up Rules and Actions.

Feature Preferences

Feature preferences define the color, material and texture properties associated with a particular feature. You open the *Feature Preferences* window by choosing **GeoFeature/ Feature Preferences**.



Each feature type (listed by feature ID and description) has its own set of preferences. You choose a feature description to define, and specify all the surface material codes

(SMCs) that apply to the feature type. Features get their texture, color, and material assignments based on the Feature Type / SMC pair assigned to them. For example, a wood building and a stone building, both of which have the same Feature Type, receive different textures when they are projected.

You can also create new feature types to define preferences for features that are not USGS or NIMA standard items; for example, the Golden Gate Bridge or the Statue of Liberty.

Once you have defined preferences for each of the features you will be projecting, you can write them to a file. By default this file is called dfad.prefs and is saved to the Creator directory. However, this file can be saved in any directory under any name (with a .prefs suffix) and manually loaded in the Project panel or GeoFeature Preferences window, or saved with a project file and loaded automatically when the project is loaded.

You can use Feature Preferences to create light points, light strings, or a grid of light points (applied to an areal of a city, for example). If you plan to substitute OpenFlight models for any of the features, Feature Preferences lets you specify the library substitution files to use. Feature Preferences also controls how features with height will be projected onto the terrain.

Projection Preferences

Global projection preferences affect how library substitution is performed and how texture is applied during manual and batch feature projection. You set projection preferences in the GeoFeature menu. Projection preferences set the path to the directory where your library substitution models are located. You can choose whether the library substitutions will be added as geometry in the database or inserted as external references.

Library Substitution/External References

Library substitution substitutes an OpenFlight model for a specified feature. Depending on the level of detail, you can apply high resolution models, low resolution models, or randomly apply models from one or both lists of library substitutions.

By setting projection preferences, you can choose to insert models into the database as external references rather than library substitutions. External references require less

memory than library substitutions, but the models may appear to float above, or intersect, the terrain because the bottom vertices of external references are not planted on the terrain faces. External references are generally used to manually place features into a database. They can be used successfully in dense or large areas highly populated with features.

Library substitutions can be inserted either manually or automatically during batch processing. To perform a manual substitution, once you have set up your feature and projection preferences, select the feature data to be projected and choose **GeoFeature/ Feature Projection/Hi Res Library Sub** or **Low Res Library Sub**. To perform an automatic substitution, you must create an action for the FID that you will be substituting.

Rules and Actions

When you project features automatically in the *Terrain* window's Feature panel, Creator needs to know how you want each feature to be projected. This is done by setting up Rules and Actions in the **GeoFeature** menu.

Actions determine which features to extract from the feature data and project, and whether library substitution is to be performed. *Rules* are a collection of actions. Only one rule can be applied to each level of detail. The order of the actions making up a rule determine the order that features will be projected. When adding actions to a rule, so you need to be careful that your feature layers are projected in the correct order (for example, projecting roads before buildings). Rules and Actions are stored in the file dfadbat1.prefs.

For each new feature created, you must open the dfad.prefs file to look up its assigned FID number for selection in the *Action Definition* window.

If you are using TCT, you can pre-project features by choosing the **Project to Terrain Data** option in the *Action Definition* window. This option projects a feature onto the elevation data and then generates the terrain around it, rather than placing the feature on top of the terrain (post-projection).

TCT has an option, **Project Linears Using Pathfinder**, that cuts drivable roads into the terrain. Pathfinder roads are drivable using the **Drive Roads** command in the **Road** menu, but you cannot use RoadPro tools to modify them. They are expensive in terms of polygon count and should only be used when your simulation requires a drivable road.

You also use Rules and Actions to project features onto existing terrain as a Batch operation. When you manually post-project features as a Batch operation, you do this through the **GeoFeature** menu's **Apply Rules to Files** option.

Palette Files

Regardless of the method used to project features, you will need to load at least a Texture palette containing all the textures your features will use. You can save the Texture palette with its loaded textures in a palette file, and add the palette to the terrain project file (*.prj).

You may also need to load custom Color and Material palette files, or specify them in the terrain project file, if your features require them.

Adding Light Points and Strings

Light points are individual vertices that represent distantly-viewed points of light. They can be applied as single, random points of light, strings of light points (light strings), irregularly shaped polygons (**Random Rectangle** and **Random Circle**), or grids of light points (light grids).

If you are adding light points to your simulation, you must consider them in your overall polygon budget. Depending on their attributes, your realtime system can process approximately four light points in the same amount of time it takes to process one polygon.

In order to project features with light points or strings, you must first create light string presets, which define your light point / string attributes, and then save the presets in a file (*.dat), which can be loaded in the *Light String Presets* window whenever you need it.

Projecting the Features

When you manually project features on terrain, you can select and project a few features in the feature layer, or you can project all the features. For example, you can perform an attribute search for all features with a FID of 420 Single Family Dwellings, and project only those. You can project features with height, project them flat, or substitute high or low resolution models using the **Feature Projection** option in the **GeoFeature** menu. Remember that when features are projected manually, they must be applied to each LOD separately.

For automatic feature projection during batch terrain generation, Rules and Actions dictate the features that will be projected in each LOD, the order in which they will be projected, and what they will look like. In the *Terrain* window Feature panel you load the feature data file (*.dfd) and specify the Rule you want to apply to each LOD you will be creating. Any cutout files being pre-projected must be loaded in the Triangle panel. Don't forget to set your terrain and batch preferences before generating the terrain.

To batch project features onto existing terrain, choose the **GeoFeature/Rules and Actions/Apply Rules to Files** command, load the feature data files, apply rules to the appropriate LODs and choose the target terrain files.

After you project features, you should test your database in your realtime system to check for errors and performance issues (see "Refining Your Database" on page 7-1 for more information).

6 Adding Special Features

You created a terrain database and populated it with features. But perhaps you also want to add other features to your database, such as modeled roads or special models. You may also want to add models that were created with other applications. This chapter describes the road-creation process including some tips for avoiding errors, and some of the tasks that must be performed when importing special models.

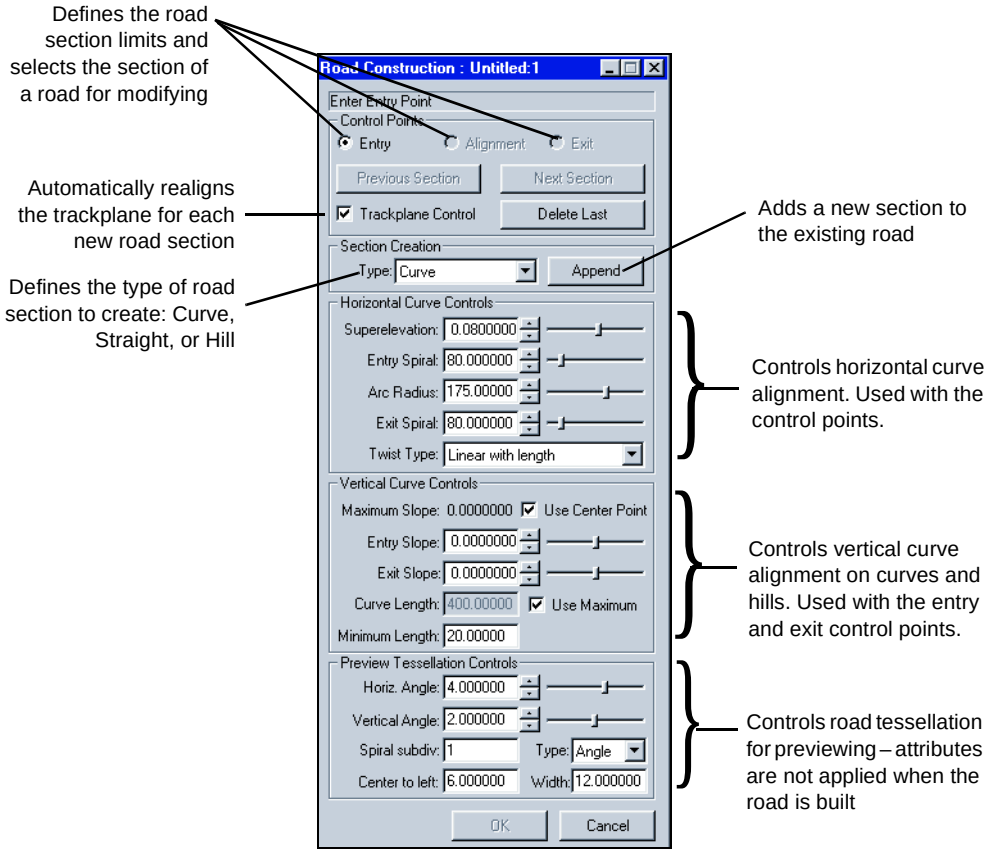
Building Roads

Roads can be modeled on existing terrain using the Creator RoadPro tools to define and model curves, hills, and arbitrary combinations of both after engineering design standards.

Creating roads with the RoadPro tools can be broken down into three basic operations: Road construction, road tessellation, and scenario data generation.

Road Construction

Road construction defines the type of road or road section to be created, its start and end points, and its curve or slope. You create roads using the Construction Tool (**Road/ Construction Tool**).



Using the Construction Tool, you define the type of road section to be created, set parameters to define curves, hills and slopes (superelevations), and then indicate the control points for the road section in the Graphic view. The road section appears green if it is a buildable road. If the road section appears red, it means the specified control points cannot be combined with the parameters in the dialog box to produce a valid road section.

Preview Tessellation Controls let you define an initial road width and centerline location so that you can view your road while it is being built. You can detect whether

definition settings must be modified for the road at a real world width. These definitions are only for viewing and are not applied to the road when it is constructed. You must use the Tessellation Tool to define the road's attributes.

All new or existing road alignments and design parameters can be interactively manipulated or modified, with dynamic update of the resulting 3D geometry.

Tips for Good Road Design

There are a few things to consider when designing a road. The speed of cars using the road and the climate of the area in which the road is being built will affect the arc radius, spiral transition and superelevation values.

- **Grid size**

  A frequent mistake made when first constructing roads is setting the grid size too small. When building small models, your grid size is usually set to about 10 or 20 units. When building roads on terrain using a small grid size, the control points are placed too close together to build mathematically feasible curves and transitions (roads that cannot be constructed appear red; buildable roads appear green). You will have greater success with your roads using a grid size of at least 100 units.

- **Superelevation**

  Nearly all curves on modern roadways employ some level of superelevation. The rate of superelevation is a function of both the radius of the curve and its design speed. Proper values for superelevation can be determined using formulas or curve tables in a highway design manual such as the *Policy on Geometric Design*, published by the American Association of State Highway and Transportation Officials (AASHTO).

  The maximum superelevation allowed on most public roads is 0.12. This is a maximum 12% bank for a highway in a warm climate. The recommended maximum superelevation for roads that are subject to snow or icy conditions is 8%, because at a low speed, cars slide off a road with a 12% bank.

- **Spiral Transitions**

  Spiral sections provide easement for horizontal curves, introducing curvature and banking. A large bank combined with a short spiral results in an abrupt transition.

Spiral lengths are typically the distance covered in four seconds of travel time at the design speed of the road. For example, at 65 m.p.h. (approximately 29 meters per second), 4 seconds imply 116 meters of spiral transition at each end of the curve. Spiral lengths are measured along the centerline of the road.

Balanced spirals (the same spiral length at each end of the curve) are desirable, but not required. In real life, the terrain, easements, and cost of property play an important role in determining the layout of curves (and of all other road sections).

If you have a dramatic spiral, you can use **Spiral Subdivision** in the Tessellation Tool's *Road LOD* dialog box to make a smoother polygonal model. When you cut more polygons, the road surface is less angular, but rendering additional polygons is more taxing on a realtime system. The default method for mapping the road surface is with curvature-based polygonization. The *Road LOD* dialog box also provides a **Spiral Type** option for custom control of the geometry and polygonization of spirals.

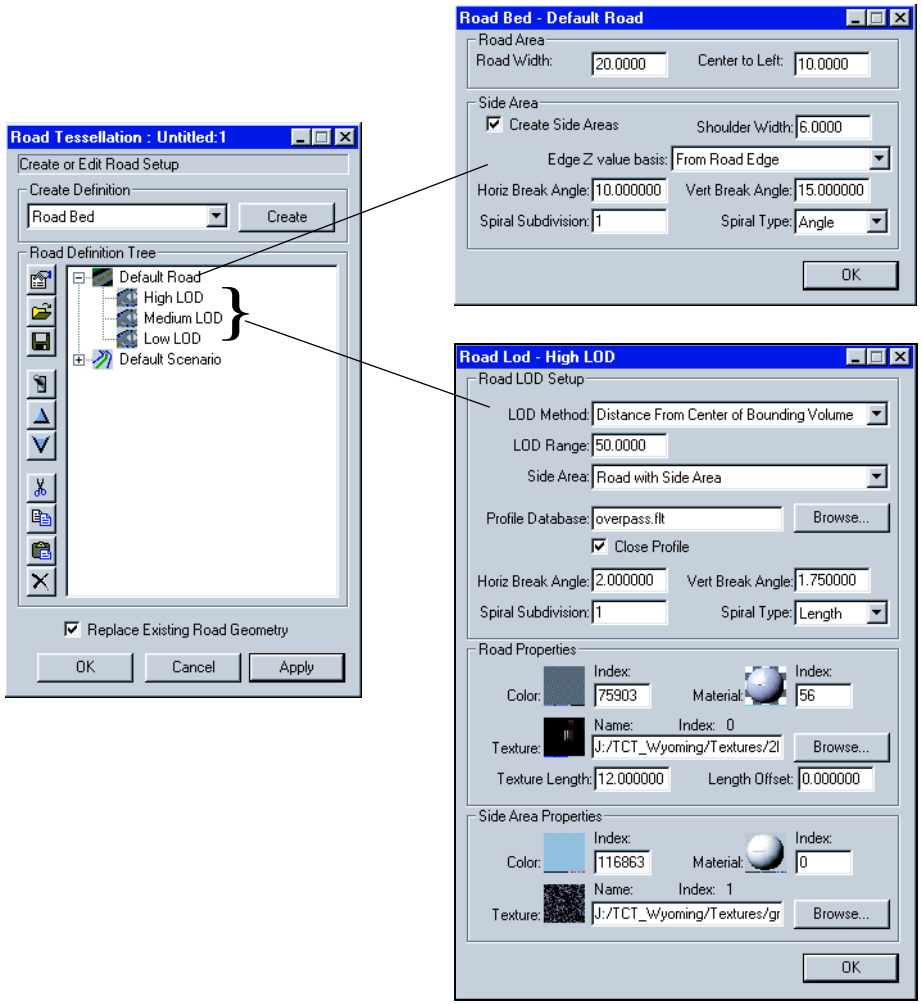
- **Arc Radius**

  The arc radius for a curve at a given speed is determined by a table of standards. For a given design speed and maximum superelevation, there are safety regulations that dictate what the minimum arc radius can be. For example, if a freeway has a maximum superelevation of 8% and a design speed of 120 km / hr, AASHTO recommends a minimum arc radius of 655 meters. For details on this and other design regulations, refer to an appropriate highway design policy or manual.

  A short arc radius results in a tighter curve. For a given design speed, a tighter curve requires a greater rate of superelevation than a more gradual curve. In gaming simulation, a curve with a short arc radius and high superelevation offers a more dramatic driving experience.

Road Tessellation

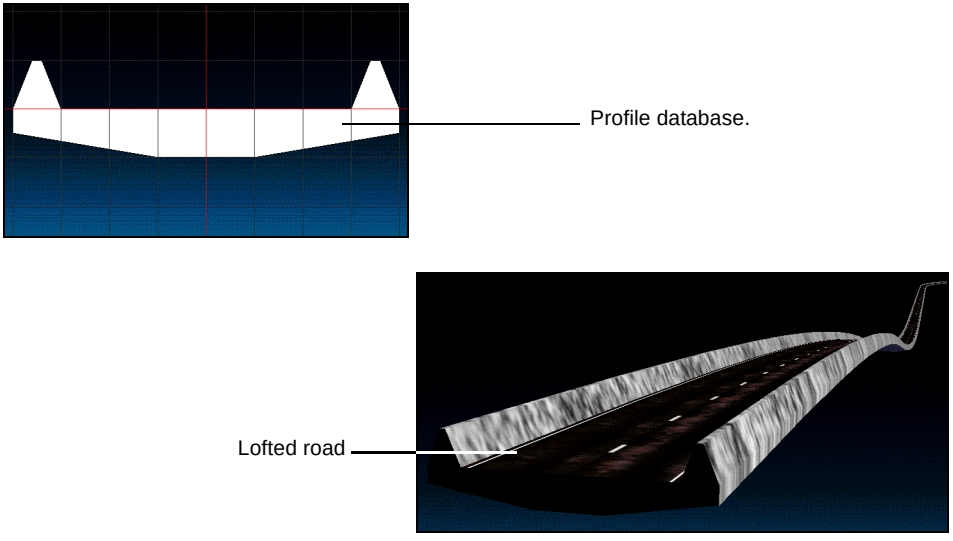
When you construct a road using the Construction Tool, you have only defined its length and any horizontal or vertical characteristics (curve and slope). The road appears in the Graphics view as a flat ribbon. You must use the Tessellation Tool to define the road's attributes.



These attributes include:

- Road width
- Road shoulder size and banking
- Color, texture, and material properties at different levels of detail
- Features such as light posts, traffic signs, or guard rails

You can use a *profile database* to loft the road. A lofted road has a three-dimensional shape that can be viewed from all sides – for example, a freeway overpass. A profile database is an OpenFlight file that contains a single polygon or small group of polygons that define the shape of the road. Be aware, however, that lofting a road produces more polygons.



When you define road attributes, you can apply them to an existing road, or you can immediately apply them to any new road section created by choosing the **Auto Tessellate** option.

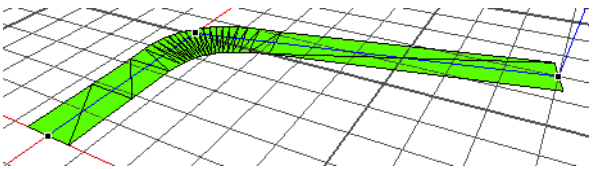
Road attribute definitions can be saved in a setup file (*.rds) which can be loaded at any time, edited, and applied to new or existing roads.

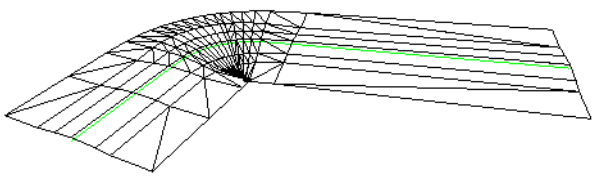
Tips for Tessellating Roads

Keep these tips in mind when defining road components and tessellating roads:

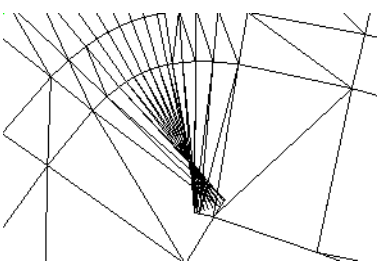
- Select the entire road when tessellating. Although you can select sections of a road for tessellating, textures applied in this manner may not line up correctly, and the distance between features may not be the same.
- Use the **Preview Tessellation** controls in the Construction Tool. Even though the road you build is mathematically logical (appears green in the Construction Tool), applying a wide width definition to a tightly-curved road section can result in the curve turning back on itself, as shown below. This can produce undesirable results when features are added, such as telephone poles that appear under the road. Use **Preview Tessellation** to view the road at the width you think you will be using. This helps you catch potential problems can be caught before the road is built.

Road appears green in the Construction Tool





But when tessellated, the road width causes the curve to turn back on itself



- When defining LODs, be sure the next higher LOD does not switch in before the lower LOD switches out. This can result in one or more LODs not switching in.

Driving the Road

The **Drive Road** command lets you drive along the road you have built. The driving path follows the eyepoint along the center of a lane (or at a specified offset), at a given speed, so you can check the characteristics of the road and the placement of roadside features. However, this routine does not include a simulation of vehicle dynamics.

Scenario Data

The **Road/Write Paths** command generates and writes path data in ASCII text data sets for use by vehicle dynamics and scenario control/traffic simulation subsystems in a driving simulator. These data sets contain raw analytic data.

> **Note:** Creator does not include subsystem routines for processing or querying the data during actual realtime simulation—these routines must be created by the developer.

The two data sets are *roadway centerline files* and *scenario lane files.*

The **Write Paths** source files can be modified or customized using the Creator API. The Creator API is included with the base Creator application, but it must be installed using a Custom Install.

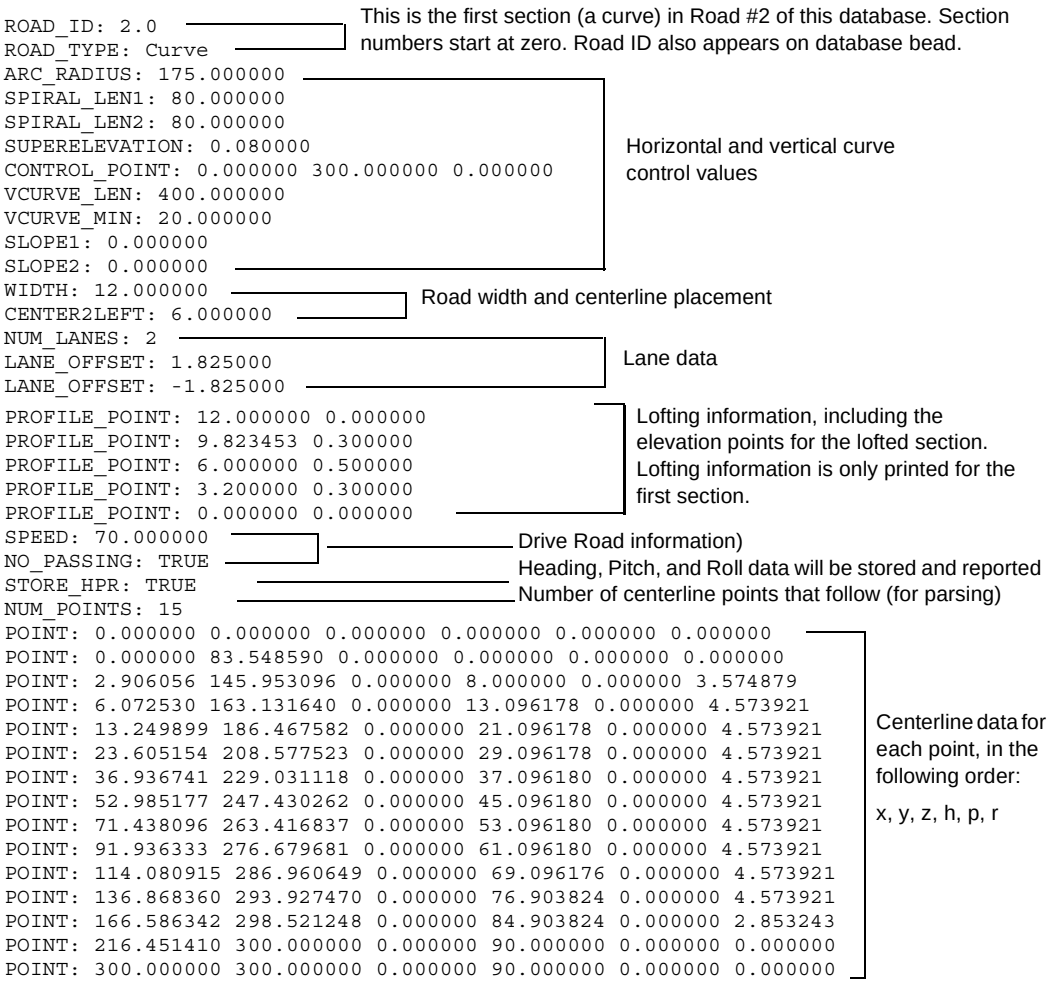
Roadway Centerline Files

Every road section has a centerline, regardless of whether Scenario data has been applied to it. Data written to a roadway centerline file consists of the road's construction data – its control points, horizontal and vertical curve controls, and data from the **Scenario** and **Lane** definitions.

A road scenario contains information that is written to the roadway centerline file, such as the profile database and its profile points.

Lanes provide additional scenario paths that can be "driven" by the traffic animation software. Scenario paths are sequences of unclosed wireframe faces. Until you create a lane, the only drivable path in a road is the centerline.

The following figure is an example of a roadway centerline file.



```
ROAD_ID: 2.1
ROAD_TYPE: Hill
VCURVE_LEN: 241.761226
VCURVE_MIN: 20.000000
SLOPE1: 0.000000
SLOPE2: 0.248179
WIDTH: 12.000000
CENTER2LEFT: 6.000000
NUM_LANES: 2
LANE_OFFSET: 1.825000
LANE_OFFSET: -1.825000
PROFILE_NAME: /usr/people/mike/road/crown.flt
SPEED: 70.00000
NO_PASSING: FALSE
STORE_HPR: TRUE
NUM_POINTS: 6
POINT: 300.000000 300.000000 0.000000 90.000000 0.000000 0.000000
POINT: 358.238774 300.000000 0.000000 90.000000 0.000000 0.000000
POINT: 426.357384 300.000000 2.381659 90.000000 4.000000 0.000000
POINT: 495.145434 300.000000 9.620488 90.000000 8.000000 0.000000
POINT: 565.298943 300.000000 22.005999 90.000000 12.000000 0.000000
POINT: 600.000000 300.000000 30.000000 90.000000 13.937990 0.000000
```

Data for the second section (a hill) in Road #2 of this database

```
ROAD_ID: 2.2
ROAD_TYPE: Straight
SLOPE1: 0.248179
SLOPE2: 0.248179
WIDTH: 12.000000
CENTER2LEFT: 6.000000
NUM_LANES: 2
LANE_OFFSET: 1.825000
LANE_OFFSET: -1.825000
PROFILE_NAME: /usr/people/mike/road/crown.flt
SPEED: 70.000000
NO_PASSING: FALSE
STORE_HPR: TRUE
NUM_POINTS: 2
POINT: 600.000000 300.000000 30.000000 90.000000 13.937990 0.000000
POINT: 697.055698 300.000000 54.087162 90.000000 13.937990 0.000000
```

Data for the third section (straight) in Road #2 of this database

Scenario Lane Files

Scenario lane data contains a list of points along the lane and a few lane attributes. It specifies the heading, pitch and roll or surface normal vector at each point along the lane's centerline. Scenario lanes do not contain road design parameters, cross-section data, or other surface information for vehicle dynamics processing.

Scenario lane files are written out as formatted ASCII text files. Each lane is written to a separate file. The following is an example of a scenario lane file.

```
SPEED: 70.000000
NO_PASSING: TRUE
STORE_HPR: FALSE
NUM_POINTS: 15
POINT: 1.825000 0.000000 0.000000 0.000000 0.000000 1.000000
POINT: 1.825000 83.548590 0.000000 0.000000 -0.005916 0.999982
POINT: 4.709779 145.699599 -0.113794 0.060563 -0.017077 0.998018
POINT: 7.844403 162.719437 -0.145535 0.076406 -0.023500 0.996800
POINT: 14.947160 185.812793 -0.145535 0.072392 -0.033905 0.996800
POINT: 25.194769 207.692893 -0.145535 0.066969 -0.043650 0.996800
POINT: 38.387769 227.933866 -0.145535 0.060242 -0.052546 0.996800
POINT: 54.269375 246.141745 -0.145535 0.052343 -0.060418 0.996800
POINT: 72.530470 261.962133 -0.145535 0.043425 -0.067115 0.996800
POINT: 92.815621 275.087105 -0.145535 0.033662 -0.072506 0.996800
POINT: 114.730002 285.261199 -0.145535 0.023369 -0.076437 0.996801
POINT: 137.280563 292.155597 -0.145535 0.017922 -0.077705 0.996815
POINT: 166.748252 296.705716 -0.090845 0.007396 -0.049316 0.998756
POINT: 216.451410 298.175000 0.000000 0.000000 0.000000 1.000000
POINT: 300.000000 298.175000 0.000000 0.000000 0.000000 1.000000

SPEED: 70.000000
NO_PASSING: FALSE
STORE_HPR: FALSE
NUM_POINTS: 6
POINT: 300.000000 298.175000 0.000000 0.000000 0.000000 1.000000
POINT: 358.238774 298.175000 0.000000 -0.034942 0.000000 0.999389
POINT: 426.357384 298.175000 2.381659 -0.104656 0.000000 0.994509
POINT: 495.145434 298.175000 9.620488 -0.173860 0.000000 0.984770
POINT: 565.298943 298.175000 22.005999 -0.224488 0.000000 0.974477
POINT: 600.000000 298.175000 30.000000 -0.224488 0.000000 0.974477

SPEED: 70.000000
NO_PASSING: FALSE
STORE_HPR: FALSE
NUM_POINTS: 2
POINT: 600.000000 298.175000 30.000000 -0.240872 0.000000 0.970557
POINT: 697.055698 298.175000 54.087162 -0.240872 0.000000 0.970557
```

Speed limit (a path attribute)

Drive Road information

Up-vector will be stored and reported

Number of lane center points that follow (for parsing)

Lane data for each point, in the following order:

x, y, z, i, j, k

Data for the same lane, but in the next section of the road

Data for the same lane, but in the next section of the road

Adding Special Models

Depending upon the type of simulation you are creating, you may add specialized features by hand. These features can include one-of-a kind models, such as the Statue of Liberty, or features that must be cut in by hand if the terrain was not generated with TCT.

Importing Models From Other Applications

Adding models created in other applications may require so much manual editing that it is sometimes less work to re-create them within Creator. The model data must be converted from its native file format to OpenFlight. Because OpenFlight files have a hierarchical structure and most modeling applications do not, the data conversion is not one-for-one. It may take a great deal of manual rework to create an acceptable model that you can incorporate into your simulation. The work may include:

- Creating the proper hierarchy nodes and placing child nodes under them
- Rearranging the structure so it can be culled efficiently
- Manually reducing the number of polygons to stay within the polygon budget
- Deleting hidden polygons
- Reapplying color, texture and material properties that were not converted with the model or were not properly mapped
- Eliminating concave polygons and T vertices.

7 Refining Your Database

Once you create your terrain and add your features, you should check the performance of the database simulation on your realtime system. You may need to reduce the number of feature polygons or modify the database hierarchy to improve performance.

Balancing Culling and Drawing

The amount of time your realtime system spends culling the data to be viewed must be balanced against the amount of time spent rendering the result to the screen so that both are done as efficiently as possible.

When your realtime application is determining what should be culled and what should be drawn to the screen, an entire group is either culled or drawn. Small groups result in longer culling time (more groups must be evaluated to decide what must be drawn or discarded); large groups result in longer drawing time (if even a small area of the group is within the field of view, the entire group must be drawn).

- If the system spends too much time culling, increase the culling (group) size.
- If the system spends too much time drawing, decrease the culling (group) size.

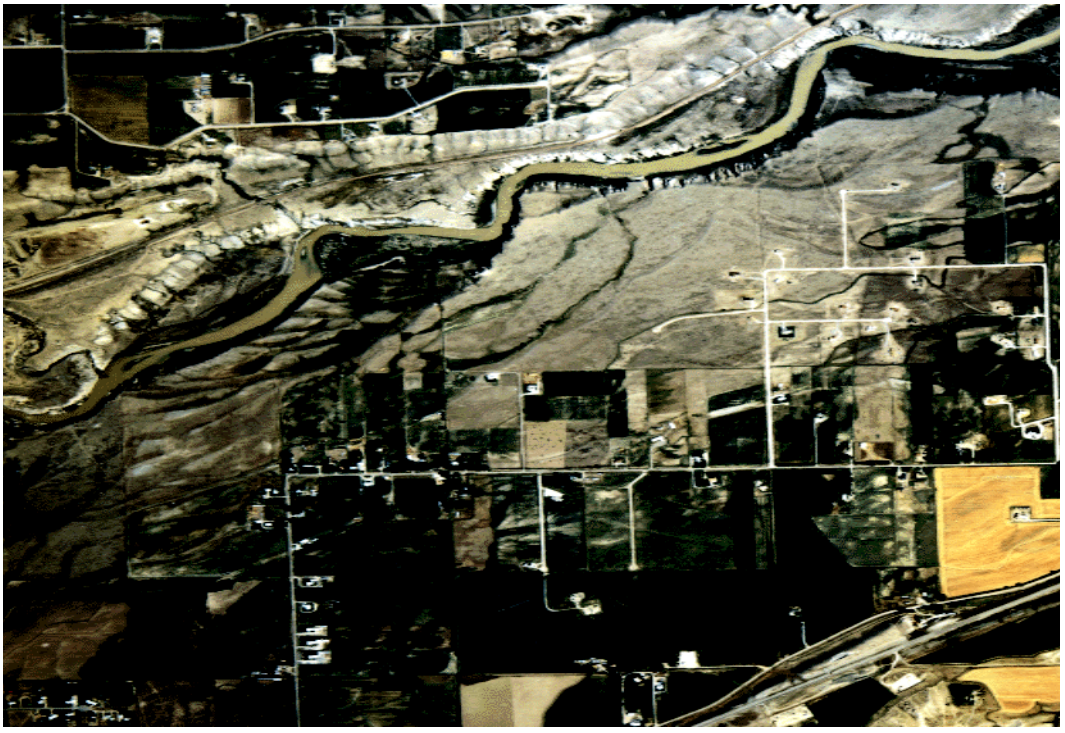
When LODs switch in the runtime, the larger the group size, the larger the terrain area that changes. When the group size is too large, however, features on the terrain may appear suddenly as the new LOD switches in. This is called “LOD popping.”

Most realtime systems give statistics for culling, drawing, application processing, and collision detection. Once you create the terrain and models, you should evaluate your simulation’s performance on your realtime system and make adjustments to the terrain’s group size based on the results displayed by the statistics. For example, you might adjust the group size, which is the basic LOD and culling element.

Reducing Polygon Count

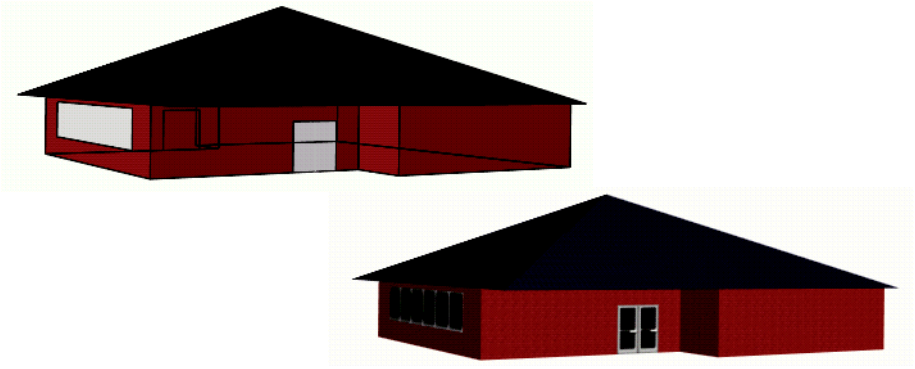
What are some ways to reduce polygon count without sacrificing the quality of your simulation?

- Use highly-detailed textures to substitute for polygonal features at lower levels of detail and save the polygon budget for higher levels of detail. The terrain polygon count for a lower LOD should be at least 75% of the next highest LOD in order to prevent gaps in the terrain.

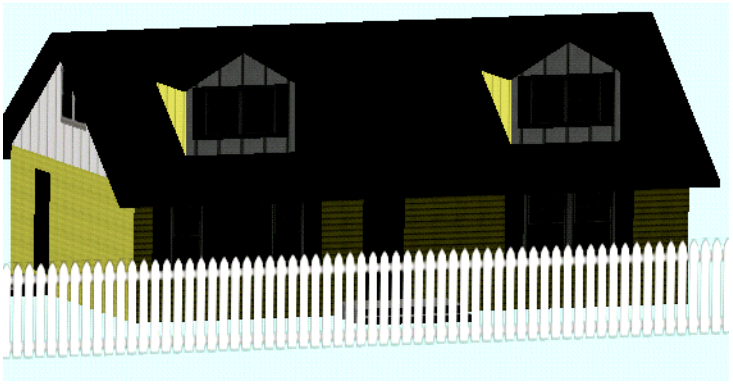


- Where possible, combine polygons. In the following example, rather than creating a separate polygon for each window and door on the building, a large polygon was

created for all the windows and another for the double doors, and a multi-window and double door texture was applied.



- Use textures to substitute for polygon detail. In the example below, rather than model all the actual geometry of a fence, an RGBA texture of a picket fence was applied to a rectangular polygon. The alpha component between the pickets is transparent, allowing you to view the house through the fence.



- Remove polygons that will not be visible from the simulation participants' viewpoint. For example, remove bottom (base) polygons such as the undersides of the dormers and the roof in the picture above.

8 Finishing Your Simulation

The entire sequence of tasks involved in creating and optimizing a terrain simulation is an iterative process. Your test terrain must be tested, adjusted, regenerated, and retested, if necessary, before moving on to your large area terrain. You add any textures needed, add your low resolution features, then your high resolution features such as library substitutions and cutout files, features that must be cut in by hand (if you are not using TCT), roads, and other, special models. During all these stages, you must check your terrain database for accuracy and performance, both in Creator and in your realtime system, make modifications, and check again.

In some cases, making changes at one point in the process can cause problems in other areas, requiring you to take a few steps back.

Finally, you must test the entire simulation in your realtime system with the special effects your image generator provides, such as motion and daylight effects. Does the simulation meet the criteria you set out during the planning stages? Does it perform at an acceptable frame rate? When your simulation passes these final tests, you are ready to publish it.

Index

A

Algorithm, advantages and disadvantages 3-14
Applications processing, effect on terrain 2-1
Applying textures 4-1
Area block size, setting 3-7
ASCII text reports, for driving simulation 6-8
Attribute searching 5-4
Auto Tessellate 6-6

B

Batch feature projection 5-19
Batch GeoPut 4-3
Batch processing
    area block size 3-7

C

CAT terrain conversion
    advantages and disadvantages 3-16
    description of 3-18
Checking and correcting feature data 5-3
Clip textures 4-4
Collision detection 2-1
Color palette file 2-5
Construction Tool 6-2
Contour properties
    combining with face attri butes 3-9
    Elevation scale 3-10
Converting feature files to OpenFlight 6-12
Correlated data, scenario lane files 6-10
Culling
    effect on terrain 2-1

D

Data conversion
    terrain data 3-1
Data conversion, feature data 5-2
Database origin
    relative to elevation data lat/long 3-5
Database planning
    data organization 2-3
    estimating polygon budget 2-2
    hardware and software requirements 2-1
    polygon budget allocation 2-2
    project organization 2-4
    reducing polygon count 7-1
    simulation type 2-1
ded file, converting source data 3-2
ded.prefs file 2-4
ded1.prefs file 2-5
Delaunay terrain conversion
    advantages and disadvantages 3-15
    description of 3-17
Detail textures 4-6
dfad.prefs file 2-5
dfadbat1.prefs file 2-5
DFD construction layer - see Feature layer
Driving a road 6-8

E

Elevation data
    importing into Terrain window 3-3
Elevation scale
    changing size and color 3-10
Ellipsoid projection 3-13
Error checking 3-18

F

Factors in terrain generation 3-1
Feature data
    converting raw source data 5-2
    editing 5-10
    importing 5-3
Feature layer 5-6
Feature preferences
    defining features for projection 5-15
    dfad.prefs file 2-5
    light strings 5-16
Feature projection
    batch 5-19
    pre-projecting features 5-17
    setting up Rules and Actions 5-17
Features
    automatic feature projection 5-19
    automatic post projection 5-12
    checking feature data 5-3
    choosing features to import 5-4
    creating new features 5-10
    cutout files 5-12
    eliminating features 5-2
    external references 5-16
    feature preferences 5-15
    feature reduction methods 5-9
    features created with other applications 6-12
    footprint files 5-12
    global projection preferences 5-16
    library substitution files 5-12, 5-16

light points and strings 5-18
light strings 5-16
manual feature projection 5-19
manual post projection 5-11
manually cutting in features 6-11
modifying feature data 5-3
order of projection 5-2
palette files 5-18
performing attribute searches 5-4
pre-projection 5-12, 5-17
projection methods 5-11
sequence of tasks for applying 5-1
setting up Rules and Actions 5-17
special features 6-11
types 5-1
Flat Earth terrain projection 3-11

G

Geocentric terrain projection 3-12
GeoFeature menu
Apply Rules to Files 5-18
Feature Construction 5-11
Feature Construction/Reduce Feature Vertices 5-9
Feature Preferences 5-15
Feature Projection 5-19
Feature Projection preferences 5-16
Import 5-3
NewFeature Layer 5-10
Reduce Feature Vertices 5-8, 5-9
Rules and Actions 5-17
GeoPut Texture tool 4-3
Geospecific texture
applying 4-3
Batch GeoPut menu option 4-3
clip textures 4-4
description of 4-2
mipmaps 4-5
overview 4-2
Grid, setting for building roads 6-3

H

High resolution insets 4-5

I

Indirect Texture, required files 4-8
Irregular Mesh 3-17

L

Lambert Conic Conformal terrain projection 3-13
Level of detail, see "LOD"
Library substitution, projection preferences 5-16
Light points
adding with features 5-18
consideration in polygon budget 5-18
Light Source palette file 2-5
Light String Preset file 2-6
LOD
factors affecting number of 3-7
setting switching distances 3-8
simulation type considerations 2-1
Lofting roads 6-6

M

Map panel, settings 3-10
Map projection
Flat Earth 3-11
Geocentric 3-12
Lambert Conic Conformal 3-13
Trapezoidal 3-13
UTM 3-11
Material palette file 2-5
Mipmaps 4-5

P

Palette files, in feature projection 5-18
Path files 6-8
Polygon budget
allocation 2-2
considerations 2-2
reducing polygon count 7-1
Polymesh
advantages and disadvantages 3-14
Irregular Mesh terrain processing 3-17
terrain processing 3-16
Post projection
automatic 5-12
manual 5-11
Previewing roads 6-2
Profile database 6-6
Project Feature Only, without terrain 5-3
Project file
external files 2-4
using to organize data 2-4
Project, organizing 2-3
Projection preferences 5-16

R

Realtime considerations
- applications processing 2-1
- collision detection 2-1
- culling 2-1
- rendering 2-1

Reducing the number of features 5-9
Reducing vertices
- after features are imported 5-9

Road menu
- Drive Road 6-8
- Write Paths 6-8

Roads
- applying attributes 6-6
- Construction Tool 6-2
- design tips 6-3
- generating path files 6-8
- grid size 6-3
- lofted roads 6-6
- Preview Tessellation controls 6-2
- roadway centerline files 6-8
- scenario definition 6-8
- tessellation tips 6-7
- Tessellation Tool 6-5

Roadway centerline files 6-8
Rules and Actions
- dfadbat1.prefs file 2-5
- setting up 5-17

S

Scenario data
- roadway centerline files 6-8
- scenario lane files 6-10

Scenario lane files 6-10
Selecting a terrain area for processing 3-5
Simulation requirements 2-1

T

TCT terrain conversion
- advantages and disadvantages 3-16
- description of 3-18
- pre-projecting features 5-12, 5-17
- projecting features without terrain 5-3

Terrain menu
- Batch GeoPut 4-3

Terrain processing
- advantages of Batch processing 3-5
- applying geospecific texture 4-3
- choosing a terrain algorithm 3-14
- choosing type of processing 3-5
- contour properties 3-9
- database origin 3-5
- ellipsoid projection 3-13
- error checking 3-18
- geospecific texture 4-2
- importance of testing 3-1
- importing elevation data 3-3
- indirect texture 4-7
- LOD switching distances 3-8
- Map panel settings 3-10
- methods for area selection 3-5
- number of LODs 3-7
- setting area block size 3-7

Terrain Project files
- Color palette file 2-5
- ded.prefs 2-4
- ded1.prefs 2-5
- dfad.prefs 2-5
- dfadbat1.prefs 2-5
- Light Source palette file 2-5
- Light String Preset file 2-6
- Material palette file 2-5
- Texture palette file 2-5

Terrain window
- contour properties 3-9
- preferences file 2-4
- Project panel 2-3

Terrain, design considerations 3-1
Tessellation Tool 6-5
Texture
- as substitution for polygons 7-1
- Batch GeoPut 4-3
- blending with face attributes 4-2
- clip textures 4-4
- detail textures 4-6
- geospecific texture 4-2
- high resolution insets 4-5
- indirect texture 4-7
- methods of application 4-1
- mipmaps 4-5

Texture palette file 2-5
Trapezoidal terrain projection 3-13

U

Universal Transverse Mercator, see "UTM terrain projection"

UTM terrain projection 3-11

W

Write Paths, Road menu 6-8