FLIGHT Worlds

Order Number: TBD

June 1991

This manual describes the FLIGHT notion of worlds, how users may construct their own worlds, including information on how to invoke the FLIGHT world compiler and use of the FLIGHT world editor.

Revision/Update Information:	This document supercedes *FLIGHT Worlds* V2.4
Operating System and Version:	VMS Version 5.4
Software Version:	FLIGHT, Version 2.5

digital equipment corporation

Table of Contents

List of Figures

List of Tables

1 Worlds

A FLIGHT world is a collection of masses, objects, NDB transmitters, VOR transmitters, ILS installations, viewpoints, entry points and takeoff assists in binary form. The world compiler is used to compress and convert textual world descriptions into binary form which is acceptable by the FLIGHT server. For more information about invoking the world compiler, see Chapter 2.

A FLIGHT world source file is a text file which a designer creates to describe the objects which are found in a world. The following types of objects may be found in a world:

- Masses

  Masses are spherical, gravitationally attractive bodies (such as planets, moons, and asteroids).

- Objects

  Objects are generic, three-dimensional constructions which are displayed to the pilot of the aircraft during flight. Objects may be constructed to be targets suitable for destruction if desired. For example, terrain is classified as an object.

- Navigation aids

  Navigation aids are Non-Directional Beacons (NDBs), Very-high-frequency Omni-Range transmitters (VORs), and Instrument Landing System installations (ILSs). Each may be picked up by aircraft equipped with appropriate instrumentation.

- Takeoff assists

  Takeoff assists are equivalent to catapults in the FLIGHT environment.

- Viewpoints

  Viewpoints are locations in the world from which a pilot may watch himself while flying.

- Entry points

Entry points are locations in the world where a simulation may start. For example, commonly, entry points are locations at the ends of runways, and so on.

Each object is described by a series of commands (similar to those used in DCL) in the world source file. An example of such a command is the GEOMETRY LINE command.

```
OPEN OBJECT="Mountain"
SIGNATURE RADAR 10,100,50
```

The preceeding example shows two lines of a sample source file, one using the OPEN command and the other using the SIGNATURE command. For a description of each command and its qualifiers, see Chapter TBS.

1.1 Organization of World Source Files

World source files contain a sequence of commands. However, the commands the world source file contains may not be specified in an arbitrary order. The following rules apply to the order of the commands in a world source file:

- Objects must be defined between OPEN OBJECT and CLOSE OBJECT pairs.
- Objects must be defined before they are referenced in PLACE commands.

The world compiler imposes the following limits on the contents of a world.

- 64 masses
- 256 object placements

  This allows for 3 levels of detail with 300 points, 300 lines, and 200 triangles per level of detail. Up to 20 volumes may be specified for each object.
- 256 entry points
- 20 viewpoints
- 256 NDB transmitters
- 256 VOR transmitters
- 256 ILS installations
- 64 takeoff assists

There is no reference mechanism available in the world compiler to allow objects defined in one world to be placed in another, except by including the source descriptions of those objects into the other world's sources.

1.2 Structuring World Source Files

Because object descriptions can be quite lengthy (2000 source lines is not uncommon for the example worlds), a mechanism is provided to allow designers to include multiple source files into a single world's compilation using the INCLUDE command. The INCLUDE command compiles the world commands in an included file just as if they were in the original file.

The following example shows a large world broken up into smaller source files.

```
INCLUDE LOGAN_AIRPORT.DAT
INCLUDE MAYNARD.DAT
INCLUDE MARTHAS_VINEYARD.DAT
```

1.3 Commenting Designs

The world commands which FLIGHT uses are intended to be self-describing. It is often useful, however, when the designer adds comments to the source files to add to the readability of the designs. This is especially useful when a designer must go back to a design and change some aspect of it.

Using the exclamation character, designers can add comments to their source files. By placing an exclamation character before the beginning of a line in a source file, a designer can add arbitrary text to the source file following the exclamation character. The world compiler will ignore any line with an exclamation character preceding it, even if the source line following the exclamation character contains valid world commands.

The following example shows how information is added to a design using exclamation characters.

```
OPEN OBJECT="Mountain"
!
! Mountains have a very large radar signature
!
SIGNATURE RADAR 1,1,1
!
! The geometry is for a 1-foot high mountain. Use the MAGNIFICATION
! qualifier on the PLACE command to get them to their real size.
```

```
!
GEOMETRY POINT ...
!
CLOSE OBJECT
```

2 The World Compiler

The world compiler is the processor which converts world source files into the binary form which the FLIGHT server can read. The world compiler may only be invoked as a batch processor. That is, there is a single command which invokes the compiler and then it runs to completion without any user interaction.

2.1 DCL Command

The following describes the invocation of the world compiler.

FLIGHT/COMPILE=WORLD

Invokes FLIGHT's world compiler.

FORMAT

FLIGHT/COMPILE=WORLD *input-filespec*

Qualifiers	**Defaults**
/(NO)LISTING(=filespec)	*/NOLISTING*
/(NO)OUTPUT(=filespec)	*/OUTPUT=*.WORLD*
/(NO)LOG	*/NOLOG*
/DEFINE=(name, (name,...))	*None*

PARAMETERS

input-filespec

Specifies the input file to be compiled. Note that wildcards are not allowed in the input file specification. The default file type for this parameter is DAT.

DESCRIPTION

The FLIGHT/COMPILE=WORLD command starts FLIGHT's world compiler. See Section 2.2 for a description of the commands that are used in world source files to describe the contents of a world.

As the world source file is compiled, errors may be encountered. FLIGHT will list the source line containing the error along with an informative message about the error to the terminal. If a listing file was requested via the LISTING qualifier, then the informative message is logged there as well.

QUALIFIERS

/(NO)LISTING(=filespec)

Specifies whether or not the compiler creates a listing file. If you omit the file specification, the compiler defaults to the name of the source file and a file type of LIS. The default is NOLISTING.

/(NO)OUTPUT(=filespec)

Specifies whether or not the compiler creates an output file containing the compiled world. If you omit the file specification, the compiler defaults to the name of the source file and a file type of WORLD. The default is OUTPUT=<inputfile>.WORLD.

Specifying NOOUTPUT can be useful to ensure that a world compiles correctly, but you do not want to use the disk space required to store the compile world file.

/(NO)LOG

Specifies whether or not the compiler outputs an informational message whenever it encounters the beginning of a world object definition. This may be used to watch the progress of the compiler as it compiles each object in your world sources.

/DEFINE=(name, (name, ...))

Specifies the set of symbol names which should be considered as defined for the duration of this compilation. This has the same effect as if "_DEFINE name" directives were placed into the beginning of the source file.

2.2 Compiler Directives

The following section provides complete descriptions of a set of directives which may be specified in a source file to control the behavior of the compiler.

All compiler directives begin with an underscore (_).

_DEFINE

Directs the compiler to define a symbol

FORMAT

_DEFINE *symbol*

Qualifiers	**Defaults**
None.	*None.*

PARAMETERS

symbol

Specifies the name of the symbol to define. Symbol names are case sensitive, but if the symbol name is not in quotes, it is converted to uppercase.

DESCRIPTION

Use the _DEFINE directive to tell the compiler that a particular symbol name should be defined as valid. Once defined, the _IFDEFINED and _IFNOTDEFINED directives may be used to conditionally compile sections of source code.

Up to 16 symbols may be defined during a single compilation. Symbols definitions may not be removed or negated in any way.

QUALIFIERS

None

_IFDEFINED

Directs the compiler to conditionally compile source lines based on the existence of a symbol definition.

FORMAT

_IFDEFINED *symbol*

Qualifiers	Defaults
None.	*None.*

PARAMETERS

symbol

Specifies the name of the symbol to check. If the symbol is defined, then the lines following the _IFDEFINED directive and up to the matching _ENDIF directive will be compiled. If not defined, then those lines will be ignored.

DESCRIPTION

Use the _IFDEFINED directive to tell the compiler that lines between this directive and its matching _ENDIF directive should be compiled if the specified symbol name is defined.

QUALIFIERS

None

EXAMPLES

```
_DEFINE SOLID

.
.
.
OPEN OBJECT="Tunnel"
.
.
.
_IFDEFINED SOLID
VOLUME 0 0 0 0 0/DESTROY=()
```

```
_ENDIF
.
.
.
CLOSE OBJECT
```

This example shows a _DEFINE directive causing the compiler to consider the SOLID symbol defined. Later in the source, an object conditionally compiles a VOLUME command if the SOLID symbol is defined. Due to the presence of the _DEFINE directive, that symbol is defined.

A more common scenario is to use the /DEFINE qualifier on the FLIGHT/COMPILE command. This allows for selection of the form of the object that is desired at the command line; no editing of the source file is needed in order to change the outcome of a particular compilation.

_IFNOTDEFINED

Directs the compiler to conditionally compile source lines based on the absence of a symbol definition.

FORMAT

_IFNOTDEFINED *symbol*

Qualifiers	**Defaults**
None.	*None.*

PARAMETERS

symbol

Specifies the name of the symbol to check. If the symbol is not defined, then the lines following the _IFNOTDEFINED directive and up to the matching _ENDIF directive will be compiled. If defined, then those lines will be ignored.

DESCRIPTION

Use the _IFNOTDEFINED directive to tell the compiler that lines between this directive and its matching _ENDIF directive should not be compiled if the specified symbol name is defined.

QUALIFIERS

None

_ENDIF

Directs the compiler to end the processing effect produced by a matching _IFDEFINED or _IFNOTDEFINED directive.

FORMAT

_ENDIF

Qualifiers	**Defaults**
None.	*None.*

PARAMETERS

none

DESCRIPTION

Use the _ENDIF directive to end the processing effect produced by a matching _IFDEFINED or _IFNOTDEFINED directive. Note that these directives may be nested and that an _ENDIF directive is required for each _IFDEFINED or _IFNOTDEFINED directive specified.

QUALIFIERS

None

2.3 World Commands

The following section provides complete descriptions of all the source commands used to describe a world.

ASSIST

Specifies a takeoff assist path for the world.

FORMAT

ASSIST *xpos ypos zpos xrot yrot zrot*

Qualifiers	Defaults
/TARGET_RADIUS=feet	*/TARGET_RADIUS=10.0*
/ASSIST_RADIUS=feet	*/ASSIST_RADIUS=200.0*
/THRUST=pounds	*/THRUST=20000.0*
/LINEUP_TOLERANCE=degrees	*/LINEUP_TOLERANCE=10.0*

PARAMETERS

xpos ypos zpos

Specifies the base of the assist path. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

xrot yrot zrot

Specifies the rotation of the assist path, where the base of the assist path is the pivot point, and the starting orientation of the assist path is due East. All rotations are counterclockwise from the positive axis of rotation.

DESCRIPTION

Use the ASSIST command to define a takeoff assist path which users may invoke during a flight simulation. These assist paths are typically used for catapult simulation and provide a thrust assist for a certain distance from the assist base point.

For an ASSIST to be invoked, the user's aircraft must be stopped within the target radius of the assist base point and lined up to the assist path within the angular lineup tolerance. The keybinding command invoke assist is the mechanism for invoking an assist.

QUALIFIERS

/TARGET_RADIUS=feet

Specifies the maximum distance that an aircraft can be from the assist base point and still be able to invoke this assist.

/ASSIST_RADIUS=feet

Specifies how far along the assist path that the assist is provided. No check is made on the lineup of the aircraft to the assist path once an assist is successfully invoked.

/THRUST=pounds

Specifies how much constant thrust assist is provided while the aircraft is still withing the assist radius.

/LINEUP_TOLERANCE=degrees

Specifies the limit of alignment of an aircraft attempting to use this assist. If the aircraft is not properly lined up within the specified tolerance, FLIGHT will not allow invocation of the assist.

CHARACTERISTICS

Specifies one or more characteristics of a world object

FORMAT

CHARACTERISTICS *characteristic (,characteristic)*

Qualifiers	**Defaults**
None	*None*

PARAMETERS

characteristic (,characteristic...)

Tells FLIGHT that the object currently open for definition should have the named characteristics. Table 2-1 list the possible characteristics that you can supply on a CHARACTERISTICS command.

Table 2-1 Object Characteristics Keywords

Keyword	Description
UNDETECTABLE	Indicates that aircraft flying in this world can fly through this object without being damaged or destroyed
FILLER	Indicates that this object consists solely of connections between other world objects. The intent is to provide the ability to fill the gaps between existing objects. See the /XREF qualifier to the GEOMETRY POINT command.
HORIZON_CLIPPING	Indicates that this object is of a type which FLIGHT's level of detail capabilities do not suffice. When this object's origin is occluded by a world mass, it will not be displayed, regardless of the level of detail being displayed.

DESCRIPTION

Use the CHARACTERISTICS command to specify known characteristics of the currently opened object

Note that the last CHARACTERISTICS command encountered during an object definition is used for all object characteristics. That is, you may not have multiple CHARACTERISTICS commands to accumulate CHARACTERISTICS for an object.

QUALIFIERS *None*

CLOSE OBJECT

Closes an object open for definition

FORMAT

CLOSE OBJECT

Qualifiers	Defaults
None	*None*

PARAMETERS

None

DESCRIPTION

Use the CLOSE command to close the currently-open object. There must be a CLOSE command for every OPEN in a world definition. If a CLOSE command is omitted, the open object will not be added to the world.

FLIGHT performs a variety of checks to verify the integrity of the object being closed. One or more error messages may be output at the CLOSE which actually refer to information which was (or should have been) specified earlier in the world source file.

QUALIFIERS

None

CLOSE SUBOBJECT

Closes a subobject open for definition

FORMAT

CLOSE SUBOBJECT

Qualifiers	**Defaults**
None	*None*

PARAMETERS

None

DESCRIPTION

Use the CLOSE command to close the currently-open subobject. There must be a CLOSE command for every OPEN in a world definition. If a CLOSE command is omitted, the open subobject will not be added to the world.

QUALIFIERS

None

DEFINE COLOR

Specifies a color value for use in the current world.

FORMAT

DEFINE COLOR *color-name*

Qualifiers	**Defaults**
/RGB_VALUE=(red,green,blue)	*None*

PARAMETERS

color-name

Specifies the color to be defined. If the name specified is known by the X server on which this color will be displayed, then the RGB_VALUE qualifier is not necessary. If you define your own color name, then you must specify the RGB_VALUE value.

Color names are case-sensitive.

DESCRIPTION

The DEFINE COLOR command specifies a color name which can be used by objects in their GEOMETRY COLOR commands. If a GEOMETRY COLOR command references an undefined color name, a color definition is automatically performed.

Colors are assigned during a simulation on a first-come first-served basis. So the order by which colors are defined determines the likelihood of them being used as requested. For example, 30 colors might be defined in a world. However, on a system with few color resources, perhaps only new 5 colors may be allocated. The 5 colors that will be allocated will be the first 5 defined in the world. Therefore, the remaining 25 colors must be mapped to the closest color that the display can handle.

It is important to understand that the first color defined, either explicitly with a DEFINE COLOR command, or implicitly by a new color name specified in a GEOMETRY COLOR command, is used as the world background color.

QUALIFIERS

/RGB_VALUE=(red,green,blue)

Specifies the exact red, green and blue components of the named color. Each value is on a scale from 0.0 to 1.0. If this qualifier is not specified, then FLIGHT will rely on the X server's knowledge of colors to determine what the appearance of objects using this color will be.

EXAMPLES

```
DEFINE COLOR "Brown"
```

Defines a color called Brown. The command relies on the display device knowing what the color Brown looks like. If it does not, the results are undefined.

```
DEFINE COLOR "Off-White"/RGB_VALUE=(0.83,0.90,1.0)
```

Defines a color called Off-White. The command specifies the exact appearance of the color.

DEFINE COORDINATES

Controls how coordinate specifications are interpreted

FORMAT

DEFINE COORDINATES *method*

Qualifiers	**Defaults**
/MASS="mass-name"	*None*

PARAMETERS

method

Specifies the method of coordinate specification to be used from this command until the next DEFINE COORDINATS command. Table 2-2 lists the coordinate method keywords and their meaning.

Table 2-2 Coordinate Methods

Keyword	Description
ABSOLUTE	Coordinates will be treated as absolute coordinates in the three dimensional universe. They should be specified as three floating-point numbers in order X, Y, then Z. Map directions (North, South, etc) are determined by the placements of objects relative to the nearest mass.
SPHERICAL	Coordinates will be treated as latitude, longitude and altitude values based ona particular mass. The mass to base the coordinate on must be specified by the /MASS qualifier. The coordinate value should be specified as latitude, then longitude, then altitude. A latitude or longitude value is specified by two integers for degrees followed by minutes, then a single floating point value specifying seconds.
MAP	Coordinates will be treated as distances east and south of the zero latitude and zero longitude point of the specified mass. The mass is named via the /MASS qualifier. The order of coordinates is distance-east, altitude, distance-south.

DESCRIPTION

Use the DEFINE COORDINATES command to specify that FLIGHT should interpret coordinates in a new way. This command affects all coordinate specifications (xpos, ypos, zpos triplets) in all commands except the MASS command, which always acts as if a DEFINE COORDINATES ABSOLUTE directive was in effect.

When a world is started the coordinate method defaults to ABSOLUTE.

All rotations of objects are in relation to their orientation after placement. That is, rotations are applied after a MAP or SPHERICAL placement has occured such that the object is on the surface of the specified mass.

QUALIFIERS

/MASS="mass-name"

Specifies the mass which new coordinate specifications are in relation to. This is only valid for coordinate methods MAP and SPHERICAL. See the MASS command for a description of how to specify and name a mass.

EXAMPLES

```
DEFINE COORDINATES ABSOLUTE
PLACE "Tower" 30000.0 10300.0 4000.0  0.0 20.0 0.0
```

This sets the coordinate specification to absolute coordinates and then places an object called "Tower" at specific 3D coordinates with specific rotations.

```
DEFINE COORDINATES SPHERICAL/MASS="Earth"
PLACE "Fir Tree" "43 10 0.0" "18 0 0.0" 8.0  0.0 10.0 0.0
```

This sets the coordinate specification to spherical coordinates and then places an object called "Fir Tree" at a latitude of 43 degrees, 10 minutes and 0.0 seconds north, 18 degrees, 0 minutes and 0.0 seconds east on a mass called "Earth". To specify southerly or westerly coordinates, use negative values. Once placed onto the mass called "Earth" such that the object is upright for that latitude and longitude, it is rotated by 10 degrees. A Y rotation would rotate the tree in the axis of its trunk. Note that the tree’s origin is being placed 8 feet from the surface of the mass.

```
DEFINE COORDINATES MAP/MASS="Moon"
PLACE "Mountain" 20.0 0.0 10000.0  0.0 0.0 0.0
```

This sets the coordinate specification to map coordinates and then places an object called "Mountain" at a distance 20 feet east and

10,000 feet south of the zero latitude and zero longitude point of a mass called "Moon". The object is not rotated, and the mountain object is placed such that its origin is in contact with the surface of the mass (an altitude of zero).

ENTRY

Specifies a simulation entry point for the world.

FORMAT

ENTRY *"entry-description" xpos ypos zpos xrot yrot zrot*

Qualifiers	Defaults
None	*None*

PARAMETERS

entry-description

Specifies a descriptive string about the entry point. For example, "Makalu Island, Runway 28". This is the string that appears in the Entry Points menu of the simulator configuration dialog box.

xpos ypos zpos

Specifies the location at which an aircraft using this entry point would appear at the beginning of an interactive simulation. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

xrot yrot zrot

Specifies the rotation that an aircraft using this entry point would have applied to it at the beginning of an interactive simulation. All rotations are counterclockwise from the positive axis of rotation.

DESCRIPTION

Use the ENTRY command to specify an entry point into the world. During an interactive simulation, all entry points of a world are presented to the user for selection of where to begin a simulation. Common usage of this command is to place entry points at the ends of runways. Entry points do not need to be located on the ground.

Note that if you do not provide an entry point near a significant location in a world (such as an airport or geographic feature), the user will be required to manually fly an aircraft from a nearby entry point that has been defined.

QUALIFIERS *None*

GEOMETRY LINE

Specifies the endpoints of a single line in an object.

FORMAT

GEOMETRY LINE *point-index1 point-index2*

Qualifiers	**Defaults**
None	*None*

PARAMETERS

point-index1 point-index2

the indices of the two points which make up the line.

DESCRIPTION

The GEOMETRY LINE command specifies the location of a single line in the currently open object's geometry. GEOMETRY LINE commands may only be used after a GEOMETRY LOD command has been specified. A maximum of 300 lines may be loaded into each level of detail for each object.

The color of the line is determined by the color of the object or subobject that contains the line. The color is specified by the GEOMETRY COLOR command. The order in which the point indices are specified is not significant, but the points must have been loaded before being referenced by a GEOMETRY LINE command. Other than that restriction, GEOMETRY POINT and GEOMETRY LINE commands may be mixed freely.

QUALIFIERS

None

EXAMPLES

```
GEOMETRY LINE 8 12
```

A single line is loaded into the most recently specified level of detail and is defined to be drawn from the eighth point loaded into the current level of detail to the twelfth point loaded into the current level of detail.

GEOMETRY LOD

Specifies a change in an object's geometry level of detail specification.

FORMAT

GEOMETRY LOD *level-of-detail-index min-angle-applies*

Qualifiers	Defaults
None	*None*

PARAMETERS

level-of-detail-index

the new geometry level of detail to be loaded with points and lines. Valid level of detail indices are 1 through 3.

min-angle-applies

Specifies an angle in degrees. This angle is used to determine when the level of detail specification should be used to display the object. The angle is computed by figuring out how wide the object appears to be to someone viewing it. The further the object appears to be, the smaller the angle. Careful tuning of this number will produce imperceptible changes between levels of detail as the object recedes into the distance or approaches the viewer.

DESCRIPTION

The GEOMETRY LOD command is used to specify the conditions under which a level of detail specification should be used to display the object. A level of detail specification is a geometric description of the object with some given level of complexity. LOD 1 is used for the greatest level of geometric complexity (most lines and points). LOD 3 is used for the least complicated geometry. FLIGHT also provides a fourth LOD, which merely results in the object not being displayed.

The rationale for levels of detail at all is so that as the object moves further and further away from the viewer, less and less complicated geometries can be used to display the object. This is an important performance consideration.

Each level of detail specification is prefixed by a GEOMETRY LOD command, followed by GEOMETRY LINE and GEOMETRY POINT commands. Further, each specification is a complete description of the geometry of the object which is totally unrelated from the descriptions of other levels of detail, except, of course, in that they all describe the same object. You do not have to specify anything more than 1 level of detail, but you must specify the GEOMETRY LOD command for that level of detail. When the visual size of the object goes below the minimum angle for that level of detail, the FLIGHT-supplied fourth level of detail will be used; the object will disappear.

QUALIFIERS *None*

EXAMPLES

```
GEOMETRY LOD 1 20.0

GEOMETRY POINT and GEOMETRY LINE commands for level of detail 1

GEOMETRY LOD 2 8.0

GEOMETRY POINT and GEOMETRY LINE commands for level of detail 2

GEOMETRY LOD 3 0.5
```

GEOMETRY POINT and GEOMETRY LINE commands for level of detail 3 In this example, FLIGHT would use the description outlined in the first level of detail while the object was visually greater than or equal to 5.0 degrees of angular width. The second level of detail would automatically be used when the object was visually reduced to between 5.0 and 1.0 degrees of width. The third would be used once the 1.0 degree threshhold had been crossed.

GEOMETRY POINT

Specifies the location of a single point in an object.

FORMAT

GEOMETRY POINT *(x y z)*

Qualifiers	Defaults
/XREF=("object-name", lod, point-index)	*/NOXREF*

PARAMETERS

(x y z)

The location of the point relative to the origin of the object. Do not specify this value if the /XREF qualifier is used.

DESCRIPTION

The GEOMETRY POINT command specifies the location of a single point in the currently open object's geometry. GEOMETRY POINT commands may only be used after a GEOMETRY LOD command has been specified.

Points are numbered in the order that they are loaded, starting at 1 and going up to a maximum of 300 points per each level of detail. So each level of detail may have up to 300 points, each numbered from 1 to 300. Points may be referenced in geometry line commands multiple times in a geometric description, but only within the same level of detail description.

Units are in feet. Forward is +X, up is +Y and right is +Z. This is a required orientation for the geometry if you want the object to fly forward. Further, you should describe the geometry such that the origin of the geometry is near the typical center of gravity for the object, since all rotation of the object in flight is around the geometry origin.

You may specify only the absolute x-y-z coordinate *or* the coordinate via the XREF qualifier.

QUALIFIERS

/XREF=("object-name", lod, point-index)

Specifies that the point should assume the coordinates of an instance of another world object. This is only permitted in objects which have the FILLER characteristic (see the CHARACTERISTICS command).

The object name is the name of an instance of a world object created via the PLACE command. An object instance is named via the PLACE command's /NAME qualifier. The lod argument indicates which level of detail the point index is taken from, and point index is the actual point to use for the cross reference.

EXAMPLES

```
GEOMETRY POINT 12.0 -6.66 0.11
```

A single point is loaded into the most recently specified level of detail for the currently open object. The point is defined to be 12.0 feet in front of, 6.66 feet below, and 0.11 feet to the right of the geometry center.

```
GEOMETRY POINT/XREF=("Duggan Airport", 1, 14)
```

A single point is loaded into the most recently specified level of detail for the currently open object. The point is defined to be the same world location as the 14th point of the first level of detail of the world object named "Duggan Airport". This name is taken from a PLACE command's /NAME qualifier, *not from an OPEN OBJECT name.*

GEOMETRY TRIANGLE

Creates a triangle polygon

FORMAT

GEOMETRY TRIANGLE *point-index1 point-index2 point-index3*

Qualifiers	**Defaults**
/ATTRIBUTES=attribute-list	*None.*

PARAMETERS

point-index1 point-index2 point-index3

Specifies the indices of the three points which make up the triangle.

DESCRIPTION

Use the GEOMETRY TRIANGLE command to create a triangle for the current level of detail in the currently open object. A maximum of 200 triangles may be created in each level of detail for each object.

The color of the triangle is determined by the color of the object or subobject that contains the triangle. The color is specified by the GEOMETRY COLOR command. The order in which the point indices are specified is not significant, but the points must have been loaded before being referenced by a GEOMETRY TRIANGLE command.

Specifying TERRAIN on the /ATTRIBUTE qualifier tells FLIGHT that objects will treat this polygon as the ground, regardless of its orientation. For example, if a mountain were completely described with triangle polygons with the TERRAIN attribute, that mountain would be impervious to any contact, but aircraft could be landed on it, and vehicles could be driven over it.

FLIGHT does not currently support display of solid objects. Instead, the wireframe description of the object is used. See the GEOMETRY LINE command for a description of how to describe that wireframe form.

QUALIFIERS

/ATTRIBUTES=attribute-list

Specifies the attributes of the triangle polygon. The possible attributes are shown in Table 2-3. If you specify more than one attribute, you must enclose the attribute list in parentheses.

Table 2-3 Triangle Polygon Attributes

Keyword	Description
TERRAIN	This polygon is treated as being impervious to all contact and is the mechanism for specifying the contours of the ground. Specifying terrain on triangles in a level of detail other than 1 is ignored.
WATER	This polygon is treated as being impervious to all contact and is the mechanism for specifying a water surface. Specifying water on triangles in a level of detail other than 1 is ignored.

When an object comes into contact with a triangle which has either the TERRAIN or WATER attribute set, it is only allowed to move across that triangle if it has the appropriate ’capability’ characteristic. For example, if the triangle has the WATER attribute, then objects attempting to move across the triangle must have the WATER_CAPABLE characteristic set. If it is not set, then the engines of the object will be rendered ineffective.

Note that if multiple attributes are set on a triangle, objects attempting to move across the triangle must have equivalent ’capability’ characteristics set.

EXAMPLES

```
GEOMETRY TRIANGLE 8 12 9/ATTRIBUTES=TERRAIN
```

A single triangle is created in the current level of detail for the currently open object. The triangle is defined to be fill the triangular area encompassed by the eight, twelfth, and ninth points in the current level of detail’s point list.

Because the TERRAIN attribute is specified, this polygon is treated as if it were the ground, and simulated objects interact with it appropriately. That is, aircraft can land on it, vehicles can drive over it and weapons explode on contact with it. At no time does contact with an object's TERRAIN polygons cause damage to it.

ILS

Specifies the placement of an ILS installation in the world

FORMAT

ILS *"station-id" xpos ypos zpos frequency azimuth width*

Qualifiers	**Defaults**
/(NO)BC	*None.*
/GS=(xpos, ypos, zpos, slope, height	*None.*

PARAMETERS

station-id

Specifies the identification string for the ILD installation.

xpos ypos zpos

Specifies placement the ILS installation. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

frequency

Specifies the transmission frequency of the ILS. Frequency values are expressed in hertz (Hz). For example, 110.3 MHz would be expressed as 110300000.

azimuth

Specifies the azimuth that the localizer signal is centered on (that is, opposite to the runway heading). Azimuth values are expressed in degrees, where zero is North, 90 is East, etc.

width

Specifies the width of the localizer signal (full left to right) which gives a full-scale needle deflection. Width values are expressed in degrees.

DESCRIPTION

Use the ILS command to place an Instrument Landing System installation in a world. These installations work in conjunction with appropriately-equipped VOR instruments, which may be added to any aircraft. Up to 10 ILS transmitters may be placed per world.

QUALIFIERS

/(NO)BC

Specifies that this installation's localizer transmitter provides back-course guidance information. Note that when on a back course approach in an aircraft, the glide slope signal (if present) is to be ignored.

/GS=(xpos, ypos, zpos, slope, height

Specifies that this installation includes a glide-slope transmitter. The slope value indicates the angle of the glide slope and is specified in degrees. The height value indicates the total height of the glide slope (full top to bottom) which gives a full-scale needle deflection.

INCLUDE

Includes a world source file into the current compiler stream.

FORMAT

INCLUDE *input-file-spec*

Qualifiers	**Defaults**
None	*None*

PARAMETERS

input-file-spec

Specifies the name of the source file to include into the current compiler stream. You must specify a complete file specification. No wildcards are allowed. The default file type used for an included file is DAT.

DESCRIPTION

The INCLUDE command takes the text contents of the specified file and includes it exactly as the compiler finds it. There is no restriction on the number of included levels permitted. That is, an included file may include other files using the INCLUDE command, and those file may, in turn, include additional files. The nesting of includes is indicated in the listing file produced by the compiler which is available using the LISTING qualifier with the FLIGHT/COMPILE=WORLD command. For a description of the world compiler command, see Section 2.1.

QUALIFIERS

None

MASS

Specifies a gravitational mass for the world.

FORMAT

MASS *weight xpos ypos zpos radius*

Qualifiers	**Defaults**
/ADM=value	*1.0*
/NAME="name"	*None.*
/ORIENTATION=<orientation-spec>	*See text.*
/CHARACTERISTICS=(char,char,...)	*/CHARACTERISTICS=WATER*

PARAMETERS

weight

Specifies the weight of the mass in pounds.

xpos ypos zpos

Specifies the location of the center of the mass. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

radius

Specifies the radius of the mass in feet. This radius is used to determine sea level for the purposes of air density calculations. ~~All contact with the surface of a mass is lethal, immediately destroying all forms of objects.~~

~~In order to have surfaces on which aircraft can land and vehicles can drive, you must specify surfaces separately with the GEOMETRY TRIANGLE/ATTRIBUTES=TERRAIN command.~~

DESCRIPTION

Use the MASS command to define a gravitational mass in the world. These gravitational masses are used for all gravitational force calculations. That is, if you do not specify any MASS definitions, then the world simulated is effectively in space. There will be no air and no gravity. Up to 64 masses may be specified per world.

In addition to providing a gravity and atmospheric source, FLIGHT draws the horizon line of all masses automatically. A mass seen at a great distance will appear as a circle. A large mass seen up close will appear to have a flat horizon.

QUALIFIERS

/ADM=value

Specifies a value which is used to calculate the air density due to this mass. Values greater than 1.0 cause the air to become denser, and the extent of the atmosphere greater. Values less than 1.0 reverse those effects. The default is 1.0, which is the atmosphere of the Earth. This gives one atmosphere at sea level and a vacuum at 150,000 feet above sea level.

/NAME="name"

Specifies the name of the mass. If not specified, this mass may not be reference by other objects in the world. See the DEFINE COORDINATES command for an example of where a mass may be referenced by name.

/ORIENTATION

Specifies a set of one or more orientation values which allow the designer to orient a mass such that its north pole is not pointing in the absolute +Y direction, and that its zero longitude point is not in the absolute +X direction.

Generally, this is only useful when multiple masses have been created and their relative orientations and placements are important.

Table 2-4 shows the keywords which may be used with the /ORIENTATION qualifier and their meanings.

Table 2-4 ORIENTATION Types

Keyword	Meaning
NORTH	Specifies a set of three rotations which orient the north pole of the mass. These rotations are applied to a base north pole which points in the +Y direction.
ZERO_LONGITUDE	Specifies a set of three rotations which orient the zero longitude great circle (the equator). Because the three rotations result in a point on the surface of the mass, this

point is also used for zero latitude. These rotations are applied to a base zero longitude which points in the +X direction.

/CHARACTERISTICS=(char, char, ...)

Specifies a set of one or more characteristics of the mass.

Table 2-5 shows the keywords which may be used with the /CHARACTERISTICS qualifier and their meanings.

Table 2-5 CHARACTERISTICS Keywords

Keyword	Meaning
TERRAIN	Specifies that the mass is terrain covered, requiring objects which attempt to move across its surface to be TERRAIN_CAPABLE.
WATER	Specifies that the mass is water covered, requiring objects which attempt to move across its surface to be WATER_CAPABLE.

Note that specifying both TERRAIN and WATER as characteristics of a mass means that only objects which have *both* TERRAIN_CAPABLE and WATER_CAPABLE characteristics will be able to successfully move across the mass's surface.

The default is /CHARACTERISTICS=WATER.

EXAMPLES

```
MASS "1.6013740E+25" 0.0 "-2.0902922E+07" 0.0 "-2.0902922E+07" -
    /NAME="Earth"/ORIENTATION=(NORTH=(0.0, 0.0, 23.0),ZERO=(0.0, 90.0, 0.0))
```

This example shows the environment for the Earth. Note that the center of the mass source is at a distance below the mathematical origin of the world equal to the radius of the mass (the Earth). The Air Density Modifier value was allowed to default to 1.0, which is by definition equal to Earth-normal conditions. The quotes on the numbers are required when using the exponent syntax.

The second line is a continuation of the first by using the dash on the end of the first line. It specifies that the name of the mass is the Earth, and that the north pole should be tilted 23 degrees clockwise as viewed from the +Z axis (from the right of the mass), and that the zero longitude and latitude point should be rotated 90 degrees clockwise as viewed from the +Y axis (above the mass).

In summary, the command defines a gravitational mass weighing $1.6013740 * 10^{25}$ pounds, with a radius of $2.0902922 * 10^{7}$ feet (3,958.88 mi), having the name "Earth", and its north rotated 23 degrees from straight up (matching the Earth's angle from a perpendicular to the plane of the ecliptic), with the zero longitude and latitude great circles meeting on the +Z side of the mass (in the absolute world coordinate system).

NDB

Specifies the placement of an NDB in the world

FORMAT

NDB *xpos ypos zpos frequency class "station-id"*

Qualifiers	Defaults
None	*None*

PARAMETERS

xpos ypos zpos

Specifies placement or an NDB transmitter. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

frequency

Specifies the transmission frequency of the NDB. Frequency values are expressed in hertz (Hz). For example, 320 KHz would be expressed as 320000.

class

Specifies the transmitter class of the NDB. Table 2-6 shows the possible class designations.

Table 2-6 NDB Class Designations

NDB Class	Range	Type
LMM	15nm	Locator Middle Marker
LOM	15nm	Locator Outer Marker
MH	25nm	Homing Beacon
H	50nm	Homing Beacon
HH	75nm	Homing Beacon

station-id

Specifies the identification string for the NDB transmitter. This string is displayed in aircraft ADF-type instruments' station id readout when the instrument is picking up the signal from this NDB. The station-id string must be exactly three characters long (enclose the string in quotes and add spaces if necessary).

DESCRIPTION

Use the NDB command to add a Non-Directional Beacon to the world. These beacons work in conjunction with the ADF instrument which may be added to any aircraft. Up to 50 NDBs may be placed per world.

QUALIFIERS

None

OPEN OBJECT

Begins a world object definition.

FORMAT

OPEN OBJECT

Qualifiers	Defaults
None	*None*

PARAMETERS

None

DESCRIPTION

Use the OPEN command to begin the definition of a new world object type. Note that this command does not add a physical object to a world environment, but rather defines a type of object for placement in the world by the PLACE command. You may only have one world object open for definition at a time.

An object definition is completed using the CLOSE command. Table 2-7 lists the commands which are used to describe a world object.

Table 2-7 Commands Used to Describe World Objects

Commands	
OPEN OBJECT	CHARACTERISTICS
GEOMETRY	SIGNATURE
VOLUME	CLOSE OBJECT

QUALIFIERS

None

OPEN SUBOBJECT

Begins a subobject definition.

FORMAT

OPEN SUBOBJECT="subobject-name"

PARAMETERS

None

DESCRIPTION

The OPEN SUBOBJECT command begins a new subobject definition. The CLOSE SUBOBJECT command indicates the completion of that subobject definition. Commands describing the attributes of the subobject should be placed between the OPEN SUBOBJECT and CLOSE SUBOBJECT commands.

For more information about subobjects, see the description of the SUBOBJECT command in this chapter.

Subobjects are given names purely as a convenience to the user. No portion of FLIGHT makes use of world subobject names.

Up to 255 subobjects may be specified per parent object, and subobjects may be nested up to 16 levels deep. That is, a world object may have may have 255 subobjects, and each of those may have 255 subobjects and so on, for 16 levels. Many thousands of subobjects may be created for a single world object. However, that would surely slow the simulator to a crawl.

Table 2-8 lists the commands used to describe a subobject.

Table 2-8 Commands Applicable to Subobjects

Commands		
GEOMETRY	POSITIONING	SUBOBJECT
VOLUME		

QUALIFIERS *None*

EXAMPLES

```
OPEN OBJECT="Building Complex"
.
.
.
   OPEN SUBOBJECT = "Building 1"
   .
   .
   .
      OPEN SUBOBJECT = "Main Entrance"
      .
      .
      .
      CLOSE SUBOBJECT
      OPEN SUBOBJECT = "3rd Floor Balcony"
      .
      .
      .
      CLOSE SUBOBJECT
   .
   .
   .
   CLOSE SUBOBJECT
.
.
.
CLOSE OBJECT
```

The OPEN OBJECT command begins the description of an object called "Building Complex". Within the description, a subobject is opened by use of the OPEN SUBOBJECT command to begin the description of some component of the complex.

While that subobject is still open, two more subobjects are created. Because they are created within the confines of the "Building 1" subobject, they are considered subobjects of "Building 1". If "Building 1" is destroyed, so are all its subobjects.

Each subobject is closed as needed by the CLOSE SUBOBJECT command, and finally the object itself is closed by the CLOSE OBJECT command.

The indentation on the OPEN SUBOBJECT and CLOSE SUBOBJECT command are permitted, but not required.

PLACE

Specifies the placement of a world object.

FORMAT

PLACE *"object_name" xpos ypos zpos xrot yrot zrot*

Qualifiers	**Defaults**
/MAGNIFICATION=magnification	*/MAGNIFICATION=1.0*
/NAME="name"	*None.*

PARAMETERS

object_name

Specifies the name of the object to place in the world. This object name must match the name of a previously-defined world object.

xpos ypos zpos

Specifies the positioning of the origin of the world object. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

xrot yrot zrot

Specifies rotation of the object from its rest (design) orientation. All rotations are counterclockwise from the positive axis of rotation.

DESCRIPTION

Use the PLACE command to add objects to the world environment. The OPEN command is used to originally define objects to be placed into a world. Objects, once defined, may be placed many times at different locations, orientations and sizes.

QUALIFIERS

/MAGNIFICATION=magnification

Specifies magnification of the object. A magnification value of 2.0 would double the size of the object, while a value of 0.1 would reduce the object size to one-tenth its designed size. This resizing is only applied to the current placement and does not affect other placements of the same object.

/NAME="name"

Specifies the name of the instance of the object being placed. This name is only used by objects with the FILLER attribute, which need to reference other object's points via the /XREF qualifier. Using that qualifier, an object instance's name must be supplied. The name required is assigned using this qualifier.

POSITIONING

Specifies information about the placement of subobject geometry relative to its parent.

FORMAT

POSITIONING *x-position y-position z-position x-rotation y-rotation z-rotation*

Qualifiers	**Defaults**
None.	*None.*

PARAMETERS

x-position y-position z-position

Specifies the position of a subobject relative to the subobject's parent object's geometry origin. Because subobjects may be nested 4 levels, the parent of the subobject may be another subobject.

x-rotation y-rotation z-rotation

Specifies the rotation of a subobject relative to the parent object's rotation.

DESCRIPTION

The POSITIONING command is used to specify where the subobject geometry should be in relation to the parent object or subobject. Both position and orientation must be specified.

QUALIFIERS

None

SIGNATURE RADAR

Specifies information about the radar detectability of a world object.

FORMAT

SIGNATURE RADAR *front,top,side*

Qualifiers	Defaults
None	*None*

PARAMETERS

front,top,side

Specifies the radar cross section in square feet of the object as seen from the front, top and side. When viewed from an angle, FLIGHT interpolates the three values to determine the appropriate value.

DESCRIPTION

The SIGNATURE RADAR command specifies information about the radar detection cross section of the currently open object or subobject. In order for a subobject to have a radar signature, all of its parent objects must have the SIGNATURE RADAR command specified, even if no cross section value is specified.

Radar cross sections are additive. So if a subobject contributes 30 square feet and the root object contributes 10, the total radar cross section of the object is 40 square feet.

QUALIFIERS

None

EXAMPLES

```
SIGNATURE RADAR 10,100,30
```

The currently open object or subobject is declared to have a radar cross section of 10 square feet when viewed from the front, 100 square feet when viewed from above and 30 square feet when viewed from the side.

VIEWPOINT

Specifies an alternate viewpoint for the world

FORMAT

VIEWPOINT *xpos ypos zpos "description"*

Qualifiers	**Defaults**
None	*None*

PARAMETERS

xpos ypos zpos

Specifies the location of the viewpoint. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

description

Specifies a descriptive string about the viewpoint. For example, "Norwood Tower". This is the string that appears in the title bar of the world view window during an interactive flight simulation.

DESCRIPTION

Use the VIEWPOINT command to define a viewing location which is stationary in the world. While flying in an aircraft, the user has the ability to view his aircraft from any of the alternate viewpoints placed into the world. Viewpoints do not have names but are numbered according to their order of definition, starting at one. Up to 16 alternate viewpoints may be defined per world.

QUALIFIERS

None

VOLUME

Specifies information about the spherical volumes that an object occupies.

FORMAT

VOLUME *volume-radius center-x center-y center-z volume-damage*

Qualifiers	**Defaults**
/DESTROY=destroy-keyword	*/DESTROY=SELFDESTRUCT*

PARAMETERS

volume-radius

Specifies the radius of the spherical volume being added to the object's volume list.

center-x center-y center-z

Specifies the location of the center of the spherical volume being added to the object's volume list. This value is specified in relation to the object's center.

volume-damage

Specifies the amount of damage that this volume can accrue before being considered destroyed. When a volume is destroyed, it's destruction effect is performed. This effect is specified by the /DESTROY qualifier. When two objects collide, it is actually their volumes which collide. When two volumes collide, the one with lesser remaining damage will be destroyed. Therefore, weapons such as bullets should have low volume damage levels, whereas the tail section of an aircraft might have a relatively high damage level. All damage values are unitless and arbitrarily assigned. What is important is how the damage values assigned relate to each other.

DESCRIPTION

The VOLUME command is used to specify what space (relative to the object's geometry center) is actually occupied by the object currently being defined. This is used for collision detection checks. Up to 20 volumes may be specified per object, and volumes are unrelated to levels of detail, which are used in geometry specification. The order in which volumes are defined is of no importance. You may choose not to specify any volumes, in which case a single volume is defined for you which encompasses all the points of the geometry, centered at the object's geometry origin with the SELF-DESTRUCT destruction effect.

QUALIFIERS

/DESTROY=destroy-keyword

Specifies what should happen if this volume is destroyed due to collision damage. This allows you to bind destruction of the volume to changes in the object's performance and capabilities.

Table 2-9 lists the possible destruction keywords permitted and their meaning.

Table 2-9 Volume Destruction Keywords

Keyword	Description
SELFDESTRUCT	Indicates that the entire object that this volume is within should be destroyed. This is regardless of how many levels of subobjects it is nested within.
LOCALDESTRUCT	Indicates that the current level object should be destroyed. So if this volume is part of a subobject, that subobject and all of its subobjects will be destroyed. If this volume is part of a root object, then the entire object will be destroyed.

EXAMPLES

```
VOLUME 3.0 -10.0 0.5 2.0 20.0/DESTROY=SELFDESTRUCT
```

In this example, the volume has a radius of 3.0 feet. A total of 20 damage points have been allotted to the volume before it is considered destroyed. When it is destroyed, the entire object to which this volume is a part will be destroyed as well due to the presence of the SELFDESTRUCT destruction keyword.

VOR

Specifies the placement of a VOR in the world

FORMAT

VOR *xpos ypos zpos frequency class "station-id"*

Qualifiers	Defaults
None	*None*

PARAMETERS

xpos ypos zpos

Specifies placement the VOR transmitter. See the DEFINE COORDINATES command for a description of how FLIGHT interprets this value.

frequency

Specifies the transmission frequency of the VOR. Frequency values are expressed in hertz (Hz). For example, 117.7 MHz would be expressed as 117700000.

class

Specifies the transmitter class of the VOR. Table 2-10 shows the possible class designations. Refer to the Airman's Information Manual for a complete description of VOR Standard Service Volumes; in particular the H class has a complex coverage volume.

Table 2-10 VOR Class Designations

VOR Class	Range	Flight Level	Type
T	25nm	to FL120	Terminal Service
L	40nm	to FL180	Low Altitude Service
H	130nm	FL180 to FL450	High Altitude Service

station-id

Specifies the identification string for the VOR transmitter. This string is displayed only by the world previewer.

DESCRIPTION

Use the VOR command to place a Very-high-frequency Omni-Range transmitter in the world. These transmitters work in conjunction with the VOR instrument, which may be added to any aircraft. Up to 32 VOR transmitters may be placed per world.

QUALIFIERS

None

3 The World Editor

The FLIGHT world editor is an interactive tool which is intended to aid in the development of worlds. It provides only the most rudimentary capabilities. In fact, *editor* is something of a misnomer, because you cannot modify any component of a world or a world object.

The world editor only provides access to the objects of a world, not the placements, masses, entry points, viewpoints, or navigational information.

3.1 DCL Command

The following describes the invocation of the world compiler.

FLIGHT/EDIT=WORLD

Invokes FLIGHT's world editor..

FORMAT

FLIGHT/EDIT=WORLD *input-filespec*

Qualifiers	Defaults
None	*None*

PARAMETERS

input-filespec

Specifies the world to be edited. Note that wildcards are not allowed in the input file specification. The default file type for this parameter is WORLD. The default locations of world files is SYS$LIBRARY. If the world you want to edit is not located in that directory, you must specify a full path in the *input-filespec* parameter.

DESCRIPTION

The FLIGHT/EDIT=WORLD command starts FLIGHT's world editor. When the editor is invoked, messages will be output listing the total number of objects, object placements, VOR, ILS and NDB installations in the world file. When the editor is finished reading the world file, a dialog is presented to the user.

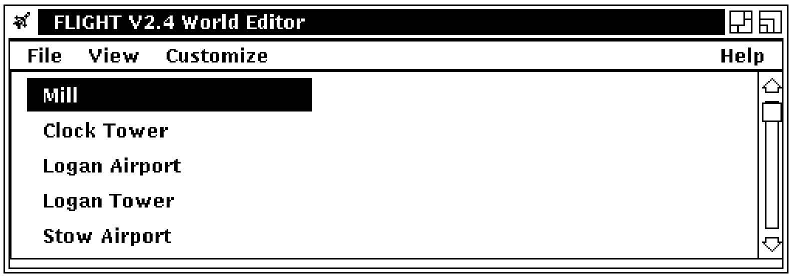
QUALIFIERS

None

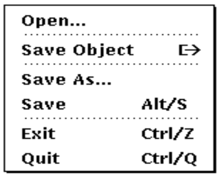
3.2 Interactive Use

The following figure shows the dialog displayed when the world editor is invoked. It allows selection from among the objects in the world, and selection from the various options available through the *File*, *View* and *Customize* menu options.

Figure 3-1 World Editor Dialog



3.3 The File Menu



The *File* menu allows you to

- Open a new hangar for editing. No warning is given if you have modified the currently open hangar.
- Save the source description of the currently selected object to a file
- Save a pictorial representation of the components of the currently selected object in *PostScript*™ format.
- Save the world being edited to a file that you name
- Save the world being edited to the file it came from
- Exit the world editor, saving any changes you have made
- Quit the world editor. This does not save the current world.

3.3.1 Open Hangar

Select this option to open a new hangar for editing. When selected, a file selection dialog box will appear, prompting you for the name of the new hangar. Once loaded, the main window list of objects will be updated to reflect the contents of the new hangar.

3.3.2 Save Object as Source

Select this option to create a source file describing the currently selected object. When selected, a file selection box will be presented to you, allowing you to specify the name of the file that FLIGHT should write the source description into.

The source description will be without any comments, and will only describe the actual object selected, not including any placement commands relating to it.

3.3.3 Save Object as PostScript™

Select this option to create a file containing a *PostScript*™ description of the components of the object. When selected, a file selection box will be presented to you, allowing you to specify the name of the file that FLIGHT should write the *PostScript*™ desription into.

Each page of the output file will show three views of a component of the selected object. The views displayed are top, front and rights side views. Each point of the object is numbered, and the name of the object is shown in the upper right corner of the page.

3.3.4 Save World to New File

Select this option to create a new world file containing any changes made so far. When selected, a file selection box will be presented to you, allowing you to specify the name of the file that FLIGHT should write the world file into.

3.3.5 Save World

Select this option to create a new version of a world file containing any changes made so far. When selected, FLIGHT writes the world contents into the file that was originally edited.

3.3.6 Exit

Select this option to exit the world editor and save the current world contents to a new version of the file being edited. When selected, FLIGHT writes out the world, then exits.

3.3.7 Quit

Select ths option to quit the world editor. When selected, FLIGHT immediately exits without saving any changes you have made.

3.4 The View Menu

Geometry... Alt/G

The *View* menu allows you to

- View the geometry of the currently selected object.

3.5 View Object Geometry

Select this option to view the geometry of the currently selected object. When selected, a window is created with the geometry of the current object displayed in it. You may use the mouse to view the geometry from a variety of angles. In general, the front of the aircraft will attempt to keep pointing at the mouse pointer.

In addition, the following keys are defined to allow you to control other aspects of viewing.

Table 3-1 Viewer Keyboard

Key	Description
Next Screen	Zooms in by 3X
Prev Screen	Zooms out by 3X
Up Arrow	Zooms in by 0.3X
Down Arrow	Zooms out by 0.3X
MB1	(Mouse Button 1) Locks the view at the current orientation. Another press of MB1 causes the view to unlock and follow the mouse pointer again.
F6	Take a PostScript snapshot of the current picture and place it into the current film canister.
F8	Quit the geomtry viewer. This is identival to the exit function.
F10	Exit the geometry viewer.

Note that in the lower left corner of the viewer window is a numeric display. This display indicates the current view angle to the object in the window. It can be useful for tuning the level of detail transitions of the object.

3.6 The Customize Menu

Film Canister...
Use Last Saved Settings
Use System Defaults
Save Current Settings

The *Customize* menu allows you to

- Specify the name of the file which should be used as your film anister for storing snapshots taken during the editing session.
- Reload the last set of preferences which you saved to disk.
- Set your preferences to the default FLIGHT preferences.
- Save your current preferences to disk.

3.6.1 Film Canister

Select this option to specify a new film canister filename. The film canister file is written to whenever a snapshot is taken in FLIGHT. In the world editor, this may be done in the geometry viewer.

3.6.2 Use Last Saved Settings

Select this option to reestablish the preferences which you last saved to disk.

3.6.3 Use System Defaults

Select this option to set the preferences for the current setting to the defaults defined by FLIGHT. If you had no preferences saved to disk prior to starting the world editor, the defaults would be in effect.

3.6.4 Save Current Settings

Select this option to save your preferences set through the *Customize* menu to disk. The file created is DECW$USER_DEFAULTS:FLT$DEFAULTS.DAT