OpenFlight® Scene Description Database Specification

Version 15.7.0
April, 2000

550 S. Winchester Blvd., Suite 500, San Jose, CA 95128 Phone (408) 261-4100 Fax (408) 261-4103

OpenFlight Scene Description Database Specification,v15.7.0 (April, 2000)

Contents

ADOPTER REGISTRATION AGREEMENT FOR OPENFLIGHT® SCENE DESCRIPTION DATABASE SPECIFICATION

The purpose of this agreement is to enable third parties who agree to adopt the MultiGen-Paradigm® Inc. OpenFlight Scene Description Database Specification, on a non-exclusive basis, to receive ongoing access to technical updates to OpenFlight. Registered "Adopters" will also receive marketing support onMultiGen-Paradigm's World Wide Web (WWW) site once your product is completed. To become a registered adopter, complete and sign this registration form and return to MultiGen-Paradigm Inc., 550 S. Winchester Boulevard, Suite 500, San Jose, CA 95128, Attn: OpenFlight Registration, or fax the completed form to (408) 261-4103.

____, whose place of business is

____ (hereinafter "User")

desires to obtain a copy of the OpenFlight Scene Description Database Specification (hereinafter "OpenFlight.") OpenFlight contains information belonging to MultiGen-Paradigm, Inc., a California corporation located in San Jose, California. The parties wish to define their rights with respect to OpenFlight. Therefore, it is agreed as follows:

OpenFlight is the property of MultiGen-Paradigm, Inc. and is protected under the copyright and trademark laws of the United States of America. MultiGen-Paradigm, Inc. hereby grants to User a non-exclusive, non-transferable limited right to use OpenFlight as follows:

a. For reading OpenFlight into a computer program or database for in-house use, or as a feature of a commercial product.

b. For writing data from a computer program or database into OpenFlight for in-house use or as a feature of a commercial product.

Any attempt to sub-license, assign, or transfer all or any part of the OpenFlight Specification is prohibited without the prior written consent of MultiGen-Paradigm, Inc.

OpenFlight and MultiGen are registered trademarks of MultiGen-Paradigm, Inc. User agrees to indicate MultiGen Inc.'s ownership of its trademarks in any of User's published references to such trademarks. User shall not at any time use any name or trademark which is confusingly similar to a MultiGen-Paradigm, Inc. trademark.

OPENFLIGHT IS OFFERED FOR USE BY USER "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. ALL SUCH WARRANTIES ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL MULTIGEN-PARADIGM, INC. BE RESPONSIBLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OF OPENFLIGHT.

Adopter (User):

Signature ____ Date ______________________________________

Printed Name ____________________________________ Title _____________________________________

1 OpenFlight® Scene Description

This document describes the concepts of the OpenFlight Scene Description Database Specification created and maintained by MultiGen Incorporated. OpenFlight databases can be created and edited using MultiGen software tools and applications.

Document Conventions

Lines and paragraphs that contain a discussion of material new to the current release of the software are marked with a revision bar, such as the one to the left.

Concepts Supported in OpenFlight

The OpenFlight database format supports both simple and relatively sophisticated real-time software applications. The full implementation of OpenFlight supports variable levels of detail, degrees of freedom, sound, instancing (both within a file and to external files), replication, animation sequences, bounding volumes for real-time culling, scene lighting features, light points and light point strings, transparency, texture mapping, material properties, and other features.

A simple application that interprets an OpenFlight database can implement a subset of the database specification and use databases that contain that subset. Such an application scans for the color palette, faces, and vertices, and ignores groups, objects, and other more sophisticated features.

Database Hierarchy

The OpenFlight database hierarchy organizes the visual database into logical groupings and facilitates real-time functions such as field-of-view culling, level-of-detail switching, and instancing. Each OpenFlight database is organized in a tree structure.

The database tree structure consists of nodes (historically called beads). Almost every node can have child nodes and sibling nodes. Each node type has data attributes specific to its function in the database. The principal node types are as follows:

Header: There is one header node per database file. It is always the first node in the file and represents the top of the database hierarchy and tree structure.

Group: A group node distinguishes a logical subset of the database. Group nodes can be transformed (translated, rotated, scaled, etc.). The transformation applies to itself and to all its children. Groups can have child nodes and sibling nodes of any type, except a header node.

Object: An object node contains a logical collection of geometry. It is effectively a low-level group node that offers some attributes distinct from the group node.

Face: A face node represents geometry. Its children are limited to a set of vertices that describe a polygon, line, or point. For a polygon, the front side of the face is viewed from an in-order traversal of the vertices. Face attributes include color, texture, material, and transparency.

Mesh: A mesh node defines geometric primitives that share attributes and vertices. See "Mesh Node Records" on page 18 for more information.

Light point: A light point node represents a collection of light point vertices or a replicated string of a single light point vertex. A light point is visible as one or more self-illuminated small points that do not illuminate surrounding objects.

Subface: A subface node is a face node that is coplanar to, and drawn on top of, its superface. Subfaces can themselves be superfaces. This construct resolves the display of coplanar faces. A subface is introduced, after a face node, by a push subface control record and concluded by a pop subface control record.

Light source: A light source node serves as the location and orientation of a light source. The light source position and direction are transformed by the transformations above it in the tree (if any).

Sound: A sound node serves as the location for a sound emitter. The emitter position is the sound offset transformed by the transformations above it in the tree (if any).

Text: A text node draws text in a string with a specified font, without injecting the actual geometry into the database as face nodes. This is a leaf node and therefore cannot have any children.

Vertex: A vertex node represents a list of one or more double precision 3D coordinates. For each coordinate, the node references a vertex attribute record that is stored in the vertex palette record. Vertex attributes include x, y, z and optionally include color, normal and texture mapping information. Vertex nodes are the children of face nodes and light point nodes.

Morph vertex: A morph vertex node is a second vertex node. The vertex and morph vertex represent the two endpoints of a path between which the actual vertex may be interpolated. One endpoint represents the minimum (non morphed) weighting and the other represents the maximum (fully morphed) weighting. Each endpoint (or weight) is a reference into the vertex palette record. All vertex attributes may be morphed. Morph vertex nodes are the children of face nodes.

Clip region: A clip node defines a set of clipping planes. Any geometry, of the clip node's children, that falls outside the specified clipping planes is not displayed.

Degree of freedom: A degree-of-freedom (DOF) node serves as a local coordinate system with a predefined set of internal transformations. It specifies the articulation of parts in the database and set limits on the motion of those parts.

Level of detail: A level-of-detail (LOD) node serves as a switch to turn the display of everything below it on or off based on its range from the viewer, according to its switch-in, switch-out distance and center location.

Switch: A switch node is a more general case of an LOD node. It allows the selection of zero or more children by invoking a selector mask. Any combination of children can be selected per mask and the number of definable masks is unlimited.

External reference: An external reference node serves to reference a node in another database file, or an entire database file. The referenced (child) node or database is considered an external part of the referencing (parent) database.

Instancing

Instancing is the ability to define all or part of a database once, then reference it one or more times while applying various transformations. OpenFlight supports internal and external instancing with operations such as Rotate, Translate, Scale, and Put.

An internal instance is a subtree of the database that has been declared an instance definition. An instance definition represents the root of a stand-alone subtree within the database. It is introduced by an instance definition record that contains a unique instance ID. An instance definition is invoked by an instance reference record in a subsequent part of the database tree.

An external instance refers to an entire database file. It is introduced by an external reference node. An external reference node contains the name of the (child) database file to attach to that point in the referencing (parent) database tree. It also includes attributes that determine whether the child uses its own color, material, and texture palettes, or those of its parent.

Instance definitions can themselves contain instance definitions and references. Internal instances cannot reference themselves. External instances should not reference themselves directly or indirectly. The result of such use is undefined.

Replication

Replication instances a subtree of the database several times, applying a transformation each time. For example, a string of trees can be represented by a single group node that is instantiated and translated to a new position several times.

Replication is legal for group, face, and light point nodes. Therefore a replication record is an ancillary record of a group, face, or light point node. In conjunction with a replication record there will be one or more ancillary transformation records.

Bounding Volumes

Bounding volumes can be used by the application to determine if a particular subtree of the database is in view. A bounding volume can be a box, a sphere, or a cylinder. Each group node can have only one bounding volume. The volume normally encompasses the full geometric extent of the group node’s children, including any instances and replications. A bounding volume record is an ancillary record of a group node.

1 OpenFlight File Format

The tree structure of an OpenFlight database is stored on disk as a file. The file consists of a linear stream of binary records. All OpenFlight records begin with a 4 byte sequence. The first two bytes identify the record type (opcode) and the second two bytes specify the length of the record. Given this regular record structure, the length of all OpenFlight records is limited to the largest value that can be encoded with 2 bytes or 16 bits (65535).

- The length value includes any implicit padding inserted between fields, or appended to the end, of each record (in accordance with the ANSI C structures; see appendix F.3.9-3.5.2.1).
- All records are a multiple of 4 bytes in size.
- All records are only guaranteed to be 4 byte aligned.

For fixed-size records, this maximum size is sufficient. For variable-size records, this limitation is being addressed with the continuation record, which is new in this version.

OpenFlight Record Types

There are four major categories of records: control records, node primary records, ancillary records, and continuation records.

Control records mark the hierarchy of the tree. A push control record (a record containing the push opcode) indicates an increase in the depth of the tree. A push control record drops you down one level in the tree. A pop control record (a record containing a pop opcode) returns you to the previous level of hierarchy. All records between a push and a pop represent sibling nodes at the same level of hierarchy. Other control records include: instance definition, instance reference, push subface, pop subface, push attribute, and pop attribute.

Each node is represented on disk by one primary record and zero or more ancillary records. The primary record identifies a node type and includes most of the node attribute data. Additional node attributes, such as comments, long ID, and transformations, are stored in subsequent ancillary records. Ancillary records follow the primary record, but precede any control records. Child nodes are introduced by a push control record and are concluded by a pop control record.

Palette records are ancillary records of the header node. Palette records generally follow the header node's primary record, with the exception of behavior (linkage) palette records. Behavior palette records, if present, are the last (non-control) records in the file.

Continuation records are used to "continue" a record in the OpenFlight Scene Description file stream, when the original record is larger than 65 bytes. It appear in the stream immediately following the record that it "continues." The data contained in the continuation record is defined by the original record and is assumed to be directly appended onto the content of the original record. Multiple continuation records may follow a record, in which case all continuation records would be appended (in sequence) to the original record

Many records include an eight character ASCII ID consisting of the first seven characters of the node name plus a terminating <nil> character. If the node ID is longer than seven characters, an ancillary long ID record containing the complete ID follows the node primary record.

For example, a record with an object opcode is followed by a push control record. Next comes a record with a face opcode, also followed by a push control record. After that comes the vertex list record(s) that describe the vertices of the face, and then a pop control record. This, in turn, may be followed by another face record for the next face in the same object, or by a pop record to return to object level.

The fields within each OpenFlight record are stored in big-endian byte order. OpenFlight database files have the extension ".flt" by convention.

Control Records

Control records indicate a change in the level of the database hierarchy. The three basic types of control records are: level changes, instance definition, and instance reference. Level changes are indicated by push and pop control records. Instance definitions and references are indicated by instance definition and instance reference control records.

Hierarchy Level Change Records

A database contains three distinct types of hierarchy: generic, subface, and attribute. Hierarchy may be skipped by scanning past the push control record for the corresponding pop control record.

Generic	A push level control record introduces a generic subtree of the database hierarchy. A pop level control record concludes that subtree.
Subface	A push subface control record introduces a subtree of coplanar faces. A pop subface control record concludes that subtree.
Extension	A push extension control record introduces a subtree of user defined records. A pop extension control records concludes that subtree.

Attribute	A push attribute control record introduces a subtree of records reserved for internal use by MultiGen Inc. A pop attribute control record concludes that subtree.

Push Level Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Level Opcode 10
Unsigned Int	2	Length of the record = 4

Pop Level Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Level Opcode 11
Unsigned Int	2	Length of the record = 4

Push Subface Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Subface Opcode 19
Unsigned Int	2	Length of the record = 4

Pop Subface Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Subface Opcode 20
Unsigned Int	2	Length of the record = 4

Push Extension Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Extension Opcode 21
Unsigned Int	2	Length of the record = 24
Char	18	Reserved
Unsigned Int	2	Vertex reference index; -1 if not vertex extension

Pop Extension Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Extension Opcode 22
Unsigned Int	2	Length of the record = 24
Char	18	Reserved
Unsigned Int	2	Vertex reference index; -1 if not vertex extension

Push Attribute Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Subface Opcode 122
Unsigned Int	2	Length of the record = 8
Int	4	Vertex Number in Vertex List (if following a vertex list else -1)

Pop Attribute Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Subface Opcode 123
Unsigned Int	2	Length of the record = 4

Hierarchy Instancing Records

An instance definition record introduces a stand-alone subtree of the database. The subtree is referenced one or more times from different branches in the database by instance reference records. At the point of reference, the subtree is copied (or possibly shared) as a child of the current parent node.

Instance Definition Record Format

Data type	Length (bytes)	Description
Int	2	Instance Definition Opcode 62
Unsigned Int	2	Length of the record
Int	2	Spare
Int	2	Instance definition number

Instance Reference Record Format

Data type	Length (bytes)	Description
Int	2	Instance Reference Opcode 61
Unsigned Int	2	Length of the record
Int	2	Spare
Int	2	Instance definition number

Node Primary Records

Header Record

The header record is the primary record of the header node and is always the first record in the database file. Attributes within the header record provide important information about the database file as a whole.

Format revision level indicates the OpenFlight version of the file. Correctly interpreting the attributes of other records, such as the face and vertex records, depends upon the format revision. The format revision encompasses both Flight and OpenFlight versions.

Some representative values for format revision are:

11	Flight V11
12	Flight V12
14	OpenFlight v14.0 and v14.1
1420	OpenFlight v14.2
1510	OpenFlight v15.1
1540	OpenFlight v15.4

This document describes OpenFlight version 15.7.0, therefore the attribute descriptions are based upon a format revision level of 1570.

Geographic attributes such as projection type, latitude, and longitude may be stored in the header record. The MultiGen Series II and Creator Terrain options set the value of these attributes when creating terrain databases. Positive latitudes reference the northern hemisphere and negative longitudes reference the western hemisphere.

Delta X and Y attributes indicate the placement of the database when several separate databases, each with a local origin of zero, are used to represent an area

Header Record Format.

Data Type	Length (bytes)	Description
Int	2	Header Opcode 1
Unsigned Int	2	Length of the record

Data Type	Length (bytes)	Description
Char	8	ID field (not currently used)
Int	4	Format revision level
Int	4	Edit revision level
Char	32	Date and time of last revision
Int	2	Next Group node ID number
Int	2	Next LOD node ID number
Int	2	Next Object node ID number
Int	2	Next Face node ID number
Int	2	Unit multiplier/divisor, always equal to 1
Int	1	Vertex coordinate units
		0 = Meters
		1 = Kilometers
		4 = Feet
		5 = Inches
		8 = Nautical miles
Int	1	if TRUE set texwhite on new faces
Boolean	4	Flags (bits, from left to right)
		0 = Save vertex normals
		1 = Packed Color mode
		2 = CAD View mode
		3-31 = Spare
Int	4*6	Reserved
Int	4	Projection type
		0 = Flat earth
		1 = Trapezoidal

Data Type	Length (bytes)	Description
		2 = Round earth
		3 = Lambert
		4 = UTM
Int	4*7	Reserved
Int	2	Next DOF node ID number
Int	2	Vertex storage type
		1 = Double precision float
Int	4	Database origin
		100 = OpenFlight
		200 = DIG I/DIG II
		300 = Evans and Sutherland CT5A/CT6
		400 = PSP DIG
		600 = General Electric CIV/CV/PT2000
		700 = Evans and Sutherland GDF
Double	8	Southwest database coordinate x
Double	8	Southwest database coordinate y
Double	8	Delta x to place database
Double	8	Delta y to place database
Int	2	Next sound node ID number
Int	2	Next path node ID number
Int	4*2	Reserved
Int	2	Next Clip node ID number
Int	2	Next Text node ID number
Int	2	Next BSP node ID number
Int	2	Next Switch node ID number

Data Type	Length (bytes)	Description
Int	4	Reserved
Double	8	Southwest corner latitude
Double	8	Southwest corner longitude
Double	8	Northeast corner latitude
Double	8	Northeast corner longitude
Double	8	Origin latitude
Double	8	Origin longitude
Double	8	Lambert upper latitude
Double	8	Lambert lower latitude
Int	2	Next Light source node ID number
Int	2	Next Light point node ID number
Int	2	Next Road node ID number
Int	2	Next CAT node ID number
Int	2	Reserved
Int	2	Reserved
Int	2	Reserved
Int	2	Reserved
Int	4	Earth ellipsoid model 0 = WGS 1984 1 = WGS 1972 2 = Bessel 3 = Clarke 1866 4 = NAD 1927
Int	2	Next Adaptive node ID number
Int	2	Next Curve node ID number

Data Type	Length (bytes)	Description
Int	2	Reserved
Double	8	Delta z to place database (used in conjunction with existing Delta x and Delta y values)
Double	8	Radius (distance from database origin to farthest corner)
Unsigned Short	2	Next Mesh node ID number
Unsigned Short	2	Reserved

Mesh Node Records

A mesh defines a set of geometric primitives that share attributes and vertices. In previous versions of OpenFlight, the fundamental geometric construct was the polygon, and each polygon was represented by a unique set of attributes and vertices. Meshes, by constrast, represent "sets" of related polygons, each sharing common attributes and vertices. Using a mesh, related polygons can be represented in a much more compact format. Each mesh consists of one set of "polygon" attributes (color, material, texture, etc.), a common "vertex pool" and one or more geometric primitives that use the shared attributes and vertices. Using a mesh, you can represent triangle strips, triangle fans, quadrilateral strips and indexed face sets.

A mesh node is defined by three distinct record types:

- *Mesh* - defines the "polygon" attributes associated to all geometric primitives of the mesh.
- *Local Vertex Pool* - defines the set of vertices that are referenced by the geometric primitives of the mesh.
- *Mesh Primitive* - defines a geometric primitive (triangle-strip, triangle-fan, quadrilateral-strip or indexed face set) for the mesh.

A mesh node record contains one Mesh record, one Local Vertex Pool record, and one or more Mesh Primitive records as shown in the following example:

```
MESH
LOCAL VERTEX POOL
PUSH
MESH PRIMITIVE
MESH PRIMITIVE
```

```
...
MESH PRIMITIVE
POP
```

Mesh

The mesh record is the primary record of a mesh node and defines the common "polygon-like" attributes associated to all geometric primitives of the mesh. (See “Mesh Node Records” on page 20.) To that end, the contents of the mesh record are identical to those of the Polygon (or Face) record.

Mesh Record Format

Data type	**Length (bytes)**	**Description**
Unsigned Short	2	Mesh Opcode 84
Unsigned Short	2	Length of the record
Char	8	7 character ASCII id, 0 terminates
Int	4	IR color code
Int	2	Relative priority
Int	1	Draw type
		0 = Draw solid with backface culling
		1 = Draw solid, no backface culling
		2 = Draw wireframe
		3 = Draw wireframe and close
		4 = Surround with wireframe in
		8 = Omnidirectional light
		9 = Unidirectional light
		10 = Bidirectional light

Data type	Length (bytes)	Description
Int	1	Texture white = if TRUE, draw textured face white
Unsigned Int	2	Color name index
Unsigned Int	2	Alternate color name index
Int	1	Reserved
Int	1	Template (billboard)
		0 = Fixed, no alpha blending
		1 = Fixed, alpha blending
		2 = Axial rotate
		4 = Point rotate
Int	2	Detail texture pattern index, -1 if none
Int	2	Texture pattern index, -1 if none
Int	2	Material index, -1 if none
Int	2	Surface material code (for DFAD)
Int	2	Feature ID (for DFAD)
Int	4	IR material code
Unsigned Int	2	Transparency
		0 = Opaque
		65535 = Totally clear
Unsigned Int	1	LOD generation control
Unsigned Int	1	Line style index
Boolean	4	Flags (bits from left to right)
		0 = Terrain
		1 = No color

Data type	Length (bytes)	Description
		2 = No alternate color
		3 = Packed color
		4 = Terrain culture cutout (footprint)
		5 = Hidden, not drawn
		6-31 = Spare
Unsigned Int	1	Light mode
		0 = Use face color, not illuminated
		1 = Use vertex colors, not illuminated
		2 = Use face color and vertex normal
		3 = Use vertex color and vertex normal
Unsigned Int	1	Reserved
Unsigned Int	2	Reserved
Boolean	4	Reserved
Unsigned Int	4	Packed color, primary (A, B, G, R)
Unsigned Int	4	Packed color, alternate (A, B, G, R)
Int	2	Texture mapping index
Int	2	Reserved
Unsigned Int	4	Primary color index
Unsigned Int	4	Alternate color index
Int	2	Reserved
Int	2	Reserved

Local Vertex Pool

This record defines a set of vertices that is referenced by the geometry of the mesh.

Note: Currently the Local Vertex Pool is used exclusively in the context of meshes, but it is designed in a general way so that it may appear in other contexts in future versions of the OpenFlight Scene Description.

Local Vertex Pool Record Format

Data type	**Length** (bytes)	Description
Unsigned Short	2	Local Vertex Pool Opcode 85
Unsigned Short	2	Length of the record **Note:** Since the length of this record is represented by an unsigned short, the maximum length of the vertex pool is 216 - 1 (or 65535 bytes). If the entire vertex pool cannot fit into this size, one or more Continuation records will follow. (See “Continuation Record” on page 113.)
Unsigned Int	4	**numverts**: Number of vertices contained in this record
Unsigned Int	4	**attrmask**: 32 bit mask indicating what kind of vertex information is contained in this vertex list. Bits are ordered from left to right (bit 1 is leftmost). The bits have the following definitions:

Bit #	Description
1	**Has Position** - indicats that each vertex in the list includes x, y, and z coordinates (three double-precision floating point values)

Data type	**Length** (bytes)	Description
		2 **Has Color Index**- indicates that each vertex in the list includes a color value that is a color table index (one integer value)
		3 **Has RGB Color** - indicates that each vertex in the list includes a color value that is a packed RGB color value (one integer value)
		NOTE: Bits 2 and 3 are mutually exclusive - a vertex can have either a color index or an RGB color value or neither, but cannot have both a color index and an RGB value.
		4 **Has Normal** - indicates that each vertex in the list includes a normal (three single-precision floating point values)
		5 **Has Base UV** - indicates that each vertex in the list includes uv texture coordinates for the base texture (two single-precision floating point values)
		6 **Has UV 1** - indicates that each vertex in the list includes uv texture coordinates for layer 1 (two single-precision floating point values)

Data type	**Length** (bytes)		Description
		7	**Has UV 2** - indicates that each vertex in the list includes uv texture coordinates for layer 2 (two single-precision floating point values)
		8	**Has UV 3** - indicates that each vertex in the list includes uv texture coordinates for layer 3 (two single-precision floating point values)
		9	**Has UV 4** - indicates that each vertex in the list includes uv texture coordinates for layer 4 (two single-precision floating point values)
		10	**Has UV 5** - indicates that each vertex in the list includes uv texture coordinates for layer 5 (two single-precision floating point values)
		11	**Has UV 6** - indicates that each vertex in the list includes uv texture coordinates for layer 6 (two single-precision floating point values)
		12	**Has UV 7** - indicates that each vertex in the list includes uv texture coordinates for layer 7 (two single-precision floating point values)
		13-32	**Spare** - reserved for future expansion

The following fields are repeated for each vertex in the list, depending on the bits set in the attrmask field above

Field	Data Type	Length (bytes)	Description
X_N	Double	8	X coordinate N - present if attrmask includes Has Position
Y_N	Double	8	Y coordinate N - present if attrmask includes Has Position
Z_N	Double	8	Z coordinate N - present if attrmask includes Has Position
$color_N$	Unsigned int	4	Color for vertex N - present if attrmask includes Has Color Index or Has RGB Color. Value is color table index if attrmask includes Has Color Index, packed color value (A, B, G, R) if attrmask includes Has RGB Color.
normal i_N	float	4	i component of normal for vertex N - present if attrmask includes Has Normal
normal j_N	float	4	j component of normal for vertex N - present if attrmask includes Has Normal
normal k_N	float	4	k component of normal for vertex N - present if attrmask includes Has Normal
$uBase_N$	float	4	u texture coordinate for base texture of vertex N - present if attrmask includes Has Base UV.
$vBase_N$	float	4	v texture coordinate for base texture of vertex N - present if attrmask includes Has

Field	Data Type	Length (bytes)	Description
			Base UV.
$u1_N$	float	4	u texture coordinate for layer 1 of vertex N - present if attrmask includes Has UV 1.
$v1_N$	float	4	v texture coordinate for layer 1 of vertex N - present if attrmask includes Has UV 1.
$u2_N$	float	4	u texture coordinate for layer 2 of vertex N - present if attrmask includes Has UV 2.
$v2_N$	float	4	v texture coordinate for layer 2 of vertex N - present if attrmask includes Has UV 2.
$u3_N$	float	4	u texture coordinate for layer 3 of vertex N - present if attrmask includes Has UV 3.
$v3_N$	float	4	v texture coordinate for layer 3 of vertex N - present if attrmask includes Has UV 3.
$u4_N$	float	4	u texture coordinate for layer 4 of vertex N - present if attrmask includes Has UV 4.
$v4_N$	float	4	v texture coordinate for layer 4 of vertex N - present if attrmask includes Has UV 4.
$u5_N$	float	4	u texture coordinate for layer 5 of vertex N - present if attrmask includes Has UV 5.
$v5_N$	float	4	v texture coordinate for layer 5 of vertex N - present if attrmask includes Has UV 5.
$u6_N$	float	4	u texture coordinate for layer 6 of vertex N - present if attrmask includes Has UV 6.
$v6_N$	float	4	v texture coordinate for layer 6 of vertex N -

Field	Data Type	Length (bytes)	Description
			present if attrmask includes Has UV 6.
u $_N$	float	4	u texture coordinate for layer 7 of vertex N -
			present if attrmask includes Has UV 7.
$v7_N$	float	4	v texture coordinate for layer 7 of vertex N -
			present if attrmask includes Has UV 7.

Mesh Primitive

This record defines a geometric primitive (Triangle Strip, Triangle Fan, Quadrilateral Strip, or Indexed Polygon) for a mesh.

Mesh Primitive Record Format

Data type	**Length** (bytes)	Description
Unsigned Short	2	Mesh Primitive Opcode 86
Unsigned Short	2	Length of the record
Unsigned Short	2	Primitive Type - can be one of the following values:
		1 - Triangle Strip
		2 - Triangle Fan
		3 - Quadrilateral Strip
		4 - Indexed Polygon
		Note: This field specifies how the vertices of the primitive
		are interpreted.
Unsigned Short	2	Specifies the length (in bytes) of each of the vertex indices
		that follow - will be 1, 2, or 4
Unsigned Int	4	Number of vertices contained in this primitive.

The following field is repeated for each vertex in the primitive. These vertices are interpreted according to the Primitive Type field, above.

Field	Data Type	Length (bytes)	Description
vertex index$_N$	var	var	Index of vertex N of the mesh primitive. The size of each of these indices is specified in the indexsize field above.

Each mesh primitive is represented using the Mesh Primitive record above. The following descriptions explain how the vertices of each primitive type are interpreted as geometry:

- **Triangle Strip** - This mesh primitive defines a connected group of triangles in the context of the enclosing mesh. Each triangle shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One triangle is defined for each vertex presented after the first two vertices. For odd n, vertices n, n + 1, and n + 2 define triangle n. For even n, vertices n + 1, n, and n + 2 define triangle n. N vertices represent N - 2 triangles.
- **Triangle Fan** - Like the Triangle Strip, this mesh primitive also defines a connected group of triangles in the context of the enclosing mesh. Each triangle shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One triangle is defined for each vertex presented after the first two vertices. Vertices 1, n + 1, and n + 2 define triangle n. N vertices represent N - 2 triangles.
- **Quadrilateral Strip** - This mesh primitive defines a connected group of quadrilaterals in the context of the enclosing mesh. Each quadrilateral shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One quadrilateral is defined for each pair of vertices presented after the first pair. Vertices 2n - 1, 2n, 2n + 2, and 2n + 1 define quadrilateral n. N vertices represent N-3 quadrilaterals.
- **Indexed Polygon** -This mesh primitive defines a single polygon in the context of the enclosing mesh. This primitive is similar to the other mesh primitives in that it also shares the polygon attributes of the enclosing mesh. It is different from the other mesh primitive types in that while triangle strips/ fans and quadrilateral strips describe a set of connected triangles/quadrilaterals, the indexed polygon defines a single polygon. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One polygon is defined by the sequence of vertices in this record. N vertices represent 1 N-sided closed polygon or 1 (N-1)-sided unclosed polygon.

Group Record

The group record is the primary record of the group node. Groups are the most generic hierarchical node present in the database tree. Attributes within the group record provide bounding volumes that encompass the group's children and real-time control flags.

Relative priority specifies a fixed ordering of the group relative to its sibling nodes. Ordering is from left (lesser values) to right (higher values). Nodes of equal priority may be arbitrarily ordered. All nodes have an implicit (default) value of zero.

Animation flags indicate that a group's immediate children represent an animation sequence, each child node being one frame of the sequence. The value of each flag indicates the animation should cycle forward, or forwards and backwards. Forward animations cycle from the first child node to the last child node. The animation may stop after the last child or repeat, starting over with the first child. Swing animations cycle forwards and backwards in a "swinging" manner.

Special effect ID1 and ID2 are application-defined attributes. Their values can be used to enhance the meaning of existing attributes, such as the animation flags, or extend the interpretation of the group node. Normally, the value of these attributes is zero.

Significance can be used to assist real-time culling and load balancing mechanisms, by defining the visual significance of this group with respect to other groups in the database. Normally the value of this attribute is zero.

Layer ID is used by the MultiGen Instrumentation Option to identify (for display) a collection of groups, independent of their locations in the hierarchy. Normally the value of this attribute is zero.

Group Record Format

Data type	Length (bytes)	Description
Int	2	Group Opcode 2
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	2	Relative priority
Int	2	Reserved
Boolean	4	Flags (bits, from left to right)
		0 = Reserved

Group Record Format

		1 = Forward animation
		2 = Swing animation
		3 = Bounding box follows
		4 = Freeze bounding box
		5 = Default parent
		6-31 = Spare
Int	2	Special effect ID1 - application defined
Int	2	Special effect ID2 - application defined
Int	2	Significance
Int	1	Layer code
Int	1	Reserved
Int	4	Reserved

Object Record

The object record is the primary record of the object node. Objects are low-level grouping nodes that contain attributes pertaining to the state of it child geometry. Only face and light point nodes may be the children of object nodes.

The time-of-day object flags can be used to inhibit the display of certain objects, depending on the current time of day.

The illumination flag, when set, makes an object self-illuminating, and is not subject to lighting calculations. In practice, geometric normals should be ignored.

The flat shading flag, when set, indicates that lighting calculations should produce a faceted appearance to the object's geometry. In practice, geometric normals should be constrained to face normals.

The shadow flag indicates the object represents the shadow of the rest of the group. When used as part of a moving model (e.g., an aircraft), the application can apply appropriate distortions, creating a realistic shadow on the terrain or runway.

Relative priority specifies a fixed ordering of the object relative to its sibling nodes. Ordering is from left (lesser values) to right (higher values). Nodes of equal priority may be arbitrarily ordered. All nodes have an implicit (default) value of zero.

Transparency applies to all an object's children (geometry). The value should be modulated with each face's transparency and material alpha calculation, as described in the Face Record and Material Record sections.

Object Record Format

Data type	Length (bytes)	Description
Int	2	Object Opcode 4
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Boolean	4	Flags (bits from to right)
		0 = Don't display in daylight
		1 = Don't display at dusk
		2 = Don't display at night
		3 = Don't illuminate
		4 = Flat shaded
		5 = Group's shadow object
		6-31 = Spare
Int	2	Relative priority
Unsigned Int	2	Transparency
		0 = Opaque
		65535 = Totally clear
Int	2	Special effect ID1 - application defined
Int	2	Special effect ID2 - application defined
Int	2	Significance
Int	2	Spare

Face Record

The face record is the primary record of the face node. A face contains attributes describing the visual state of its child vertices. Only vertex and morph vertex nodes may be children of faces. This should not be confused with the fact that faces may have subfaces.

If a face contains a non-negative material code, its apparent color is a combination of the face color and material color, as described in the Material Record section. If a face contains a non-addictive material with an alpha component and the transparency field is set, the total transparency is the product of the material alpha, face, and object transparency.

If a face is a unidirectional or bidirectional light point, the face record is followed by a vector record (Vector Opcode 50) that contains the unit vector indicating the direction in which the primary color is displayed. For bidirectional light points, the alternate color is displayed in the opposite direction (180 degrees opposed).

Note: This method of defining light points is obsolete after version 15.2. Such light point faces will be turned into the new light point record when it is read into MultiGen II v1.4 or later.

Relative priority specifies a fixed ordering of the face relative to its sibling nodes. Ordering is from left (lesser values) to right (higher values). Nodes of equal priority may be arbitrarily ordered. All nodes have an implicit (default) value of zero.

Face Record Format

Data type	Length (bytes)	Description
Int	2	Face Opcode 5
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	IR color code
Int	2	Relative priority
Int	1	Draw type
		0 = Draw solid with backface culling
		1 = Draw solid, no backface culling
		2 = Draw wireframe

Data type	Length (bytes)	Description
		3 = Draw wireframe and close
		4 = Surround with wireframe in alternate color
		8 = Omnidirectional light
		9 = Unidirectional light
		10 = Bidirectional light
Int	1	Texture white = if TRUE, draw textured face white
Unsigned Int	2	Color name index
Unsigned Int	2	Alternate color name index
Int	1	Reserved
Int	1	Template (billboard)
		0 = Fixed, no alpha blending
		1 = Fixed, alpha blending
		2 = Axial rotate
		4 = Point rotate
Int	2	Detail texture pattern index, -1 if none
Int	2	Texture pattern index, -1 if none
Int	2	Material index, -1 if none
Int	2	Surface material code (for DFAD)
Int	2	Feature ID (for DFAD)
Int	4	IR material code
Unsigned Int	2	Transparency
		0 = Opaque
		65535 = Totally clear

Data type	**Length (bytes)**	**Description**
Unsigned Int	1	LOD generation control
Unsigned Int	1	Line style index
Boolean	4	Flags (bits from left to right)
		0 = Terrain
		1 = No color
		2 = No alternate color
		3 = Packed color
		4 = Terrain culture cutout (footprint)
		5 = Hidden, not drawn
		6-31 = Spare
Unsigned Int	1	Light mode
		0 = Use face color, not illuminated
		1 = Use vertex colors, not illuminated
		2 = Use face color and vertex normal
		3 = Use vertex color and vertex normal
Unsigned Int	1	Reserved
Unsigned Int	2	Reserved
Boolean	4	Reserved
Unsigned Int	4	Packed color, primary (A, B, G, R)
Unsigned Int	4	Packed color, alternate (A, B, G, R)
Int	2	Texture mapping index
Int	2	Reserved
Unsigned Int	4	Primary color index
Unsigned Int	4	Alternate color index

Data type	Length (bytes)	Description
Int	2	Reserved
Int	2	Reserved

Light Point Record

The light point record is the primary record of the light point node. A light point contains attributes describing the visual state of its child vertices. Only vertex nodes may be children of light point nodes.

Light points are geometric points that represent real world light sources such as runway lights, vehicle lights, street lights, and rotating beacons. Light points differ from light sources in that they do not illuminate the scene around them. They are primarily used to model important visual cues without incurring the tremendous rendering overhead associated with light sources.

Most light point attributes are specific to these unique requirements. Light points can be displayed on special purpose calligraphic imaging systems, the more familiar raster variety, or even hybrid raster/calligraphic (RASCAL) systems.

Light Point Record Format

Data type	Length (bytes)	Description
Int	2	Light Point Record Opcode 111
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	2	Surface material code (for DFAD)
Int	2	Feature ID (for DFAD)
Unsigned Int	4	Back color for all bidirectional points
Int	4	Display mode
		0 = RASTER

Data type	Length (bytes)	Description
		1 = CALLIGRAPHIC
		2 = EITHER
Float	4	Intensity - scalar for front colors
Float	4	Back intensity - scalar for back color
Float	4	Minimum defocus - limit (0.0 - 1.0) for calligraphic points
Float	4	Maximum defocus - limit (0.0 - 1.0) for calligraphic points
Int	4	Fading mode
		0 = enable perspective fading calculations
		1 = disable calculations
Int	4	Fog Punch mode
		0 = enable fog punch through calculations
		1 = disable calculations
Int	4	Directional mode
		0 = enable directional calculations
		1 = disable calculations
Int	4	Range mode
		0 = use depth (Z) buffer calculation
		1 = use slant range calculation
Float	4	Minimum pixel size
		Minimum diameter of points in pixels
Float	4	Maximum pixel size
		Maximum diameter of points in pixels

Data type	Length (bytes)	Description
Float	4	Actual size Actual diameter of points in database coordinates
Float	4	Transparent falloff pixel size Diameter in pixels when points become transparent
Float	4	Transparent falloff exponent >= 0 - falloff multiplier exponent (1.0 = linear falloff)
Float	4	Transparent falloff scalar > 0 - falloff multiplier scale factor
Float	4	Transparent falloff clamp Minimum permissible falloff multiplier result
Float	4	Fog scalar >= 0 - adjusts range of points for punch threw effect.
Float	4	Reserved
Float	4	Size difference threshold Point size transition hint to renderer
Int	4	Directional type 0 = OMNIDIRECTIONAL 1 = UNIDIRECTIONAL 2 = BIDIRECTIONAL
Float	4	Horizontal lobe angle - total angle in degrees

Data type	Length (bytes)	Description
Float	4	Vertical lobe angle - total angle in degrees
Float	4	Lobe roll angle - rotation of lobe about local Y axis in degrees
Float	4	Directional falloff exponent >= 0 - falloff multiplier exponent (1.0 = linear falloff)
Float	4	Directional ambient intensity - of points viewed off axis
Float	4	Animation period in seconds
Float	4	Animation phase delay in seconds - from start of period
Float	4	Animation enabled period in seconds
Float	4	Significance - drop out priority for RASCAL lights (0.0 - 1.0)
Int	4	Calligraphic draw order - for rendering consistency
Boolean	4	Flags (bits, from left to right) 0 = reserved 1 = No back color TRUE = don’t use back color for bidirectional points FALSE = use back color for bidirectional points 2 = reserved 3 = Calligraphic proximity occulting (Debunching)

Data type	Length (bytes)	Description
		4 = Reflective, non-emissive point
		5-7 = Randomize intensity
		0 = never
		1 = low
		2 = medium
		3 = high
		8 = Perspective mode
		9 = Flashing
		10 = Rotating
		11 = Rotate Counter Clockwise
		Direction of rotation about local Z axis
		12 = reserved
		13-14 = Quality
		0 = Low
		1 = Medium
		2 = High
		3 = Undefined
		15 = Visible during day
		16 = Visible during dusk
		17 = Visible during night
		18-31 = Spare
Float	4	Axis of rotation for rotating animation, x coordinate
Float	4	Axis of rotation for rotating animation, y coordinate

Data type	Length (bytes)	Description
Float	4	Axis of rotation for rotating animation, z coordinate

Degree-of-Freedom Record

The degree-of-freedom (DOF) record is the primary record of the DOF node. The DOF node specifies a local coordinate system and the range allowed for translation, rotation, and scale with respect to that coordinate system.

The DOF record can be viewed as a series of applied transformations consisting of the following elements:

[PTTTRRRSSSP]

where "P" denotes "put," "T" denotes "translate," "R" denotes "rotate," and "S" denotes "scale."

It is important to understand the order in which these transformations are applied to the geometry. A pre-multiplication is assumed, so the sequence of transformations must be read from right to left, in order to describe its effect on the geometry contained below the DOF. In this manner, a DOF is interpreted as a Put followed by three Scales, three Rotates, three Translates, and a Put.

Taking the transformations in right to left order, they represent:

1. A Put (3 point to 3 point transformation). This matrix brings the DOF coordinate system to the world origin, with its x-axis aligned along the world x-axis and its y-axis in the world x-y plane. Testing against the DOF's constraints is performed in this standard position. This matrix is therefore the inverse of the last (See Step 11 below).

2. Scale in x.

3. Scale in y.

4. Scale in z.

5. Rotation about z (yaw).

6. Rotation about y (roll).

7. Rotation about x (pitch).

8. Translation in x.
9. Translation in y.
10. Translation in z.
11. A final Put. This matrix moves the DOF coordinate system back to its original position in the scene.

The DOF record specifies the minimum, maximum, and current values for each transformation. Only the current value affects the actual transformation applied to the geometry. The increment value specifies discrete allowable values within the range of legal values represented by the DOF.

Degree-of-Freedom Record Format

Data type	Length (bytes)	Description
Int	2	Degree-of-Freedom Opcode 14
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Double	8	Origin of the DOF’s local coordinate system; x coordinate
Double	8	Origin of the DOF’s local coordinate system; y coordinate
Double	8	Origin of the DOF’s local coordinate system; z coordinate
Double	8	Point on the x axis of the DOF’s local coordinate system; x coordinate
Double	8	Point on the x axis of the DOF’s local coordinate system; y coordinate
Double	8	Point on the x axis of the DOF’s local coordinate system; z coordinate
Double	8	Point in xy plane of the DOF’s local coordinate system; x coordinate
Double	8	Point in xy plane of the DOF’s local coordinate system; y coordinate
Double	8	Point in xy plane of the DOF’s local coordinate system; z coordinate
Double	8	Minimum z value with respect to the local coordinate system

Data type	Length (bytes)	Description
Double	8	Maximum z value with respect to the local coordinate system
Double	8	Current z value with respect to the local coordinate system
Double	8	Increment in z
Double	8	Minimum y value with respect to the local coordinate system
Double	8	Maximum y value with respect to the local coordinate system
Double	8	Current y value with respect to the local coordinate system
Double	8	Increment in y
Double	8	Minimum x value with respect to the local coordinate system
Double	8	Maximum x value with respect to the local coordinate system
Double	8	Current x value with respect to the local coordinate system
Double	8	Increment in x
Double	8	Minimum pitch (rotation about the x axis)
Double	8	Maximum pitch
Double	8	Current pitch
Double	8	Increment in pitch
Double	8	Minimum roll (rotation about the y axis)
Double	8	Maximum roll
Double	8	Current roll
Double	8	Increment in roll
Double	8	Minimum yaw (rotation about the z axis)
Double	8	Maximum yaw
Double	8	Current yaw
Double	8	Increment in yaw
Double	8	Minimum z scale (about local origin)

Data type	**Length (bytes)**	**Description**
Double	8	Maximum z scale (about local origin)
Double	8	Current z scale (about local origin)
Double	8	Increment for scale in z
Double	8	Minimum y scale (about local origin)
Double	8	Maximum y scale (about local origin)
Double	8	Current y scale (about local origin)
Double	8	Increment for scale in y
Double	8	Minimum x scale (about local origin)
Double	8	Maximum x scale (about local origin)
Double	8	Current x scale (about local origin)
Double	8	Increment for scale in x
Boolean	4	Flags (bits, from left to right) 0 = x translation is limited 1 = y translation is limited 2 = z translation is limited 3 = Pitch rotation is limited 4 = Roll rotation is limited 5 = Yaw rotation is limited 6 = x scale is limited 7 = y scale is limited 8 = z scale is limited 9 = Reserved 10 = Reserved 11-31 = Spare

Vertex List Record

A vertex list record is the primary record of a vertex node. Each record references one or more vertices in the vertex palette. A vertex node is a leaf node in the database and therefore cannot have any children.

Vertex List Record Format

Data type	Length (bytes)	Description
Int	2	Vertex List Opcode 72
Unsigned Int	2	Length of the record
Int	4	Byte offset into vertex palette of the actual vertex

Note: number of vertices in the list is determined by: (Length of this record - 4) / 4

Morph Vertex List Record

A morph vertex list record is the primary record of a morph vertex node. A morph vertex node is a leaf node in the database and therefore cannot have any children.

Each record references one or more pairs of vertices (weights) in the vertex palette. One weight is the 0 percent morph attributes and the other weight is the 100 percent morph attributes. Since each weight references a vertex, all vertex attributes including color, normal, and texture coordinates may be morphed.

When the eyepoint approaches the switch-in distance, the vertex attributes displayed are 100 percent morphed. When the eyepoint reaches the distance computed by LOD switch-in distance minus LOD transition range, the vertex attributes displayed are 0 percent morphed. Within the LOD transition range, the vertex attributes displayed are interpolated between the two known vertex attributes.

Geometric morphing is controlled by the parent LOD node. Only morph vertex nodes are affected. Both morphing and static geometry (vertices) may exist within the same branch of the database hierarchy.

Morph Vertex List Record Format

Data type	Length (bytes)	Description
Int	2	Morph Vertex List Opcode 89
Unsigned Int	2	Length of the record
Int	4	Byte offset into vertex palette of the 0% vertex
Int	4	Byte offset into vertex palette of the 100% vertex

Note: number of vertices in the list is determined by: (Length of this record - 4) / 8

Binary Separating Plane Record

The binary separating plane (BSP) record is the primary record of the BSP node. A BSP allows you to model 3D databases without depth (Z) buffer support.

An application uses this information to cull portions of the database according to which side of the plane the subtree is situated on with regard to eyepoint position and viewing direction.

This record contains an equation ax + by + cz + d = 0 that describes the separating plane.

Binary Separating Plane Record Format

Data type	Length (bytes)	Description
Int	2	Binary Separating Plane (BSP) Opcode 55
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Double	8	First plane equation coefficient (a)
Double	8	Second plane equation coefficient (b)

Data type	Length (bytes)	Description
Double	8	Third plane equation coefficient (c)
Double	8	Fourth plane equation coefficient (d)

External Reference Record

The external reference record is the primary record of the external reference node. External references allow one database to reference, or instance, a node in another database (or an entire database). At the point of reference, the referenced node/database is copied (or possibly shared) as a child of the current parent node.

The override flags allow the referencing (parent) database to control use of the referenced (child) node/database palettes. If an override flag (e.g., material) is set, the child node/database uses its own (material) palette. Otherwise, the child node/database uses the current (parent's) palette. The override flags are hierarchical and may affect references made by the child node/database.

External Reference Record Format

Data type	Length (bytes)	Description
Int	2	External Reference Opcode 63
Unsigned Int	2	Length of the record
Char	200	199-char ASCII path; 0 terminates
		(format: filename <node name>; if absent, references entire file)
Int	1	Reserved
Int	1	Reserved
Int	2	Reserved
Boolean	4	Flags (bits, from left to right)

Data type	Length (bytes)	Description
		0 = Color palette override
		1 = Material palette override
		2 = Texture and texture mapping palette override
		3 = Line style palette override
		4 = Sound palette override
		5 = Light source palette override
		6-31 = Spare
Int	2	Reserved

Level-of-Detail Record

The level-of-detail (LOD) record is the primary record of the LOD node. LOD’s are perhaps the most important hierarchical node present in the database tree. Proper use of level-of-detail modeling concepts can vastly improve real-time playback of large databases. Attributes within the LOD record provide switching and transition distances for real-time culling and load management mechanisms.

The center coordinate can be used by a real-time application to calculate the slant range distance from the eyepoint to the LOD. Based upon the result of this calculation, a real-time application can choose not to display the LOD’s children and thus reduce system load. The center of the LOD is generally the transformed center of the geometry of the LOD’s children. This should include the effects of instancing and (parent) group replication as well.

The use previous slant range flag indicates that the slant range for this LOD is the same as the previous (sibling) LOD, implying the center coordinate is also the same. The real-time application can reuse the previous slant range calculation when evaluating this LOD, thereby improving performance.

Transition range specifies the range over which real-time smoothing effects should be employed while switching from one LOD to another. Smoothing effects include geometric morphing and image blending. The smoothing effect is active between: switch-in distance minus transition range (near), and switch-in distance (far). The center distance of the effect is therefore switch-in distance minus one half the transition range.

Level-of-Detail Record Format

Data type	Length (bytes)	Description
Int	2	Level-of-Detail Opcode 73
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Double	8	Switch-in distance
Double	8	Switch-out distance
Int	2	Special effect ID1 - application defined
Int	2	Special effect ID2 - application defined
Boolean	4	Flags (bits, from left to right)
		0 = Use previous slant range
		1 = Reserved
		2 = Freeze center (don't recalculate)
		3-31 = Spare
Double	8	Center coordinate x of LOD
Double	8	Center coordinate y of LOD
Double	8	Center coordinate z of LOD
Double	8	Transition range

Sound Record

The sound record is the primary record of the sound node. A sound node represents the position and orientation of a sound emitter in the database.

Amplitude and pitch blend are relative to the amplitude in the waveform file.

Priority determines which sounds are played when more emitters populate a scene than the sound system can play simultaneously.

Falloff defines how amplitude falls off when approaching the edge of the sound lobe, with maximum amplitude at the center of the lobe.

Width defines the half angle of the sound lobe.

Doppler, absorption, and delay flags enable or disable the modeling of Doppler, atmospheric absorption, and propagation delay in the sound environment.

Direction sets the type of sound lobe: omnidirectional = 0, bidirectional = 1, or unidirectional = 2.

Active indicates a sound is to be activated when read in.

Sound Record Format

Data type	Length (bytes)	Description
Int	2	Sound Node Opcode 91
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Index into sound palette
Int	4	Reserved
Double	8	x coordinate of offset from local origin
Double	8	y coordinate of offset from local origin
Double	8	z coordinate of offset from local origin
Float	4	i component of sound direction wrt local coordinate axes
Float	4	j component of direction wrt local coordinate axes
Float	4	k component of direction wrt local coordinate axes
Float	4	Amplitude of sound
Float	4	Pitch bend of sound
Float	4	Priority of sound

Data type	Length (bytes)	Description
Float	4	Falloff of sound
Float	4	Width of sound lobe
Boolean	4	Flags (bits, from left to right) 0 = Doppler 1 = Atmospheric absorption 2 = Delay 3-4 = Direction: 0 = Omnidirec-tional 1 = Unidirec-tional 2 = Bidirec-tional 5 = Active 6-31 = Spare

Light Source Record

The light source record is the primary record of the light source node. Light sources illuminate the database. They contain position and rotation data (overriding any information stored in the light palette), an index into the light palette, and information on how the light behaves within the hierarchy.

The enabled flag indicates whether the light is turned on and, therefore, a factor of the lighting (rendering) model.

The global flag specifies whether the light shines on the entire database or only on its children (for example, the cabin light in a car).

Light Source Record Format

Data type	Length (bytes)	Description
Int	2	Light Source Record Opcode 101
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Index into light palette
Int	4	Reserved
Char	4	Flags (bits, from left to right)
		0 = Enabled
		1 = Global
		2 = Reserved
		3 = Export
		4 = Reserved
		5-31 = Spare
Int	4	Reserved
Double	8	Local or spot position x coordinate
Double	8	Local or spot position y coordinate
Double	8	Local or spot position z coordinate
Float	4	Infinite or spot yaw
Float	4	Infinite or spot pitch

Road Segment Record

A road segment record is the primary record of a road segment node. It stores the attributes used to create and modify a road segment. The children of the road node represent the geometry and paths of the road and should not be manually edited. Any modification invalidates the road segment.

Road Segment Record Format

Data type	Length (bytes)	Description
Int	2	Road Segment Opcode 87
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates

Road Construction Record

A road construction record is the primary record of a road construction node. It supersedes the Road Segment Record described previously. It is created by the Pathfinder option of MultiGen II Pro v1.5 as well as the Road Tool option beginning with Creator v2.1. It stores the parameters defining the road path construction for one road section. In practice, the children of the road construction node usually represent the geometry and paths of the road section. Although every field in the road construction record may be modified, this data makes the most sense when it is kept in sync with the geometry that is created from it. Therefore, typical usage will be read-only access from applications able to analyze the road surface from this given data.

The Road type field dictates how the following fields define the current road section. For all road types, the Entry and Exit control points lie on the boundaries of the road section. The Alignment control point is only necessary for the Curve type as it defines a horizontal tangent with the other control points.

Other fields particular to the Curve type are the horizontal curve parameters. The horizontal components of the Curve type start and end with spiral transitions of specified lengths. An Arc Radius length is used to define the constant curve area. The Superelevation is specified in a rise over run slope measured laterally across the road for the maximum banking which is used throughout the constant curve component. The banking transitions along the spiral sections in one of three ways defined by the Spiral type field.

Both the Curve and Hill types may have a vertical curve component defined by the remaining fields. Slopes are given at both the entry and exit of the section. If the given slopes don't intersect within the road segment then two vertical parabolas are constructed instead of one, and the Additional vertical parabola flag is set. Note that this flag's value is only valid when the Road Tools version field is 3 or later. This flag may also be set when convergence of the slopes creates a vertical curve length less than Minimum curve length. Otherwise, Vertical curve length is used to define the horizontal distance covered by the single parabola vertical curve.

Road Construction Record Format

Data type	Length (bytes)	Description
Int	2	Road Construction Opcode 127
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates
Char	4	Reserved
Int	4	Road type
		0 = Curve
		1 = Hill
		2 = Straight
Int	4	Road Tools version
Double	8*3	Entry control point
Double	8*3	Alignment control point
Double	8*3	Exit control point
Double	8	Arc radius
Double	8	Entry spiral length
Double	8	Exit spiral length
Double	8	Superelevation
Int	4	Spiral type
		0 = Linear with length
		1 = Linear with angle
		2 = Cosine with length

Data type	Length (bytes)	Description
Int	4	Additional vertical parabola flag
Double	8	Vertical curve length
Double	8	Minimum curve length
Double	8	Entry slope
Double	8	Exit slope

Road Path Record

A road path record is the primary record of a road path node. A road path node is a child of a road segment node. It describes a lane of the parent road segment. The child of a road path node is a face node whose vertices provide the coordinates of the center of the lane.

Road path record attributes may also be written to an ASCII file for easy access by the application. The format of the file is described in "Road Path Files," page 109.

Road Path Record Format

Data type	Length (bytes)	Description
Int	2	Road Path Opcode 92
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Char	120	Path name
Double	8	Speed limit
Boolean	4	No passing
Int	4	Vertex normal type

Data type	Length (bytes)	Description
		0 = Up-vector
		1 = Heading, Pitch, Roll
Int	480	Spare

Clip Region Record

A clip region record is the primary record of a clip node. It defines those regions in 3D space in which drawing occurs. Clip regions only clip the geometry below the clip node in the hierarchy.

The coordinates create a four-sided face that defines the clip region in space. Planes are formed along the edges of the four-sided face normal to the face; a fifth plane clips the back side of the face.

Clip Region Record Format

Data type	Length (bytes)	Description
Int	2	Clip Region Opcode 98
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	2	Reserved
Char	5	Flags for enabling the individual clip planes
		(char 1 is the plane on edge defined by the 1st two
		coordinates etc.; char 5 enables the plane that clips
		the half space behind the face)
Char	1	Reserved

Data type	Length (bytes)	Description
Double	8*12	Four coordinates defining the clip region (x0, y0, z0, x1, y1, z1, x2...)
Double	8*20	Five plane equation coefficients (ax + by +cz + d) (a0, a1, a2, a3, a4, b0, b1, b2, b3, b4, c0, c1, c2, c3, c4, d0, d1, d2, d3, d4)

Text Record

The text record is the primary record of the text node. Text draws a string of data using a specified font. The record specifies the visual characteristics of the text and formatting information.

The actual string for the text is stored in the comment record immediately following. The format of the text record is:

Text Record Format

Data type	Length (bytes)	Description
Int	2	Text Opcode 95
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Type -1 = Static 0 = Text String 1 = Float

Data type	Length (bytes)	Description	
			2 = Integer
Int	4	Draw type	
			0 = Solid
			1 = Wireframe and close
			2 = Wireframe
			3 = Surround with wireframe in alternate color
Int	4	Justification	
			0 = Left
			1 = Right
			2 = Center
Double	8	Floating point value	
Int	4	Integer value	
Int	4*5	Reserved	
Int	4	Flags	
			Bit 0 = Boxable (Unused)
			Bits 1-31 = Spare
Int	4	Color	
Int	4	Color 2 (Unused)	
Int	4	Material	
Int	4	Spare	
Int	4	Maximum number of lines (Unused)	
Int	4	Maximum number of characters	
Int	4	Current length of text (Unused)	
Int	4	Next line number available (Unused)	

Data type	Length (bytes)	Description
Int	4	Line number at top of display (Unused)
Int	4*2	Low/high values for integers
Double	8*2	Low/high values for floats
Double	8*3	Lower-left corner of rectangle around text
Double	8*3	Upper-right corner of rectangle around text
Char	120	Font name
Int	4	Draw vertical
Int	4	Draw with italic slant factor
Int	4	Draw with underline
Int	4	Line style

Switch Record

A switch record is the primary record of a switch node. A switch represents a set of masks that control the display of the switch's children.

Each mask contains one bit for each child of the switch. Each mask bit indicates that the corresponding child is selected (1) or deselected (0). Each mask selects some, none, or all of the children for display according to the state of the mask bits.

Both the switch children and mask bits begin counting from 0. Therefore the selection state, for a particular switch child is derived from a given mask with the following calculation:

```
mask_bit = 1 << (child_num % 32)
mask_word = mask_words [mask_num * num_words + child_num / 32]
child_selected = mask_word & mask_bit
```

The current mask value is an index into the set of masks and indicates the selected mask.

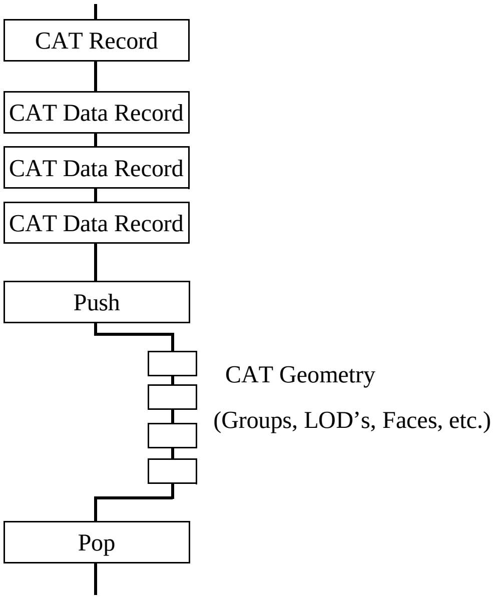
Switch Record Format

Data type	Length (bytes)	Description
Int	2	Switch Opcode 96
Unsigned Int	2	Length of record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Current mask
Int	4	Number of 32 bit words required for each mask (number of children / 32 + number of children modulo 32)
Int	4	Number of masks
Unsigned Int	Variable	Mask words (length = number of words per mask * number of masks * 4 bytes)

CAT Record

A CAT record is the primary record of a CAT node. A Continuously Adaptive Terrain skin is a hierarchical triangle mesh designed for high fidelity, real-time viewing.

A CAT skin is represented in OpenFlight by a record stream consisting of: a CAT record, a set of CAT data records, a push record, the CAT hierarchy and geometry, and a pop record. CAT hierarchy and geometry is represented by standard OpenFlight constructs of LOD's, groups, external references, faces, and vertices.



CAT Record Format

Data type	Length (bytes)	Description
Int	2	CAT Opcode 115
Unsigned Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	IR color code
Int	2	Relative priority
Int	1	Draw type
		0 = Hidden, don't draw
		1 = Draw solid, no backface
		2 = Draw wireframe
Int	1	Texture white = if TRUE, draw textured face white
Int	2	Reserved

Data type	Length (bytes)	Description
Unsigned Int	2	Color name index
Unsigned Int	2	Alternate color name index
Int	2	Detail texture pattern index, -1 if none
Int	2	Texture pattern index, -1 if none
Int	2	Material index, -1 if none
Int	2	Surface material code (for DFAD)
Int	2	Feature ID (for DFAD)
Int	4	IR material code
Int	4*2	Reserved
Int	2	Texture mapping index
Int	2	Reserved
Unsigned Int	4	Primary color index
Unsigned Int	4	Alternate color index
Int	4*2	Reserved
Boolean	4	Flags (bits, from left to right)
		0 = No color
		1 = No alternate color
		2-31 = Spare
Int	4	Reserved

Extension Node Record

An extension node record is the primary record of an extension node. It introduces a user defined node type that is defined by an extension site that utilizes the extensibility of the OpenFlight format. It specifies the site information for a third party record which contains additional data that is not represented by the standard OpenFlight records. The content of the data itself is transparent to users other than the extension site. The data can be accessed by the combination of the OpenFlight API and the data dictionary defined by the extension site.

The relationship of an extension node relative to other hierarchical nodes is defined by the standard push and pop control records. For more information about extensions, please refer to the "OpenFlight API User's Guide, Level 3: Extensions".

The extension record (Opcode 100) may also introduce a new attributes to existing nodes (See "Extension Attribute Record" on page 84.)

Extension Node Record Format

Data type	Length (bytes)	Description
Int	2	Extension Opcode 100
Unsigned Int	2	Length of the total extension record
Char	8	7 char ASCII ID; 0 terminates
Char	8	Site ID - Unique site name
Int	1	Reserved
Int	1	Revision - site specific
Unsigned Int	2	Record code - site specific
Char	n/a	Extended data - site specific

Curve Record

A curve record is the primary record of a curve node. A curve node represents one or more curve segments joined together with at least G^0 continuity. Let a curve segment be defined by 4 geometric constraints. We will call these geometric constraints control points in the curve record. The way the control points are grouped and used will be discussed below.

Let each control point be a double precision 3D coordinate, $P = (x, y, z)$.

Let the group of control points be $(P_0, P_1, ..., P_k)$.

The currently defined curve types are B-spline, Cardinal (also known as Catmull-Rom,) and Bezier.

If the curve type is Bezier, P_0, P_1, P_2, and P_3 form the first curve segment. P_3, P_4, P_5, and P_6 form the next segment, and so on. Notice that the last control point in the first segment becomes the first control point in the second segment.

If the curve type is either B-spline or Cardinal, P_0, P_1, P_2, and P_3 form the first curve segment. P_1, P_2, P_3, and P_4 from the next segment, and so on. Notice that the second control point in the first segment becomes the first control point in the second segment.

Note that the smoothness of the curve depends on how many times your renderer samples the curve equation into piece-wise linear elements. In MultiGen products, each curve segment is evenly sampled 11 times to produce 10 lines per curve segment.

Curve Record Format

Data type	**Length (**bytes**)**	Description
Int	2	Curve Opcode 126
Unsigned Int	2	Length of the total curve record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Curve type
		4 = B-spline
		5 = Cardinal
		6 = Bezier
Int	4	Number of control points
Char	8	Reserved
Double	Variable	Coordinates of control points consisting of triplets of doubles.
		(Length = number of control points * 3 * 8 bytes.)

Ancillary Records

Ancillary records follow node primary records. They contain supplementary attribute data for the node. Ancillary records are optional but must precede any control record, following the node primary record, when present. There is no order dependency between ancillary records.

Comment Record

A comment record is an ancillary record that contains text data that belongs to the preceding node primary record. The text description is a variable length ASCII string terminated by a <nil> character.

Comment Record Format

Data type	**Length** (bytes)	Description
Int	2	Text Comment Opcode 31
Unsigned Int	2	Length of the record
Char	(Length - 4)	Text description of node

Long ID Record

A long ID record is an ancillary record that contains the full name of the preceding node. It is present only when the name exceeds eight characters (seven characters plus a terminating <nil> character.)

Long ID Record Format

Data type	**Length** (bytes	Description
Int	2	Long Identifier Opcode 33
Unsigned Int	2	Length of the record
Char	(Length - 4)	ASCII ID of node

MultiTexture Record

OpenFlight supports eight textures per polygon or mesh as well as eight uv values per vertex. The current texture information stored on the polygon is referred to as "the base texture" or "texture layer 0". Each additional texture is referred to as "texture layer N". Therefore, to support eight textures per polygon, a base texture is required as well as seven additional texture layers. The additional texture layers for each polygon, mesh, and vertex will be represented in ancillary records at the Face, Mesh and Vertex primary node level as shown in the following example:

FACE
MULTITEXTURE
PUSH
VERTEX LIST
UV LIST
POP

The MultiTexture record is an ancillary record of Face and Mesh nodes. It specifies the texture layer information for the face or mesh.

Long ID Record Format

Data type	**Length** (bytes)	Description	
Unsigned Short	2	MultiTexture opcode 52	
Unsigned Short	2	Length of the record	
Unsigned Int	4	32 bit mask indicating what kind of multitexture information follows. The bits appear from left to right and have the following definitions:	
		1	**Has Layer 1** - indicates that multitexture information for texture layer 1 is present.
		2	**Has Layer 2** - indicates that multitexture information for texture layer 2 is present.
		3	**Has Layer 3** -- indicates that multitexture information for texture layer 3 is present.

Data type	**Length** (bytes)	Description	
		4	**Has Layer 4** -- indicates that multitexture information for texture layer 4 is present.
		5	**Has Layer 5** - - indicates that multitexture information for texture layer 5 is present.
		6	**Has Layer 6** -- indicates that multitexture information for texture layer 6 is present.
		7	**Has Layer 7** - - indicates that multitexture information for texture layer 7 is present.
		8-32	**Spare** - reserved for future expansion

The following fields are repeated for each multitexture layer that is specified as present by the bits set in multitexmask. This mechanism allows for "sparse" multitexture layer information to be present and does not require that the information present be contiguous.

Field	Data Type	Length (bytes)	Description
$texture_N$	Unsigned Short	2	Texture index for texture layer N
$effect_N$	Unsigned Short	2	MultiTexture effect for layer N - can be any of the following values: 0 - Texture Environment 1 - Bump Map 2-100 - Reserved by MultiGen-Paradigm for future expansion >100 - User (runtime) defined
$mapping_N$	Unsigned Short	2	Texture mapping index for texture layer N.

Field	Data Type	Length (bytes)	Description
$data_N$	Unsigned Short	2	Texture data for layer N. This is user defined. For example, it may be used as a blend percentage or color or any other data needed by the runtime to describe texture layer N

UV List Record

The uv list record is an ancillary record of vertex nodes. This record (if present) always follows the vertex list or morph vertex list record and contains texture layer information for the vertices represented in the applicable vertex list record.

UV List Record Format

Data type	**Length** (bytes)	Description	
Unsigned Short	2	UV List opcode 53	
Unsigned Short	2	Length of the record	
Unsigned Int	4	32 bit mask indicating which multitexture uv values follow.	
		The bits appear from left to right and have the following definitions:	
		1	**Has Layer 1** - indicates that multitexture uv for texture layer 1 is present.
		2	**Has Layer 2** - indicates that multitexture uv for texture layer 2 is present.
		3	**Has Layer 3** -- indicates that multitexture uv for texture layer 3 is present.

4	**Has Layer 4** -- indicates that multitexture uv for texture layer 4 is present.
5	**Has Layer 5** - - indicates that multitexture uv for texture layer 5 is present.
6	**Has Layer 6** -- indicates that multitexture uv for texture layer 6 is present.
7	**Has Layer 7** - - indicates that multitexture uv for texture layer 7 is present.
8-32	**Spare** - reserved for future expansion

The following fields are repeated for each vertex contained in the corresponding vertex list or morph vertex list record.

- If this uv list record follows a vertex list record, the following fields are repeated for each layer present (as specified by the bits set in uvmask).

Field	Data Type	Length (bytes)	Description
$u_{i, N}$	Float	4	Texture coordinate U for vertex i, layer N
$v_{i, N}$	Float	4	Texture coordinate V for vertex i, layer N

- If this uv list record follows a morph vertex list record, the following fields are repeated for each layer present (as specified by the bits set in uvmask).

Field	Data Type	Length (bytes)	Description
$u0_{i, N}$	Float	4	Texture coordinate U for the 0% vertex i, layer N

Field	Data Type	Length (bytes)	Description
$v0_{i, N}$	Float	4	Texture coordinate V for the 0% vertex i, layer N
$u100_{i, N}$	Float	4	Texture coordinate U for the 100% vertex i, layer N
$v100_{i, N}$	Float	4	Texture coordinate V for the 100% vertex i, layer N

Example

Consider a triangular face (3 vertices) that contains morph vertex information and has texture layers 1 and 3 defined. The following example shows the contents of the uv list record corresponding to the morph vertex list record representing this triangle:

Field	Data Type	Length (bytes)	Description
.	.	.	.
opcode	Unsigned Short	2	53 (UV List opcode).
length	Unsigned Short	2	200 (Length of the record)
uvmask	Unsigned Int	4	1010 0000 0000 0000 (layers 1and 3 ON, others OFF)
u0 1,1	Float	8	Texture coordinate U for the 0% vertex 1, layer 1.
v0 1,1	Float	8	Texture coordinate V for the 0% vertex 1, layer 1.
u100 1,1	Float	8	Texture coordinate U for the 100% vertex 1, layer 1.

v100 1,1	Float	8	Texture coordinate V for the 100% vertex 1, layer 1.
u0 1,3	Float	8	Texture coordinate U for the 0% vertex 1, layer 3.
v0 1,3	Float	8	Texture coordinate V for the 0% vertex 1, layer 3.
u100 1,3	Float	8	Texture coordinate U for the 100% vertex 1, layer 3.
v100 1,3	Float	8	Texture coordinate V for the 100% vertex 1, layer 3.
u0 2,1	Float	8	Texture coordinate U for the 0% vertex 2, layer 1.
v0 2,1	Float	8	Texture coordinate V for the 0% vertex 2, layer 1.
u100 2,1	Float	8	Texture coordinate U for the 100% vertex 2, layer 1.
v100 2,1	Float	8	Texture coordinate V for the 100% vertex 2, layer 1.
u0 2,3	Float	8	Texture coordinate U for the 0% vertex 2, layer 3.
v0 2,3	Float	8	Texture coordinate V for the 0% vertex 2, layer 3.
u100 2,3	Float	8	Texture coordinate U for the 100% vertex 2, layer 3.
v100 2,3	Float	8	Texture coordinate V for the 100% vertex 2, layer 3.
u0 3,1	Float	8	Texture coordinate U for the 0% vertex 3, layer 1.
v0 3,1	Float	8	Texture coordinate V for the 0% vertex 3, layer 1.
u100 3,1	Float	8	Texture coordinate U for the 100% vertex 3, layer 1.

v100 3,1	Float	8	Texture coordinate V for the 100% vertex 3, layer 1.
u0 3,3	Float	8	Texture coordinate U for the 0% vertex 3, layer 3.
v0 3,3	Float	8	Texture coordinate V for the 0% vertex 3, layer 3.
u100 3,3	Float	8	Texture coordinate U for the 100% vertex 3, layer 3.
v100 3,3	Float	8	Texture coordinate V for the 100% vertex 3, layer 3

Replicate Record

A replicate record is an ancillary record of group, face, and light (string) point nodes. It indicates the number of times the group, face, or light (string) point is instantiated. An ancillary transformation record must also be present. The transformation is iteratively applied to each instance to uniquely place it in the database.

Replicate Record Format

Data type	**Length (**bytes**)**	Description
Int	2	Replicate Opcode 60
Unsigned Int	2	Length of the record
Int	2	Number of replications
Int	2	Reserved

Road Zone Record

The road zone record is an ancillary record of the header node. It references a road zone file that contains gridded elevation data. The format of the file is described in "Road Zone Files," page 111.

Road Zone Record Format

Data type	Length (bytes)	Description
Int	2	Road Path Opcode 88
Unsigned Int	2	Length of the record
Char	120	Zone file name
Int	4	Reserved
Double	8	Lower-left x coordinate
Double	8	Lower-left y coordinate
Double	8	Upper-right x coordinate
Double	8	Upper-right y coordinate
Double	8	Grid interval
Int	4	Number of posts along x axis
Int	4	Number of posts along y axis

Transformation Records

Transformation records may be ancillary records of most nodes. All hierarchical nodes may be transformed except the header node. Some nodes may only be transformed implicitly, as part of some other operation, such as point replication within a light point string.

There are several distinct types of transformation records, all of which follow a set pattern. For any transformation, a matrix record is always present and represents the final transformation. Following the matrix record is one or more other transformation records, including the General Matrix Record. The general matrix is present when a transformation of unknown composition has been applied.

Each record specifies a discrete transformation that has been concatenated into the final matrix. Concatenation is done in the order that the records are encountered, using premultiplication.

Note: The final and general matrices are only single-precision, while the discrete transformations are double-precision.

Matrix Record Format

Data type	Length (bytes)	Description
Int	2	Matrix Opcode 49
Unsigned Int	2	Length of the record
Float	4*16	4x4 single-precision matrix, row major order

Rotate About Edge Record Format

Data type	Length (bytes)	Description
Int	2	Rotate About Edge Opcode 76
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	x, first point on edge
Double	8	y, first point on edge
Double	8	z, first point on edge
Double	8	x, second point on edge
Double	8	y, second point on edge
Double	8	z, second point on edge
Float	4	Angle by which to rotate

Translate Record Format

Data type	Length (bytes)	Description
Int	2	Translate Opcode 78
Unsigned Int	2	Length of the record
Int	4	Reserved

Data type	Length (bytes)	Description
Double	8	x, reference FROM point
Double	8	y, reference FROM point
Double	8	z, reference FROM point
Double	8	Delta x to translate node by
Double	8	Delta y to translate node by
Double	8	Delta z to translate node by

Scale (Nonuniform) Record Format

Data type	Length (bytes)	Description
Int	2	Scale Opcode 79
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	Center x
Double	8	Center y
Double	8	Center z
Float	4	x scale factor
Float	4	y scale factor
Float	4	z scale factor

Rotate About Point Record Format

Data type	Length (bytes)	Description
Int	2	Rotate About Point Opcode 80
Unsigned Int	2	Length of the record

Data type	Length (bytes)	Description
Int	4	Reserved
Double	8	x, center of rotation
Double	8	y, center of rotation
Double	8	z, center of rotation
Float	4	i, axis of rotation
Float	4	j, axis of rotation
Float	4	k, axis of rotation
Float	4	Angle by which to rotate

Rotate and/or Scale Record Format

Data type	Length (bytes)	Description
Int	2	Rotate and/or Scale Opcode 81
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	x, center of scale
Double	8	y, center of scale
Double	8	z, center of scale
Double	8	x, reference point
Double	8	y, reference point
Double	8	z, reference point
Double	8	x, TO point
Double	8	y, TO point
Double	8	z, TO point

Data type	Length (bytes)	Description
Float	4	Overall scale factor
Float	4	Scale factor in direction of axis
Float	4	Angle by which to rotate

Put Record Format

Data type	Length (bytes)	Description
Int	2	Put Opcode 82
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	x, FROM origin
Double	8	y, FROM origin
Double	8	z, FROM origin
Double	8	x, FROM align
Double	8	y, FROM align
Double	8	z, FROM align
Double	8	x, FROM track
Double	8	y, FROM track
Double	8	z, FROM track
Double	8	x, TO origin
Double	8	y, TO origin
Double	8	z, TO origin
Double	8	x, TO align
Double	8	y, TO align

Data type	Length (bytes)	Description
Double	8	z, TO align
Double	8	x, TO track
Double	8	y, TO track
Double	8	z, TO track

General Matrix Record Format

Data type	Length (bytes)	Description
Int	2	General Matrix Opcode 94
Unsigned Int	2	Length of the record
Float	4*16	4x4 single-precision matrix, row major order

Vector Record

A vector record is an ancillary record of the (pre V15.4) face node. Its only use is to provide the direction vector for old-style unidirectional and bidirectional light point faces.

Vector Record Format

Data type	Length (bytes)	Description
Int	2	Vector Opcode 50
Unsigned Int	2	Length of the record
Float	4	i component
Float	4	j component

Data type	Length (bytes)	Description
Float	4	k component

Bounding Volume Records

Bounding volumes are ancillary records for group nodes only. They generally encompass all the geometry of a group's children. A bounding volume may describe a box, sphere, or cylinder.

The center coordinate of a bounding volume is stored as a separate record. The orientation of a bounding volume is also stored as a separate record.

Applications may use the bounding volume information with culling and collision detection algorithms.

Bounding Box Record Format

Data type	Length (bytes)	Description
Int	2	Bounding Box Opcode 74
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	x coordinate of lowest corner
Double	8	y coordinate of lowest corner
Double	8	z coordinate of lowest corner
Double	8	x coordinate of highest corner
Double	8	y coordinate of highest corner
Double	8	z coordinate of highest corner

Bounding Sphere Format

Data type	Length (bytes)	Description
Int	2	Bounding Sphere Opcode 105
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	Radius of the sphere

Bounding Cylinder Format

Data type	Length (bytes)	Description
Int	2	Bounding Cylinder Opcode 106
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	Radius of the cylinder base
Double	8	Height of the cylinder

Bounding Volume Center Format

Data type	Length (bytes)	Description
Int	2	Bounding Volume Center Opcode 108

Data type	Length (bytes)	Description
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	x coordinate of center
Double	8	y coordinate of center
Double	8	z coordinate of center

Bounding Volume Orientation

Data type	Length (bytes)	Description
Int	2	Bounding Volume Orientation Opcode 109
Unsigned Int	2	Length of the record
Int	4	Reserved
Double	8	Yaw angle
Double	8	Pitch angle
Double	8	Roll angle

CAT Data Records

The CAT Data records contain the information needed to reconstruct a Continuously Adaptive Terrain skin from its OpenFlight representation. They provide the information which links faces between levels of detail within a CAT skin. CAT Data is stored as a key table with an opcode of 116. For specific detail please refer to “Key Table Records” on page 96.

Each CAT Data record describes how a face within a CAT skin is related to faces at the next finer level of detail. The coarsest level of detail is level zero. The next finer level of detail is one, and so forth. Each data record is stored in the key table using an ordinal key, counting up from zero. The face node ID is stored in the data record, thereby providing the cross reference to the OpenFlight face node that represents it.

In OpenFlight, each CAT level of detail is parented by a LOD node. Each CAT triangle strip is parented by a group node.

CAT Data Header Record Format

Data type	Length (bytes)	Description
Int	2	CAT Data Opcode 116
Unsigned Int	2	Length of the record
Int	4	Subtype = 1
Int	4	Max number of face keys
Int	4	Actual number of face keys
Int	4	Total length of packed face data
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

CAT Data Key Record Format

Data type	Length (bytes)	Description
Int	4	Face index
Int	4	Reserved
Int	4	Face data record offset from start of packed data

CAT Data Face Record Format

Data type	Length (bytes)	Description
Int	4	Level of detail to which this face belongs.
Int	4	First child index (within this table) The 1st child of this face, or -1 for Nil.
Int	4	Second child index (within this table) The 2nd child of this face, or -1 for Nil.
Int	4	Third child index (within this table) The 3rd child of this face, or -1 for Nil.
Int	4	Fourth child index (within this table) The 4th child of this face, or -1 for Nil.
Int	4	Length of face node ID string which follows
Char	variable	ASCII ID of the face node to which this record applies.

Extension Attribute Record

The extension attribute record is an ancillary record defined by an extension site that utilizes the extensibility of the OpenFlight format. It specifies the site information of a third party extended record which describes additional data that is not represented by the standard OpenFlight records. The data itself is transparent to users other than the extension site. The data can be accessed by the combination of the OpenFlight API and the data dictionary defined by the extension site.

Any hierarchical node can contain extension attribute records. Extension attributes are introduced by an push extension control record and concluded by a pop extension control record.

The extension record (Opcode 100) may also introduce a new node type (See “Extension Node Record” on page 63.)

Extension Attribute Record Format

Data type	Length (bytes)	Description
Int	2	Extension Opcode 100
Unsigned Int	2	Length of the total extension record
Char	8	7 char ASCII ID; 0 terminates
Char	8	Site ID - Unique site name
Int	1	Reserved
Int	1	Revision - site specific
Unsigned Int	2	Record code - site specific
Char	Variable	Extended data - site specific

Palette Records

Palette records are ancillary records of the header node. They contain attribute data globally shared by other nodes in the database. Other nodes, such as face nodes, reference the palette data by index.

Individual palettes contain resources such as vertex, material, light source, texture pattern, and line style definitions.

Vertex Palette Records

Double precision vertex records are stored in a vertex palette for the entire database.

The vertex palette header record signifies the start of the vertex palette. It contains a one word entry specifying the total length of the vertex palette, which is equal to the length of this header record plus the length of the following vertex records. The individual vertex records follow this header, each starting with its own opcode. The length field in the vertex palette header record makes it possible to skip over vertex records until the data is actually needed.

Vertices may be shared, and are accessed through the vertex and morph vertex list records following each face record. A face may contain all morph vertices, all non-morph vertices, or a mixture of both. Thus there can be one or more list records following each face. Consecutive vertices with the same type are grouped together within a list record. The length of each list record is determined by the number of consecutive vertices of each type. For each vertex, there is a one word field pointing to its vertex record in the vertex palette. Since this offset includes the length of the vertex header record, the value of the first pointer is 8.

Vertex Palette Header Record Format

Data type	Length (bytes)	Description
Int	2	Vertex Palette Opcode 67
Unsigned Int	2	Length of the record
Int	4	Length of this record plus length of the vertex palette

The vertex palette header record is immediately followed by vertex records. Each vertex record contains all the attributes of a vertex that has been referenced one or more times in the database.

The Color name index references a name in the color name palette.

The Hard edge flag indicates this vertex starts an edge that is to be preserved by polygon reduction or decimation algorithms.

The Normal frozen flag indicates the normal is not to be updated by shading or lighting algorithms.

The No color flag indicates the vertex does not have a color.

The Packed color flag indicates the packed color is used instead of the color index.

Vertex with Color Record Format

Data type	Length (bytes)	Description
Int	2	Vertex Coordinate Opcode 68
Unsigned Int	2	Length of the record
Unsigned Int	2	Color name index
Boolean	2	Flags (bits, from left to right) 0 = Start hard edge 1 = Normal frozen 2 = No color 3 = Packed color 4-15 = Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Int	4	Packed color (A, B, G, R)
Unsigned Int	4	Vertex color index

Vertex with Normal Record Format

Data type	Length (bytes)	Description
Int	2	Vertex with Normal Opcode 69
Unsigned Int	2	Length of the record

Data type	Length (bytes)	Description
Unsigned Int	2	Color name index
Boolean	2	Flags (bits, from left to right)
		0 = Start hard edge
		1 = Normal frozen
		2 = No color
		3 = Packed color
		4-15 = Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	4*3	Vertex normal (i, j, k)
Int	4	Packed color (A, B, G, R)
Unsigned Int	4	Vertex color index

Vertex with Texture Record Format

Data type	Length (bytes)	Description
Int	2	Vertex with UV Opcode 71
Unsigned Int	2	Length of the record
Unsigned Int	2	Color name index
Boolean	2	Flags (bits, from left to right)

Data type	Length (bytes)	Description
		0 = Start hard edge
		1 = Normal frozen
		2 = No color
		3 = Packed color
		4-15 = Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	4*2	Texture (u, v)
Int	4	Packed color (A, B, G, R)
Unsigned Int	4	Vertex color index

Vertex Record with Normal and Texture Format

Data type	Length (bytes)	Description
Int	2	Vertex with Normal and UV Opcode 70
Unsigned Int	2	Length of the record
Unsigned Int	2	Color name index
Boolean	2	Flags (bits, from left to right)
		0 = Start hard edge
		1 = Normal frozen

Data type	Length (bytes)	Description
		2 = No color
		3 = Packed color
		4-15 = Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	4*3	Vertex normal (i, j, k)
Float	4*2	Texture (u, v)
Int	4	Packed color (A, B, G, R)
Unsigned Int	4	Vertex color index

Color Palette Record

The color palette record contains all colors indexed by face and vertex nodes in the database.

The color record is divided into two sections: one for color entries and one for color names. All color entries are in 32-bit packed format (A, B, G, R). Each color consists of red, green, and blue components of 8 bits each, plus 8 bits reserved for alpha (future). The color entry section consists of 1024 ramped colors of 128 intensities each.

The color name section consists of a header followed by 0 or more color name entries. The header contains the number of names in the palette. If this value is 0, there are no names in the palette. Each color name entry contains the name string, pointer to the associated color entry, and other reserved information. The name field is a variable-length, null-terminated ASCII string, with a maximum of 80 bytes.

Color Palette Record Format

Data type	Length (bytes)	Description
Int	2	Color Palette Opcode 32

Data type	Length (bytes)	Description
Unsigned Int	2	Length of the record
Char	128	Reserved
Int	4	Brightest RGB of color 0, intensity 127
Int	4	Brightest RGB of color 1, intensity 127
etc.	...	
Int	4	Brightest RGB of color 1023
Int	4	Number of color names
Unsigned Int	2	Length of first color name entry
Int	2	Reserved
Int	2	Color entry index
Int	2	Reserved
Char	-	Color name string (variable length, up to 80 bytes)
Unsigned Int	2	Length of second color name entry
Int	2	Reserved
Int	2	Color entry index
Int	2	Reserved
Char	-	Color name string (variable length, up to 80 bytes)
etc.	...	
Unsigned Int	2	Length of last color name entry
Int	2	Reserved
Int	2	Color entry index
Int	2	Reserved
Char	-	Color name string (variable length, up to 80 bytes)

Name Table Record

The name table contains a lookup table of names referenced within the database. These names are typically used as attributes (e.g., color name index in the polygon record). The primary benefit of the name table is to allow name referencing, so each name string is only stored once. Each name entry in the name table contains fields for entry length, name index, and name string. The name index is used by the database to reference names within the table. The name field is a variable-length, null-terminated ASCII string, with a maximum of 80 bytes.

Name Table Record Format

Data type	Length (bytes)	Description
Int	2	Text Comment Opcode 114
Unsigned Int	2	Length of the record
Int	4	Number of names
Unsigned Int	2	Next available name index
Int	4	Length of first name entry
Unsigned Int	2	Name index
Char	-	Name string (variable length, up to 80 bytes)
Int	4	Length of second name entry
Unsigned Int	2	Name index
Char	-	Name string (variable length, up to 80 bytes)
etc.	...	
Int	4	Length of last name entry
Unsigned Int	2	Name identifier
Char	-	Name string (variable length, up to 80 bytes)

Material Record

The material palette contains descriptions of materials used while drawing geometry. It is composed of an arbitrary number of material records. The material records must follow the header record and precede the first push.

The material palette is not written with the database unless a face has been assigned a non-negative material code.

The appearance of a face in OpenFlight is a combination of the face color and the material properties. The face color is factored into the material properties as follows:

Ambient:

The displayed material's ambient component is the product of the ambient component of the material and the face color:

Displayed ambient (red) = Material ambient (red)* face color (red)
Displayed ambient (green) = Material ambient (green)* face color (green)
Displayed ambient (blue) = Material ambient (blue)* face color (blue)

For example, suppose the material has an ambient component of {1.0,.5,.5} and the face color is {100, 100, 100}. The displayed material has as its ambient color {100, 50, 50}.

Diffuse:

As with the ambient component, the diffuse component is the product of the diffuse component of the material and the face color:

Displayed diffuse (red) = Material diffuse (red)* face color (red)
Displayed diffuse (green) = Material diffuse (green)* face color (green)
Displayed diffuse (blue) = Material diffuse (blue)* face color (blue)

Specular:

Unlike ambient and diffuse components, the displayed specular component is taken directly from the material:

Displayed specular (red) = Material specular (red)
Displayed specular (green) = Material specular (green)
Displayed specular (blue) = Material specular (blue)

Emissive:

The displayed emissive component is taken directly from the material:

Displayed emissive (red) = Material emissive (red)
Displayed emissive (green) = Material emissive (green)
Displayed emissive (blue) = Material emissive (blue)

Shininess:

MultiGen drawing takes the shininess directly from the material. Specular highlights are tighter, with higher shininess values.

Alpha:

An alpha of 1.0 is fully opaque, while 0.0 is fully transparent. When drawing faces, MultiGen combines the transparency value of the face record with the alpha value of the material record.

The final alpha applied to a face is a floating point number between 0.0 (transparent) and 1.0 (opaque), and is computed as follows:

Final alpha = material alpha * (1.0 - ((face transparency * object transparency) / 65535))

Material Record Format

Data type	Length (bytes)	Description
Int	2	Material Record Opcode 113
Int	2	Length of the record
Int	4	Material index
Char	12	Material name
Boolean	4	Flags
		0 = Materials used
		1-31 = Spare
Float	4	Ambient red component of material 0.*
Float	4	Ambient green component of material 0.*
Float	4	Ambient blue component of material 0.*
Float	4	Diffuse red component of material 0*.
Float	4	Diffuse green component of material 0*.
Float	4	Diffuse blue component of material 0.*
Float	4	Specular red component of material 0.*
Float	4	Specular green component of material 0.*
Float	4	Specular blue component of material 0.*
Float	4	Emissive red component of material 0.*

Data type	Length (bytes)	Description
Float	4	Emissive green component of material 0.*
Float	4	Emissive blue component of material 0.*
Float	4	Shininess. (Single-precision float in the range [0.0-128.0])
Float	4	Alpha. (Single-precision float in the range [0.0-1.0], where 1.0 is opaque)
Int	4	Spare

* normalized values between 0.0 and 1.0, inclusive.

Texture Record

There is one record for each texture pattern referenced in the database. These records must follow the header record and precede the first push.

A palette and pattern system can be used to reference the texture patterns. A texture palette is made up of 256 patterns. The pattern index for the first palette is 0 - 255, for the second palette 256 - 511, etc. Note: If less than 256 patterns exist on a palette, several pattern indices are unused. The x and y palette locations are used to store offset locations in the palette for display.

Texture Pattern File Reference Format

Data type	Length (bytes)h	Description
Int	2	Texture Reference Record Opcode 64
Unsigned Int	2	Length of the record
Char	200	File name of texture pattern
Int	4	Pattern index

Data type	Length (bytes)h	Description
Int	4	x location in texture palette
Int	4	y location in texture palette

Eyepoint and Trackplane Palette Record

Eyepoint Position Format

Eyepoint/ Trackplane	Data type	Length (bytes)	Description
	Int	2	Eyepoint and Trackplane Position Opcode 83
	Unsigned Int	2	Length of the record
	Int	4	Reserved
Last Eyepoint 0	Double	8*3	x, y, z of rotation center
	Float	4*3	Yaw, pitch, and roll angles
	Float	4*16	4x4 single-precision rotation matrix
	Float	4	Field of view
	Float	4	Scale
	Float	4*2	Near and far clipping plane
	Float	4*16	4x4 single-precision fly-through matrix

Eyepoint/ Trackplane	Data type	Length (bytes)	Description
	Float	4*3	x, y, z of eyepoint in database
	Float	4	Yaw of fly-through
	Float	4	Pitch of fly-through
	Float	4*3	i, j, k vector for eyepoint direction
	Int	4	Flag (True if no fly-through)
	Int	4	Flag (True if ortho drawing mode)
	Int	4	Flag (True if this is a valid eyepoint)
	Int	4	Image offset x
	Int	4	Image offset y
	Int	4	Image zoom
	Int	4*8	Spare
	Int	4	Reserved
Eyepoint 1	Same as last eyepoint		
Eyepoint 2	Same as last eyepoint		
Eyepoint 3	Same as last eyepoint		
Eyepoint 4	Same as last eyepoint		

Eyepoint/ Trackplane	Data type	Length (bytes)	Description
Eyepoint 5	Same as last eye-point		
Eyepoint 6	Same as last eye-point		
Eyepoint 7	Same as last eye-point		
Eyepoint 8	Same as last eye-point		
Eyepoint 9	Same as last eye-point		
Trackplane 0	Int	4	Active flag
	Int	4	Spare
	Double	8*3	Trackplane origin coordinate
	Double	8*3	Trackplane alignment coordinate
	Double	8*3	Trackplane plane coordinate
	Boolean	1	TRUE if grid is visible
	Int	1	Grid type flag:
			0 - rectangular grid
			1 - radial grid
	Int	1	Grid under flag:
			0 - draw grid over scene
			1 - draw grid under scene

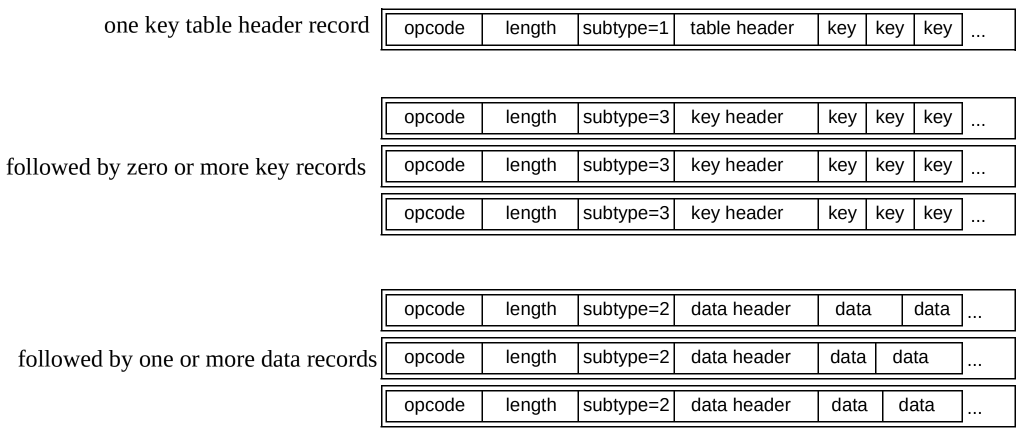
Eyepoint/ Trackplane	Data type	Length (bytes)		Description
				2 - draw grid depth buffered
	Int	1	Reserved	
	Float		4	Grid angle for ra-dial grid
	Double		8	Grid spacing in X. Radius if radi-al grid.
	Double		8	Grid spacing in Y
	Int	1	Radial grid spacing di-rection control	
	Int	1	Rectangular grid spac-ing direction control	
	Boolean	1	If TRUE, snap cursor to grid	
	Int	1	Reserved	
	Int	4	Reserved	
	Double		8	Grid size (a pow-er of 2)
	Boolean	4	Mask of visible grid quadrants	
	Int		4	Reserved
Trackplane 1	Same as last track plane			
	Same as last track plane			
through	Same as last track plane			
	Same as last track plane			

Eyepoint/ Trackplane	Data type	Length (bytes)	Description
Trackplane 9	Same as last track plane		

Key Table Records

Key table records store variable length data records and their identifiers. The linkage editor, sound palette, and CAT Data are stored as key table records. The first key table record contains the key table header and a set of keys. If all the keys cannot fit into the first record, additional key records are written. This is followed by one or more key table data records.

A Key Table consists of:


one key table header record
opcode | length | subtype=1 | table header | key | key | key ...
followed by zero or more key records
opcode | length | subtype=3 | key header | key | key | key ...
opcode | length | subtype=3 | key header | key | key | key ...
opcode | length | subtype=3 | key header | key | key | key ...
followed by one or more data records
opcode | length | subtype=2 | data header | data | data ...
opcode | length | subtype=2 | data header | data | data ...
opcode | length | subtype=2 | data header | data | data ...

For an example of the use of key table records, refer to the discussion of sound below.

Key Table Header Format

Data type	Length (bytes)	Description
Int	2	X (Opcode of record using key table for storage)

Unsigned Int	2	Length of the record
Int	4	Subtype = 1
Int	4	Max number of entries
Int	4	Number of entries
Int	4	Total length of packed data
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Key Format

Data type	Length (bytes)	Description
Int	4	Key value
Int	4	Data type
Int	4	Offset from start of packed data The offset is calculated from the start of the packed data in the data record. The length of the header information for all data records is ignored.

Key Header Format

Data type	Length (bytes)	Description
Int	2	X (Opcode of record using key table for storage)
Unsigned Int	2	Length of the record

Int	4	Subtype = 3
Int	4	Data length

Key Table Data Format

Data type	Length (bytes)	Description
Int	2	X (Opcode of record using key table for storage)
Unsigned Int	2	Length of the record
Int	4	Subtype = 2
Int	4	Data length
Char	Data length	Packed data (data is always 4 byte aligned, with unused bytes set to null)

Linkage Palette Records

Database linkages use key table records. Linkage data consists of two different constructs: nodes and arcs. Nodes usually contain data pertaining to database entities such as DOFs. In addition, the nodes may represent modeling driver functions and code nodes. The arcs contain information on how all the nodes are connected to each other. For most nodes, the value of the node is contained in the following Name Subrecord. For example, this node value can be a node name, when the node represents a database entity, or a math formula as a string, in the case of a formula node. Names are stored as null-terminated ASCII strings.

Linkage Palette Header Format

Data type	Length (bytes)	Description
Int	2	Linkage Palette Opcode 90
Unsigned Int	2	Length of the record
Int	4	Subtype = 1 (indicating this is a header, rather than data record)

Int	4	Max number of nodes, arcs, and entity references
Int	4	Number of nodes, arcs, and entity references
Int	4	Total length of data
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Immediately followed by a series of key subrecords, as below.

Key Subrecords Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Data type 0x12120001 = Node data 0x12120002 = Arc data 0x12120004 = Database entity name
Int	4	Offset from start of packed data field in linkage data record

Key subrecords repeat for all types (nodes, arcs, and entity references).

Linkage Palette Data Record Format

Data type	Length (bytes)	Description
Int	2	Linkage Palette Opcode 90
Unsigned Int	2	Length of the record
Int	4	Subtype = 2 (indicating this is a data, rather than header, record)
Int	4	Data length

Char	Data length	Packed data (in the format of node data subrecords, arc data subrecords, and entity name subrecords, as described below)

General Node Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Node type
		0x12120003 = Header node
		0x12120005 = Database entity node
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier

Formula Node Data Subrecord Format

Data type	Length (bytes)	Description

Int	4	Identifier
Int	4	Reserved
Int	4	Data type = 0x12150000
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Driver Node Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Node type 0x12140001 = Ramp driver node 0x12140004 = Variable driver node 0x12140005 = External file driver node
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier
Float	4	Current value
Float	4	Min amplitude
Float	4	Max amplitude
Float	4	Wave offset
Float	4	Min time
Float	4	Max time
Float	4	Time steps
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Arc Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Data type = 0x12120002
Int	4	Reserved
Int	4	Reserved
Int	4	Priority

Int	4	Source parameter (parameter ID if source node is a node)
Int	4	Sink parameter (parameter ID if sink node is a node; number (0...7) for variables (x1...x8), if sink node is a formula)
Int	4	Reserved
Int	4	Next source identifier
Int	4	Next sink identifier
Int	4	Node source identifier
Int	4	Node sink identifier

See “Linkage Editor Parameter IDs” on page 113 for parameter ID values and descriptions.

Database Entity Name Subrecord Format

Data type	Length (bytes)	Description
Char	Variable	Null-terminated ASCII string

Sound Palette Records

The sound palette uses key table records to store the sound index and file name. The index is the key value, and the file name is the data record, formatted as a null-terminated ASCII string. The sound palette header record indicates the number of sounds associated with the database.

Sound Palette Header Record Format

Data type	Length (bytes)	Description
Int	2	Sound Palette Opcode 93

Unsigned Int	2	Length of the record
Int	4	Subtype = 1 (indicating this is header rather than palette record)
Int	4	Max number of sounds
Int	4	Actual number of sounds in palette
Int	4	Total length of sound file names
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Followed by a series of sound key subrecords:

Sound Key Subrecord Format

Data type	Length (bytes)	Description
Int	4	Sound index
Int	4	Reserved
Int	4	Data record offset from start of packed file names (in sound palette data record)

Key records repeat for number of sounds.

Sound Palette Data Record Format

Data type	Length (bytes)	Description
Int	2	Sound Palette Opcode 93
Unsigned Int	2	Length of the record
Int	4	Subtype = 2 (indicating this is a palette rather than a sound record)

Int	4	File names' length
Char	Data length	Packed file names

Light Source Palette Record

Each of these records represents a new entry in the light source palette. Entries may be referenced by light source nodes using the palette index. Lights can be flagged as modeling lights, which illuminate a scene without being stored as part of the hierarchy. A modeling light is always positioned at the eye; its direction is stored in the palette. A light referenced by a node obtains its position and direction from the node. In this case, the palette yaw and pitch components are ignored.

Light Source Palette Element Record Format

Data type	Length (bytes)	Description
Int	2	Light Source Palette Opcode 102
Unsigned Int	2	Length of the record
Int	4	Palette index
Int	2*4	Reserved
Char	20	Light source name
Int	4	Reserved
Float	4*4	Ambient RGBA (alpha component is currently unused)
Float	4*4	Diffuse RGBA (alpha component is currently unused)
Float	4*4	Specular RGBA (alpha component is currently unused)
Int	4	Light type 0 = Infinite 1 = Local 2 = Spot
Int	4*10	Reserved
Float	4	Spot exponential dropoff term
Float	4	Spot cutoff angle (in degrees)

Float	4	Yaw
Float	4	Pitch
Float	4	Constant attenuation coefficient
Float	4	Linear attenuation coefficient
Float	4	Quadratic attenuation coefficient
Boolean	4	Modeling light (TRUE/FALSE)
Int	4*19	Spare

Line Style Palette Record

Line style records define the outline displayed around faces in wireframe or wireframe-over-solid mode. The Pattern field defines a mask to control the display of segments of the line. For example, if all the bits of the mask are set, the line is drawn as a solid line. If every other bit is on, the line is displayed as a dashed line. The Line Width field controls the width of the line in pixels. Line style 0 is the default. Faces are assigned line styles in the Line Style field of the face record. One of these records appears for each line style defined in the OpenFlight file.

Line Style Record

Data type	Length (bytes)	Description
Int	2	Line Style Record Opcode 97
Int	2	Length of record
Int	2	Line style index
Int	2	Pattern mask
Int	4	Line width

Texture Mapping Record

The texture mapping palette record defines methods and parameters used to map textures onto geometry. One record is created for each texture mapping reference in the palette. These records must follow the header record and precede the first push.

Texture Mapping Record

Data type	Length (bytes)	Description
Int	2	Texture Mapping Palette Opcode 112
Int	2	Length of record
Int	4	Reserved
Int	4	Texture mapping index
Char	20	Texture mapping name
Int	4	Texture mapping type 0 = None 1 = Put 2 = 4 Point Put 3 = Reserved 4 = Spherical Project 5 = Radial Project 6 = Reserved
Int	4	Warped flag; if TRUE, 8 point warp applied
Double	8*16	Transformation matrix (valid only for Types 1 & 2)
Variable	Variable	Parameters (see below for parameters for each mapping type)

Parameters for Put Texture Mapping (Type 1)

Data type	Length (bytes)	Description
Int	4	State of Put Texture tool 0 = Start state - no points entered 1 = One point entered

		2 = Two points entered
		3 = Three points entered
Int	4	Active geometry point
		1 = Origin point
		2 = Alignment point
		3 = Shear point
Double	8*3	x, y, z of lower-left corner of bounding box of geometry using this mapping
Double	8*3	x, y, z of upper-right corner of bounding box of geometry using this mapping
Int	4*3	Use real world size flags for each of the three put points
Double	8*3	x, y, z of the texture origin point
Double	8*3	x, y, z of the texture alignment point
Double	8*3	x, y, z of the texture shear point
Double	8*3	x, y, z of the geometry origin point
Double	8*3	x, y, z of the geometry alignment point
Double	8*3	x, y, z of the geometry shear point
Int	4	Active texture point
		1 = Origin point
		2 = Alignment point
		3 = Shear point
Int	4	Reserved; should always be set to 1
Variable	Variable	Parameters (see parameters below for each mapping type)

Parameters for 4 Point Put Texture Mapping (Type 2)

Data type	Length (bytes)	Description
Int	4	State of Put Texture tool 0 = Start state - no points entered 1 = One point entered 2 = Two points entered 3 = Three points entered 4 = Four points entered
Int	4	Active geometry point 1 = Origin point 2 = Alignment point 3 = Shear point 4 = Perspective point
Double	8*3	x, y, z of lower-left corner of bounding box of geometry using this mapping
Double	8*3	x, y, z of upper-right corner of bounding box of geometry using this mapping
Int	3*4	Use real world size flags for each of the three put points
Double	8*3	x, y, z of the texture origin point
Double	8*3	x, y, z of the texture alignment point
Double	8*3	x, y, z of the texture shear point
Double	8*3	x, y, z of the texture perspective point
Double	8*3	x, y, z of the geometry origin point
Double	8*3	x, y, z of the geometry alignment point
Double	8*3	x, y, z of the geometry shear point
Double	8*3	x, y, z of the geometry perspective point
Int	4	Active texture point 1 = Origin point

		2 = Alignment point
		3 = Shear point
		4 = Perspective point
Int	4	Reserved; should always be set to 1
Float	4	Depth scale factor
Double	8*16	Transformation matrix for the 4 point projection plane

Parameters for Spherical Project Mapping (Type 4)

Data type	Length (bytes)	Description
Float	4	Scale
Double	8*3	x, y, z of the center of the projection sphere
Float	4	Scale / (maximum dimension of the mapped geometry bounding box)
Float	4	Maximum dimension of the mapped geometry bounding box

Parameters for Radial Project Mapping (Type 5)

Data type	Length (bytes)	Description
Int	4	Active geometry point
		1 = End point 1 of cylinder center line
		2 = End point 2 of cylinder center line
Int	4	Reserved

Float	4	Radial scale
Float	4	Scale along length of cylinder
Double	8*16	Trackplane to XY plane transformation matrix
Double	8*3	x, y, z of end point 1 of cylinder center line
Double	8*3	x, y, z of end point 2 of cylinder center line

Parameters for Warped Mapping (Warped Flag Set)

Data type	Length (bytes)	Description
Int	4	Active geometry point 0 = First warp FROM point 1 = Second warp FROM point 2 = Third warp FROM point 3 = Fourth warp FROM point 4 = Fifth warp FROM point 5 = Sixth warp FROM point 6 = Seventh warp FROM point 7 = Eighth warp FROM point 8 = First warp TO point 9 = Second warp TO point 10 = Third warp TO point 11 = Fourth warp TO point 12 = Fifth warp TO point 13 = Sixth warp TO point 14 = Seventh warp TO point 15 = Eighth warp TO point

Int	4	Warp tool state 0 = Start state - no points entered 1 = One FROM point entered 2 = Two FROM point entered 3 = Three FROM point entered 4 = Four FROM point entered 5 = Five FROM point entered 6 = Six FROM point entered 7 = Seven FROM point entered 8 = All FROM point entered
Double	8*16	Trackplane to XY plane transformation matrix
Double	16*2	x, y of the first FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the second FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the third FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the fourth FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the fifth FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the sixth FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the seventh FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the eighth FROM point transformed to the XY plane by the above matrix
Double	16*2	x, y of the first TO point transformed to the

		XY plane by the above matrix
Double	16*2	x, y of the second TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the third TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the fourth TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the fifth TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the sixth TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the seventh TO point transformed to the XY plane by the above matrix
Double	16*2	x, y of the eighth TO point transformed to the XY plane by the above matrix

Continuation Record

All OpenFlight records begin with a 4 byte sequence. The first two bytes identify the record (opcode) and the second two bytes specify the length of the record. Given this regular record structure, the length of all OpenFlight records is limited to the largest value that can be encoded with 2 bytes or 16 bits (65535). In most cases, this maximum size is sufficient but there are cases where it is not. For fixed size records, this is not a problem. For variable size records, this limitation is being addressed with this version.

A new record, called the continuation record is introduced in this version to accommodate variable size records in the OpenFlight Scene Description. The continuation record is used to "continue" a record in the OpenFlight Scene Description file stream. It would appear in the stream immediately following the record that it "continues" (the record that is being continued will be referred to as the "original" record). The data contained in the continuation record is defined by the original record and is assumed to be directly appended onto the content of the original record.

Note: Multiple continuation records may follow a record, in which case all continuation records would be appended (in sequence) to the original record.

Continuation Record Format

Data type	Length (bytes)	Description
Unsigned Short	2	Continuation opcode 23
Unsigned Short	2	Length of the record
Varies	length-4	Depends on the original record. The contents of this field are to be appended directly to the end of the original record contents, before the original record contents are parsed.

2 Texture Files

Texture Pattern Files

OpenFlight does not have its own texture pattern format, but rather uses existing texture formats and references patterns by file name (See “Color Palette Record” on page 74.). File formats currently supported include:

- AT&T® image 8 format (8-bit color lookup)
- AT&T image 8 template format
- SGI intensity modulation
- SGI intensity modulation with alpha
- SGI RGB
- SGI RGB with alpha
- GIF
- JPEG/JFIF
- TIFF
- IFF/ILBM
- BMP/DIB
- PCX
- Targa™
- Alias™ Pix
- IRIS Performer™ clip texture

The format of the file is determined by the file name extension, the magic numbers within the file, or the texture attribute file, as described below.

Texture Attribute Files

A corresponding attribute file is created for each texture pattern, with the name of the attribute file the same as the texture file, followed by the extension “.attr”. These attribute files are used by the modeling software, and may not be necessary for the application using the database. They are in the following format:

Texture Attribute File Format

Data type	Length (bytes)	Description
Int	4	Number of texels in u direction
Int	4	Number of texels in v direction
Int	4	Real world size u direction
Int	4	Real world size v direction
Int	4	x component of up vector
Int	4	y component of up vector
Int	4	File format type -1 Not used 0 AT&T image 8 pattern 1 AT&T image 8 template 2 SGI intensity modulation 3 SGI intensity w/ alpha 4 SGI RGB 5 SGI RGB w/ alpha
Int	4	Minification filter type 0 - TX_POINT 1 - TX_BILINEAR 2 - TX_MIPMAP (Obsolete) 3 - TX_MIPMAP_POINT 4 - TX_MIPMAP_LINEAR 5 - TX_MIPMAP_BILINEAR 6 - TX_MIPMAP_TRILINEAR

		7 -	None
		8 -	TX_BICUBIC
		9 -	TX_BILINEAR_GEQUAL
		10 -	TX_BILINEAR_LEQUAL
		11 -	TX_BICUBIC_GEQUAL
		12 -	TX_BICUBIC_LEQUAL
Int	4	Magnification filter type	
		0 -	TX_POINT
		1 -	TX_BILINEAR
		2 -	None
		3 -	TX_BICUBIC
		4 -	TX_SHARPEN
		5 -	TX_ADD_DETAIL
		6 -	TX_MODULATE_DETAIL
		7 -	TX_BILINEAR_GEQUAL
		8 -	TX_BILINEAR_LEQUAL
		9 -	TX_BICUBIC_GEQUAL
		10 -	TX_BICUBIC_LEQUAL
Int	4	Repetition type	
		0 -	TX_REPEAT
		1 -	TX_CLAMP
		2 -	(Obsolete)
Int	4	Repetition type in u direction (see above)	
Int	4	Repetition type in v direction (see above)	
Int	4	Modify flag (for internal use)	
Int	4	x pivot point for rotating textures	
Int	4	y pivot point for rotating textures	
Int	4	Environment type	
		0 - TV_MODULATE	

		1 - TV_BLEND
		2 - TV_DECAL
		3 - TV_COLOR
Int	4	TRUE if intensity pattern to be loaded in alpha with white in color
Int	4*8	8 words of spare
Double	8	Real world size u for floating point databases
Double	8	Real world size v for floating point databases
Int	4	Code for origin of imported texture
Int	4	Kernel version number
Int	4	Internal format type
		0 - Default
		1 - TX_I_12A_4
		2 - TX_IA_8
		3 - TX_RGB_5
		4 - TX_RGBA_4
		5 - TX_IA_12
		6 - TX_RGBA_8
		7 - TX_RGBA_12
		8 - TX_I_16 (shadow mode only)
		9 - TX_RGB_12
Int	4	External format type
		0 - Default
		1 - TX_PACK_8
		2 - TX_PACK_16
Int	4	Boolean TRUE if using following 8 floats for MIPMAP kernel
Float	4*8	8 floats for kernel of separable symmetric filter
Int	4	Boolean if TRUE send:
Float	4	LOD0 for TX_CONTROL_POINT

Float	4	SCALE0 for TX_CONTROL_POINT
Float	4	LOD1 for TX_CONTROL_POINT
Float	4	SCALE1 for TX_CONTROL_POINT
Float	4	LOD2 for TX_CONTROL_POINT
Float	4	SCALE2 for TX_CONTROL_POINT
Float	4	LOD3 for TX_CONTROL_POINT
Float	4	SCALE3 for TX_CONTROL_POINT
Float	4	LOD4 for TX_CONTROL_POINT
Float	4	SCALE4 for TX_CONTROL_POINT
Float	4	LOD5 for TX_CONTROL_POINT
Float	4	SCALE5 for TX_CONTROL_POINT
Float	4	LOD6 for TX_CONTROL_POINT
Float	4	SCALE6 for TX_CONTROL_POINT
Float	4	LOD7 for TX_CONTROL_POINT
Float	4	SCALE7 for TX_CONTROL_POINT
Float	4	Clamp
Int	4	magfilteralpha: 0 = TX_POINT 1 = TX_BILINEAR 2 = None 3 = TX_BICUBIC 4 = TX_SHARPEN 5 = TX_ADD_DETAIL 6 = TX_MODULATE_DETAIL 7 = TX_BILINEAR_GEQUAL 8 = TX_BILINEAR_LEQUAL 9 = TX_BICUBIC_GEQUAL 10 = TX_BIBICUBIC_LEQUAL
Int	4	magfiltercolor:

		0 = TX_POINT 1 = TX_BILINEAR 2 = None 3 = TX_BICUBIC 4 = TX_SHARPEN 5 = TX_ADD_DETAIL 6 = TX_MODULATE_DETAIL 7 = TX_BILINEAR_GEQUAL 8 = TX_BILINEAR_LEQUAL 9 = TX_BICUBIC_GEQUAL 10 = TX_BIBICUBIC_LEQUAL
Float	4	Reserved
Float	4*8	Reserved
Double	8	Lambert conic projection central meridian
Double	8	Lambert conic projection upper latitude
Double	8	Lambert conic projection lower latitude
Double	8	Reserved
Float	4*5	Spare
Int	4	Boolean TRUE if using next 5 integers for detail texture
Int	4	J argument for TX_DETAIL
Int	4	K argument for TX_DETAIL
Int	4	M argument for TX_DETAIL
Int	4	N argument for TX_DETAIL
Int	4	Scramble argument for TX_DETAIL
Int	4	Boolean TRUE if using next for floats for TX_TILE
Float	4	Lower-left u value for TX_TILE
Float	4	Lower-left v value for TX_TILE
Float	4	Upper-right u value for TX_TILE
Float	4	Upper-right v value for TX_TILE

Int	4	Projection
		0 = Flat earth
		3 = Lambert conic
		4 = UTM
		7 = Undefined projection
Int	4	Earth model
		0 = WGS84
		1 = WGS72
		2 = Bessel
		3 = Clark 1866
		4 = NAD27
Int	4	Reserved
Int	4	UTM zone
Int	4	Image origin
		0 = Lower-left
		1 = Upper-left
Int	4	Geospecific points units
		0 = Degrees
		1 = Meters
		2 = Pixels
Int	4	Reserved
Int	4	Reserved
Int	4	Hemisphere for geospecific points units
		0 = Southern
		1 = Northern
Int	4	Reserved
Int	4	Reserved
Int	149*4	Spare
Char	512*1	Comments

Int	13*4	Reserved
Int	4	Attribute file version number
Int	4	Number of geospecific control points

If the number of geospecific control points is > 0, the following fields are also in the attribute file:

Int	4	Reserved

For each geospecific control point:

Double	8*2	Texel u, v of geospecific control point
Double	8*2	Real earth coordinate of geospecific control point (this value depends on the projection, earth model, and geospecific points units)

After all geospecific control points are listed, the following subtexture information appears:

Int	4	Number of subtexture definitions contained in the texture attribute file

If the number of subtexture definitions is >0, the following fields are repeated for each subtexture definition:

Field	Data Type	Length (bytes)	Description
name	char	32	31 character name of subtexture definition, 0 terminates.
left	int	4	Coordinate of left edge of subtexture definition measured in texels.
bottom	int	4	Coordinate of bottom edge of subtexture definition measured in texels.
right	int	4	Coordinate of right edge of subtexture definition measured in texels.
top	int	4	Coordinate of top edge of subtexture definition measured in texels.

The attribute file determines how to parse the texture pattern file, set the texture hardware and software environment for a specific pattern, or position the image in a database.

3 Road Path Files

A road path file contains the attributes of a road path node in ASCII format. The name of the file is user defined. Each attribute is denoted by a keyword, a literal colon, a space, and the value(s). Boolean values are denoted by the string literals “TRUE” and “FALSE”. For the “POINT” keyword its values consist of an XYZ coordinate and an orientation vector, separated by spaces. The orientation vector is specified as either a normal up-vector, or in degrees of heading, pitch, and roll. The “STORE_HPR” keyword specifies which method is used. For path nodes that define the road’s centerline path, construction information for the correlated road section is also stored with additional keywords. Here’s an example:

```
ROAD_ID: 2.0
ROAD_TYPE: Curve
ARC_RADIUS: 175.000000
SPIRAL_LEN1: 80.000000
SPIRAL_LEN2: 80.000000
SUPERELEVATION: 0.080000
CONTROL_POINT: 0.000000 300.000000 0.000000
VCURVE_LEN: 400.000000
VCURVE_MIN: 20.000000
SLOPE1: 0.000000
SLOPE2: 0.000000
WIDTH: 12.000000
CENTER2LEFT: 6.000000
NUM_LANES: 2
LANE_OFFSET: 1.825000
LANE_OFFSET: -1.825000
PROFILE_NAME: /usr/people/db/road/crown.flt
PROFILE_POINT: 12.000000 0.000000
PROFILE_POINT: 9.823453 0.300000
PROFILE_POINT: 6.000000 0.500000
PROFILE_POINT: 3.200000 0.300000
PROFILE_POINT: 0.000000 0.000000
SPEED: 70.000000
NO_PASSING: TRUE
STORE_HPR: TRUE
NUM_POINTS: 12
POINT: 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
POINT: 0.000000 83.548590 0.000000 0.000000 0.000000 0.000000
POINT: 2.906056 145.953096 0.000000 8.000000 0.000000 3.574879
POINT: 6.072530 163.131640 0.000000 13.096178 0.000000 4.573921
POINT: 13.249899 186.467582 0.000000 21.096178 0.000000 4.573921
POINT: 36.936741 229.031118 0.000000 37.096180 0.000000 4.573921
POINT: 71.438096 263.416837 0.000000 53.096180 0.000000 4.573921
POINT: 114.080915 286.960649 0.000000 69.096176 0.000000 4.573921
POINT: 136.868360 293.927470 0.000000 76.903824 0.000000 4.573921
POINT: 166.586342 298.521248 0.000000 84.903824 0.000000 2.853243
POINT: 216.451410 300.000000 0.000000 90.000000 0.000000 0.000000
POINT: 300.000000 300.000000 0.000000 90.000000 0.000000 0.000000
```

This is the first section (a curve) in Road #2 of the database. Section numbers start at zero. Road ID also appears on database node.

Horizontal and vertical curve control values

Road width and centerline placement

Lane information for the road section

Lofting information, including the database file that contains the lofting profile, and (X,Z) points for the lofted section. Lofting information is only printed for the first reference to the profile database.

Passing lane flag (path attribute page)

Heading, Pitch and Roll data will be stored and reported

Data for each point, in the following order:

X, Y, Z, H, P, R

4 Road Zone Files

Zone files are gridded posts files containing elevation and attribute data for a road.

The zone data is followed immediately by a series of (number of spaces in x + 1) * (number of spaces in y + 1) elevation data points. Data begins at the lower-left corner. Succeeding values go from bottom to top, then in columns from left to right.

The elevation data is followed immediately by a series of (number of spaces in x + 1) * (number of spaces in y + 1) surface types corresponding to each of the elevation data points above.

Zone File Format

Data type	Length (bytes)	Description
Int	4	Version (road tools format revision)
Int	4	Spare
Double	8	Lower-left corner x, y, z
Double	8	
Double	8	
Double	8	Upper-right corner x, y, z
Double	8	
Double	8	
Double	8	Grid interval (spacing between data points)
Int	4	Number of data points in x
Int	4	Number of data points in y
Float	4	Low z elevation data point
Float	4	High z elevation data point
Char	440	Spare

Elevation Data Point Format

Data	Length

type	(bytes)	Description
Float	4	Z elevation value

Surface Type Format

Data type	Length (bytes)	Description
Char	1	Road surface type (user defined)

5 Linkage Editor Parameter IDs

Vertex Parameters

ID	Description
258	X coordinate
259	Y coordinate
260	Z coordinate
261	Texture U coordinate
262	Texture V coordinate
265	Color
266	Hard edge flag
267	Freeze normal flag
269	Normal I component
270	Normal J component
271	Normal K component

Face Node Parameters

ID	Description
514	Color
515	Polygon drawing
516	Lighting mode
518	Relative priority
519	Draw both sides flag
520	Texture index
521	Template
522	Infrared
523	Terrain flag

525	Material index
526	Feature ID
527	Surface material code
529	Draw textured faces white
530	IR material
534	Detail texture index
535	Transparency
536	Alternate color
537	LOD control
538	Line style index
539	Light point directional mode
540	Texture mapping

Object Node Parameters

ID	Description
770	Relative priority
771	Inhibit during day flag
772	Inhibit during dusk flag
773	Inhibit during night flag
774	No illumination flag
775	Flat shading flag
776	Shadow flag
777	Transparency
778	Special #1
779	Special #2
782	Significance

LOD Node Parameters

ID	Description
1026	Switch-in distance
1027	Switch-out distance
1028	Special #1
1029	Special #2
1030	Use previous range flag
1031	Center X coordinate
1032	Center Y coordinate
1033	Freeze center flag
1034	Center Z coordinate
1036	Additive LOD’s below flag
1037	Transition distance

Group Node Parameters

ID	Description
1282	Relative priority
1284	Animation type
1286	Bounding volume type
1287	Special #1
1288	Special #2
1289	Replication count
1290	Significance
1291	Layer

DOF Node Parameters

ID	Description
1538	Current Z
1539	Minimum Z
1540	Maximum Z
1542	Current Y
1543	Minimum Y
1544	Maximum Y
1546	Current X
1547	Minimum X
1548	Maximum X
1550	Current pitch
1551	Minimum pitch
1552	Maximum pitch
1554	Current roll
1555	Minimum roll
1556	Maximum roll
1558	Current yaw
1559	Minimum yaw
1560	Maximum yaw
1562	Current Z scale
1563	Minimum Z scale
1564	Maximum Z scale
1566	Current Y scale
1567	Minimum Y scale
1568	Maximum Y scale
1570	Current X scale
1571	Minimum X scale

1572	Maximum X scale
1574	X constrained motion flag
1575	Y constrained motion flag
1576	Z constrained motion flag
1577	Pitch constrained motion flag
1578	Roll constrained motion flag
1579	Yaw constrained motion flag
1580	X scale constrained motion flag
1581	Y scale constrained motion flag
1582	Z scale constrained motion flag
1583	Repeating texture flag
1584	Membrane mode flag

Sound Node Parameters

ID	Description
1796	Amplitude
1797	Pitch bend
1798	Priority
1799	Falloff
1800	Width
1801	Doppler
1802	Absorption
1803	Delay
1804	Directivity
1805	X coordinate
1806	Y coordinate
1807	Z coordinate

1808	Direction vector I component
1809	Direction vector J component
1810	Direction vector K component
1812	Active flag

Switch Node Parameters

ID	Description
2050	Current mask index

Text Node Parameters

ID	Description
2307	Text type
2308	Draw type
2310	Color
2311	Alternate color
2312	Material index
2315	Integer value minimum
2316	Integer value maximum
2317	Float value minimum
2318	Float value maximum
2325	Current integer value
2326	Current float value

2327	Decimal places for float value
2329	Line style index
2330	Justification type
2331	Vertical flag
2332	Bold flag
2333	Italic flag
2334	Underline flag

Light Source Node Parameters

ID	Description
2819	Enabled flag
2820	Global flag
2821	X coordinate
2822	Y coordinate
2823	Z coordinate
2824	Yaw
2825	Pitch

Clip Node Parameters

ID	Description
3074	Plane 0 enable
3075	Plane 1 enable
3076	Plane 2 enable
3077	Plane 3 enable
3078	Plane 4 enable

6 OpenFlight Opcodes

Valid Opcodes

Opcode	Record Type
1	Header
2	Group
4	Object
5	Face
10	Push Level
11	Pop Level
14	Degree of Freedom
19	Push Subface
20	Pop Subface
21	Push Extension
22	Pop Extension
23	Continuation
31	Text Comment
32	Color Palette
33	Long ID
49	Transformation Matrix
50	Vector
52	MultiTexture
53	UV List
55	Binary Separating Plane
60	Replicate
61	Instance Reference

Opcode	Record Type
62	Instance Definition
63	External Reference
64	Texture Palette
67	Vertex Palette
68	Vertex with Color
69	Vertex with Color and Normal
70	Vertex with Color, Normal and UV
71	Vertex with Color and UV
72	Vertex List
73	Level of Detail
74	Bounding Box
76	Rotate about Edge
78	Translate
79	Scale (Nonuniform)
80	Rotate about Point
81	Rotate and/or Scale to Point
82	Put Transform
83	Eyepoint and Trackplane Palette
84	Mesh
85	Local Vertex Pool
86	Mesh Primitive
87	Road Segment
88	Road Zone
89	Morph Vertex List
90	Behavior (Linkage) Palette
91	Sound

Opcode	**Record Type**
92	Road Path
93	Sound Palette
94	General Matrix
95	Text
96	Switch
97	Line Style
98	Clip Region
100	Extension
101	Light Source
102	Light Source Palette
103	Reserved
104	Reserved
105	Bounding Sphere
106	Bounding Cylinder
107	Reserved
108	Bounding Volume Center
109	Bounding Volume Orientation
111	Light Point
112	Texture Mapping Palette
113	Material Palette
114	Color Name Palette
115	Continuously Adaptive Terrain (CAT)
116	CAT Data
117	Reserved
118	Reserved
119	Reserved

Opcode	Record Type
120	Reserved
121	Reserved
122	Push Attribute
123	Pop Attribute
124	Reserved
125	Adaptive Attribute
126	Curve Node

Obsolete Opcodes

Opcode	Function
3	Level of Detail
6	Vertex with ID
7	Short Vertex
8	Vertex with Color
9	Vertex with Color and Normal
12	Translate
13	Degree of Freedom
16	Instance Reference
17	Instance Definition
40	Translate
41	Rotate about Point
42	Rotate about Edge
43	Scale
44	Translate

Opcode	Function
45	Scale (Nonuniform)
46	Rotate about Point
47	Rotate and/or Scale to Point
48	Put Transform
51	Bounding Box
65	Eyepoint Palette
66	Material Palette
77	Scale
110	Histogram Bounding Volume

A Summary of Changes From Version 15.6.0 to Version 15.7.0

Description

The OpenFlight Scene Description for version 15.7.0 coincides with MultiGen Creator version 2.4 and the OpenFlight API version 2.4. The changes made for this version of the OpenFlight Scene Description are in the following sections:

- “Continuation Record ” on this page.
- “Header” on page 129
- “Mesh Overview” on page 129
- “Mesh Record” on page 130
- “Local Vertex Pool” on page 130
- “Mesh Primitive” on page 134
- “MultiTexture Overview” on page 135
- “MultiTexture Record” on page 136
- “UV List” on page 138
- “Texture Attribute” on page 141

This appendix describes these changes.

Continuation Record

All OpenFlight records begin with a 4 byte sequence. The first two bytes identify the record (opcode) and the second two bytes specify the length of the record. Given this regular record structure, the length of all OpenFlight records is limited to the largest value that can be encoded with 2 bytes or 16 bits (65535). In most cases, this maximum size is sufficient but there are cases where it is not. For fixed size records, this is not a problem. For variable size records, this limitation is being addressed with this version.

A new record, called the continuation record is introduced in this version to accommodate variable size records in the OpenFlight Scene Description. The continuation record is used to "continue" a record in the OpenFlight Scene Description file stream. It would appear in the stream immediately following the record that it "continues" (the record that is being continued will be referred to as the "original" record). The data contained in the continuation record is defined by the original record and is assumed to be directly appended onto the content of the original record.

Note: Multiple continuation records may follow a record, in which case all continuation records would be appended (in sequence) to the original record.

Field	Data Type	Length	Description
opcode	unsigned short	2	Continuation Opcode 23
length	unsigned short	2	Length of the record
content	varies	length - 4	Depends on the original record. The contents of this field are to be appended directly to the end of the original record contents before the original record contents are parsed.

In theory, any OpenFlight record may be "continued," but in practice only variable length records whose length exceeds 65535 bytes are. Following is a list of the variable length OpenFlight record types to which the continuation record may apply:

- **Extension Node and Attributes Records** opcode 100
- **Name Table Record** opcode 114
- **Local Vertex Pool** opcode 8

Example: In the following example, the color name table is "too" big to fit in 65535 bytes so the primary palette record, `NAME TABLE`, is followed by one (or more) `CONTINUATION` records. The contents of each of the continuation records is appended to the contents of the name table record before the name table data is parsed.

```
NAME TABLE
CONTINUATION
CONTINUATION
```

Header

New attributes have been added to the end of the existing Header record (opcode 1). The following new fields appear after the last reserved field (following the Next Curve node ID number) of the existing header record.

Field	Data Type	Length	Description
delta z	double	8	Delta z to place database (used in conjunction with existing Delta x and Delta y values
radius	double	8	Distance fromdatabase origin to farthest corner.
nextmeshid	unsigned short	2	Next Mesh node ID number.
Reserved	unsigned short	2	Reserved

Mesh Overview

Meshes define geometric primitives that share attributes and vertices. In previous versions of OpenFlight, the fundamental geometric construct was the polygon. Each polygon has a unique set of attributes and vertices. Meshes are used to represent "sets" of related polygons, each sharing common attributes and vertices. Using a mesh, related polygons can be represented in a much more compact format. Each mesh will share one set of "polygon" attributes (color, material, texture, etc.), a common "vertex pool" and one or more geometric primitives that use the shared attributes and vertices. Using a mesh you can represent triangle strips, triangle fans, quadrilateral strips and indexed face sets.

A mesh node actually consists of three distinct records:

- *Mesh* - defines the "polygon" attributes associated to all geometric primitives of the mesh.
- *Local Vertex Pool* - defines the set of vertices that are referenced by the geometric primitives of the mesh.
- *Mesh Primitive* - defines a geometric primitive (triangle-strip, triangle-fan, quadrilateral-strip or indexed face set) for the mesh.

A mesh node record contains one Mesh record, one Local Vertex Pool record, and one or more Mesh Primitive records as shown in the following example:

```
MESH
LOCAL VERTEX POOL
```

```
PUSH
MESH PRIMITIVE
MESH PRIMITIVE
...
MESH PRIMITIVE
POP
```

Mesh Record

The mesh record is the primary record of a mesh node and defines the common "polygon-like" attributes associated to all geometric primitives of the mesh. To that end, the contents of the mesh record are identical to those of the Polygon (or Face) record.

Field	Data Type	Length (bytes)	Description
opcode	unsigned short	2	Mesh opcode 84
length	unsigned short	2	Length of the record
id	char	8	7 character ASCII id, 0 terminates
polygon attributes	See Description	See Description	The content of this record is the same as that of the Face Record opcode 5. See the OpenFlight Scene Description specification for a description of these fields.

Local Vertex Pool

This record defines a set of vertices that is referenced by the geometry of the mesh.

Note: Currently the Local Vertex Pool will be used exclusively in the context of meshes but is designed in general way so that it may appear in other contexts in future versions of the OpenFlight Scene Description. .

Field	Data Type	Length (bytes)	Description
opcode	unsigned short	2	Local Vertex Pool opcode 85

Field	Data Type	Length (bytes)	Description
length	unsigned short	2	Length of the record NOTE: Since the length of this record is represented by an unsigned short, the maximum length of the vertex pool is 216-1 (or 65535). If the entire vertex pool cannot "fit" into this size, one or more Continuation records will follow
numvert	unsigned int	4	Number of vertices contained in this record.
attrmask	unsigned int	4	32 bit mask indicating what kind of vertex information is contained in this vertex list. Bits are ordered left to right (bit 1 is leftmost). Each of the bits of the mask are defined in the following table.

Bit #	Description
1	**Has Position** - indicates that each vertex in the list includes x, y, and z coordinates (three double-precision floating point values)
2	**Has Color Index** - indicates that each vertex in the list includes a color value that is a color table index (one integer value)
3	**Has RGBColor** - indicates that each vertex in the list includes a color value that is a packed RGB color value (one integer value)
NOTE: Bits 2 and 3 are mutually exclusive - a vertex can have either a color index or an RGB color value or neither, but cannot have both a color index and an RGB value.	
4	Has Normal - indicates that each vertex in the list includes a normal (three single-precision floating point values)
5	Has Base UV - indicates that each vertex in the list includes uv texture coordinates for the base texture (two single-precision floating point values)
6	**Has UV 1** - indicates that each vertex in the list includes uv texture coordinates for layer 1 (two single-precision floating point values)

Bit #	Description
7	**Has UV 2** - indicates that each vertex in the list includes uv texture coordinates for layer 2 (two single-precision floating point values)
8	**Has UV 3** - indicates that each vertex in the list includes uv texture coordinates for layer 3 (two single-precision floating point values)
9	**Has UV 4** - indicates that each vertex in the list includes uv texture coordinates for layer 4 (two single-precision floating point values)
10	**Has UV 5** - indicates that each vertex in the list includes uv texture coordinates for layer 5 (two single-precision floating point values)
11	**Has UV 6** - indicates that each vertex in the list includes uv texture coordinates for layer 6 (two single-precision floating point values)
12	**Has UV 7** - indicates that each vertex in the list includes uv texture coordinates for layer 7 (two single-precision floating point values)
13-32	**Spare** - reserved for future expansion

The following fields are repeated for each vertex in the list, depending on the bits set in the attrmask field above:

Field	Data Type	Length (bytes)	Description
X_N	double	8	X coordinate N - present if attrmask includes Has Position
Y_N	double	8	Y coordinate N - present if attrmask includes Has Position
Z_N	double	8	Z coordinate N - present if attrmask includes Has Position
$color_N$	unsigned int	4	Color for vertex N - present if attrmask includes Has Color Index or Has RGB Color. Value is color table index if attrmask includes Has Color Index, packed color value (A, B, G, R) if attrmask includes Has RGB Color.
normal i_N	float	4	i component of normal for vertex N - present if attrmask includes Has Normal
normal j_N	float	4	j component of normal for vertex N - present if attrmask includes Has Normal

Field	Data Type	Length (bytes)	Description
normal k_N	float	4	k component of normal for vertex N - present if attrmask includes Has Normal
$uBase_N$	float	4	u texture coordinate for base texture of vertex N - present if attrmask includes Has Base UV
$vBase_N$	float	4	v texture coordinate for base texture of vertex N - present if attrmask includes Has Base UV
$u1_N$	float	4	u texture coordinate for layer 1 of vertex N - present if attrmask includes Has UV1
$v1_N$	float	4	v texture coordinate for layer 1 of vertex N - present if attrmask includes Has UV1
$u2_N$	float	4	u texture coordinate for layer 2 of vertex N - present if attrmask includes Has UV1
$v2_N$	float	4	v texture coordinate for layer 2 of vertex N - present if attrmask includes Has UV1
$u3_N$	float	4	u texture coordinate for layer 3 of vertex N - present if attrmask includes Has UV1
$v3_N$	float	4	v texture coordinate for layer 3 of vertex N - present if attrmask includes Has UV1
$u4_N$	float	4	u texture coordinate for layer 4 of vertex N - present if attrmask includes Has UV1
$v4_N$	float	4	v texture coordinate for layer 4 of vertex N - present if attrmask includes Has UV1
$u5_N$	float	4	u texture coordinate for layer 5 of vertex N - present if attrmask includes Has UV1
$v5_N$	float	4	v texture coordinate for layer 5 of vertex N - present if attrmask includes Has UV1
$u6_N$	float	4	u texture coordinate for layer 6 of vertex N - present if attrmask includes Has UV1
$v6_N$	float	4	v texture coordinate for layer 6 of vertex N - present if attrmask includes Has UV1
$u7_N$	float	4	u texture coordinate for layer 7 of vertex N - present if attrmask includes Has UV1

Field	Data Type	Length (bytes)	Description
$v7_N$	float	4	v texture coordinate for layer 7 of vertex N - present if attrmask includes Has UV1

Mesh Primitive

This record defines a geometric primitive for a mesh.

Field	Data Type	Length (bytes)	Description
opcode	unsigned short	2	Mesh Primitive opcode 86
length	unsigned short	2	Length of the record
type	unsigned short	2	Primitive Type - can be one of the following values: 1 - Triangle Strip 2 - Triangle Fan 3 - Quadrilateral Strip 4 - Indexed Polygon **Note**: This field specifies how the vertices of the primitive are interpreted.
indexsize	unsigned short	2	Specifies the length (in bytes) of each of the vertex indices that follow - will be 1, 2, or 4
numverts	unsigned int	2	Number of vertices contained in this primitive
The following field is repeated for each vertex in the primitive. These vertices are interpreted according to the primitive type field above.			
vertex $index_N$	var	var	Index of vertex N of the mesh primitive. The size of each of these indices is specified in the indexsize field above.

There are four mesh primitive types. Each is represented using the Mesh Primitive record described above. The following describes how the vertices of each primitive type are interpreted as geometry.

Primitive Type	Description
Triangle Strip	This mesh primitive defines a connected group of triangles in the context of the enclosing mesh. Each triangle shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One triangle is defined for each vertex presented after the first two vertices. For odd n, vertices n, n + 1, and n + 2 define triangle n. For even n, vertices n + 1, n, and n + 2 define triangle n. N vertices represent N - 2 triangles.
Triangle Fan	Like the Triangle Strip, this mesh primitive also defines a connected group of triangles in the context of the enclosing mesh. Each triangle shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One triangle is defined for each vertex presented after the first two vertices. Vertices 1, n + 1, and n + 2 define triangle n. N vertices represent N - 2 triangles.
Quadrilateral Strip	This mesh primitive defines a connected group of quadrilaterals in the context of the enclosing mesh. Each quadrilateral shares the "polygon" attributes defined by the enclosing mesh. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One quadrilateral is defined for each pair of vertices presented after the first pair. Vertices 2n - 1, 2n, 2n + 2, and 2n + 1 define quadrilateral n. N vertices represent N-3 quadrilaterals.
Indexed Polygon	This mesh primitive defines a single polygon in the context of the enclosing mesh. This primitive is similar to the other mesh primitives in that it also shares the polygon attributes of the enclosing mesh. It is different from the other mesh primitive types in that while triangle strips/fans and quadrilateral strips describe a set of connected triangles/quadrilaterals, the indexed polygon defines a single polygon. This primitive contains a sequence of indices that reference vertices from the local vertex pool. One polygon is defined by the sequence of vertices in this record. N vertices represent 1 N-sided closed polygon or 1 (N-1)-sided unclosed polygon.

MultiTexture Overview

OpenFlight supports 8 textures per polygon or mesh as well as 8 uv’s per vertex. The current texture information stored on the polygon is referred to as "the base texture" or "texture layer 0". Each additional texture is referred to as "texture layer N". Therefore, to support 8 textures per polygon, a base texture is required as well as 7 additional texture layers. The additional texture layers for each polygon, mesh, and vertex will be represented in ancillary records at the Face, Mesh, and Vertex primary node levels as shown in the following example:

```
FACE
MULTITEXTURE
PUSH
VERTEX LIST
UV LIST
POP
```

MultiTexture Record

The multitexture record is an ancillary record of Face and Mesh nodes. It specifies the texture layer information for the face or mesh. .

Field	Data Type	Length (bytes)	Description
opcode	unsigned short	2	MultiTexture opcode 52
length	unsigned short	2	Length of the record
multitexmask	unsigned int	4	32 bit mask indicating what kind of multitexture information follows. The bits appear from left to right, and are defined in the following table.

Bit #	Description
1	**Has Layer 1** - indicates that multitexture information for texture layer 1 is present
2	**Has Layer 2** - indicates that multitexture information for texture layer 2 is present
3	**Has Layer 3** - indicates that multitexture information for texture layer 3 is present
4	**Has Layer 4** - indicates that multitexture information for texture layer 4 is present
5	**Has Layer 5** - indicates that multitexture information for texture layer 5 is present

Bit #	Description
6	**Has Layer 6** - indicates that multitexture information for texture layer 6 is present
7	**Has Layer 7** - indicates that multitexture information for texture layer 7 is present
8-32	**Spare** - reserved for future expansion

The following fields are repeated for each multitexture layer that is specified as present by the bits set in multitexmask. This mechanism allows for “sparse” multitexture layer information to be present, and does not required that the information be contiguous.:

Field	Data Type	Length (bytes)	Description
texture_N	unsigned short	2	Texture index for texture layer N
effect_N	unsigned short	2	MultiTexture effect for layer N - can be any of the following values: 0 - Texture Environment 1 - Bump Map 2 - 100 Reserved by MultiGen-Paradigm for future expansion >100 - User (runtime) defined
mapping_N	unsigned short	2	Texture mapping index for texture layer N
data_N	unsigned short	2	Texture data for layer N. This is user defined. For example, it may be used as a blend percentage, or color, or any other data needed by the runtime to describe texture layer N.

UV List

Note: The uv list record is an ancillary record of vertex nodes. This record (if present) always follows the vertex list or morph vertex list record and contains texture layer information for the vertices represented in the applicable vertex list record.

Field	Data Type	Length (bytes)	Description
opcode	unsigned short	2	MultiTexture opcode 52
length	unsigned short	2	Length of the record
uvmask	unsigned int	4	32 bit mask indicating which uv values follow. The bits appear from left to right, and are defined in the following table.

Bit #	Description
1	**Has Layer 1** - indicates that multitexture uv for texture layer 1 is present
2	**Has Layer 2** - indicates that multitexture uv for texture layer 2 is present
3	**Has Layer 3** - indicates that multitexture uv for texture layer 3 is present
4	**Has Layer 4** - indicates that multitexture uv for texture layer 4 is present
5	**Has Layer 5** - indicates that multitexture uv for texture layer 5 is present
6	**Has Layer 6** - indicates that multitexture uv for texture layer 6 is present
7	**Has Layer 7** - indicates that multitexture uv for texture layer 7 is present
8-32	**Spare** - reserved for future expansion

The following fields are repeated for each vertex contained in the corresponding vertex list or morph vertex list record.

- If this uv list record follows a vertex list record, the following fields are repeated for each layer present (as specified by the bits set in uvmask):

Field	Data Type	Length (bytes)	Description
$u_{i, N}$	float	4	Texture coordinate U for vertex i, layer N
$v_{i, N}$	float	4	Texture coordinate V for vertex i, layer N

- If this uv list record follows a morph vertex list record, the following fields are repeated for each layer present (as specified by the bits set in uvmask):

Field	Data Type	Length (bytes)	Description
$u0_{i, N}$	float	4	Texture coordinate U for the 0% vertex i, layer N
$v0_{i, N}$	float	4	Texture coordinate V for the 0% vertex i, layer N
$u100_{i, N}$	float	4	Texture coordinate U for the 100% vertex i, layer N
$v100_{i, N}$	float	4	Texture coordinate V for the 100% vertex i, layer N

Example: Consider a triangular face (3 vertices) that contains morph vertex information and has texture layers 1 and 3 defined. The following shows the contents of the uv list record corresponding to the morph vertex list record representing this triangle:

Field	Data Type	Length (bytes)	Description
$u0_{i, N}$	float	4	Texture coordinate U for the 0% vertex i, layer N
$v0_{i, N}$	float	4	Texture coordinate V for the 0% vertex i, layer N
$u100_{i, N}$	float	4	Texture coordinate U for the 100% vertex i, layer N
$v100_{i, N}$	float	4	Texture coordinate V for the 100% vertex i, layer N

Field	Data Type	Length (in bytes)	Value
opcode	unsigned short	2	53 (UV List opcode).
length	unsigned short	2	200 (Length of the record).
uvmask	unsigned int	4	1010 0000 0000 0000 (layers 1 and 3 ON, others OFF).
$u0_{1,1}$	float	8	Texture coordinate U for the 0% vertex 1, layer 1.
$v0_{1,1}$	float	8	Texture coordinate V for the 0% vertex 1, layer 1.
$u100_{1,1}$	float	8	Texture coordinate U for the 100% vertex 1, layer 1.
$v100_{1,1}$	float	8	Texture coordinate V for the 100% vertex 1, layer 1.
$u0_{1,3}$	float	8	Texture coordinate U for the 0% vertex 1, layer 3.
$v0_{1,3}$	float	8	Texture coordinate V for the 0% vertex 1, layer 3.
$u100_{1,3}$	float	8	Texture coordinate U for the 100% vertex 1, layer 3.
$v100_{1,3}$	float	8	Texture coordinate V for the 100% vertex 1, layer 3.
$u0_{2,1}$	float	8	Texture coordinate U for the 0% vertex 2, layer 1.
$v0_{2,1}$	float	8	Texture coordinate V for the 0% vertex 2, layer 1.
$u100_{2,1}$	float	8	Texture coordinate U for the 100% vertex 2, layer 1.
$v100_{2,1}$	float	8	Texture coordinate V for the 100% vertex 2, layer 1.
$u0_{2,3}$	float	8	Texture coordinate U for the 0% vertex 2, layer 3.
$v0_{2,3}$	float	8	Texture coordinate V for the 0% vertex 2, layer 3.
$u100_{2,3}$	float	8	Texture coordinate U for the 100% vertex 2, layer 3.
$v100_{2,3}$	float	8	Texture coordinate V for the 100% vertex 2, layer 3.
$u0_{3,1}$	float	8	Texture coordinate U for the 0% vertex 3, layer 1.
$v0_{3,1}$	float	8	Texture coordinate V for the 0% vertex 3, layer 1.
$u100_{3,1}$	float	8	Texture coordinate U for the 100% vertex 3, layer 1.

Field	Data Type	Length (in bytes)	Value
$v100_{3,1}$	float	8	Texture coordinate V for the 100% vertex 3, layer 1.
$u0_{3,3}$	float	8	Texture coordinate U for the 0% vertex 3, layer 3.
$v0_{3,3}$	float	8	Texture coordinate V for the 0% vertex 3, layer 3.
$u100_{3,3}$	float	8	Texture coordinate U for the 100% vertex 3, layer 3.
$v100_{3,3}$	float	8	Texture coordinate V for the 100% vertex 3, layer 3.

Texture Attribute

Subtexture definitions have been added to the end of the Texture Attribute File definition. The following new fields appear after the geospecific control points in the texture attribute file.

Field	Data Type	Length (bytes)	Description
numsubtextures	int	4	Number of subtexture definitions contained in texture attribute file.
If the number of subtexture definitions is > 0, the following fields are repeated for each subtexture definition:			
name	char	32	31 character name of subtexture definition, 0 terminates.
left	int	4	Coordinate of left edge of subtexture definition measured in texels.
bottom	int	4	Coordinate of bottom edge of subtexture definition measured in texels.
right	int	4	Coordinate of right edge of subtexture definition measured in texels.

Field	Data Type	Length (bytes)	Description
top	int	4	Coordinate of top edge of subtexture definition measured in texels.

Index