APPENDIX III – CORRECTIONS

This appendix contains corrections to this ICD. Changes and additions are indicated by highlighted text. Deletions are indicated by ~~struckthrough text~~. These pages supercede the corresponding sections within the document. Changes included here will be incorporated into the next revision of the ICD.

This appendix will be updated periodically. Visit the CIGI web site (http://cigi.sourceforge.net) for the latest version.

The list below gives the sections and page numbers that contain the corrections, describes each change, and provides the revision date for each change.

Revision Date	**Section**	**Page**	**Page(s) Affected**	**Description**
15 Sept. 2004	4.1.2	221	33–43	The behavior or animations has been clarified for the Stop state of the *Animation State* parameter. Table 1 has been corrected to show the proper behavior for this state. The behavior of the Stop state for the *Animation State* parameter has been clarified in Table 2. The description for the *Entity Type* parameter has also been modified to address changing of an existing entity's type. Additional minor revisions.
16 Nov. 2006	3.3.3	232	25	The Figure 20 and text have been clarified to emphasize that a submodel coordinate system may be arbitrarily rotated within the model.

4.1.2 Entity Control

The **Entity Control** packet is used to control position, attitude, and other attributes describing an entity's state. This packet applies to all entities in the simulation, including the Ownship.

Each entity is identified by a unique identifier called the Entity ID. When the Host sends an **Entity Control** packet to the IG, the IG sets the state of the entity object corresponding to the value of the *Entity ID* parameter. If the specified entity does not exist, the IG will create it.

When the IG creates an entity, it makes a copy of the geometry corresponding to the value of the *Entity Type* parameter. This copy exists as a unique and independent tree within the scene graph; therefore, any operations that modify an entity's tree (e.g., part articulations) affect only that entity and its children.

Figure 25 illustrates unique Entity IDs being assigned to multiple entities of the same type. The number assignments in this example are hypothetical.

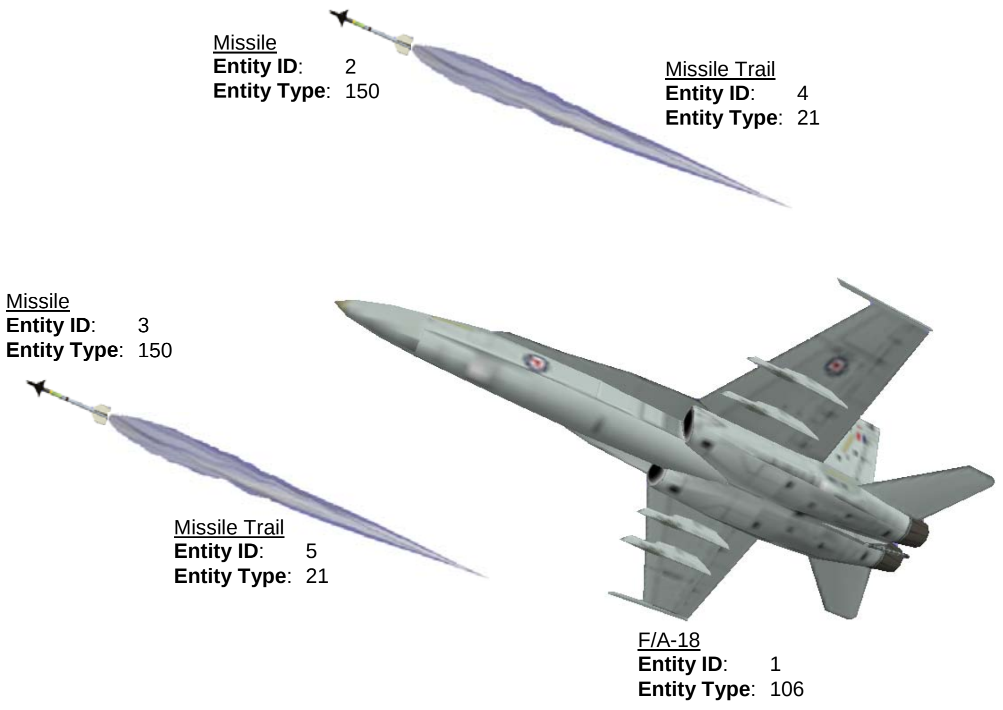

Figure 25 – Example of Entity Definitions

Entities can be attached to one another in a hierarchical relationship. In such a hierarchy, a child entity's position is specified relative to its parent's coordinate system. The Host needs only to control the parent entity in order to move all lower entities in the hierarchy as a group. No explicit manipulation of a child entity is necessary unless its position and attitude change with respect to its parent.

In the example above, the missiles and missile trails could be maneuvered in one of two ways. First, each missile and missile trail could be controlled uniquely, requiring the host to provide position and attitude data for each of the four entities. The simpler and typically preferred way would be to establish a parent-child relationship for each missile and missile trail pair. Each missile could be attached to the aircraft until launched, at which time the missile would be detached from its parent and promoted to a top-level entity. Each top-level missile would then possess its own independent hierarchy, which would be manipulated independently from the aircraft. Figure 26 illustrates the parent-child hierarchy before and after a missile is launched:

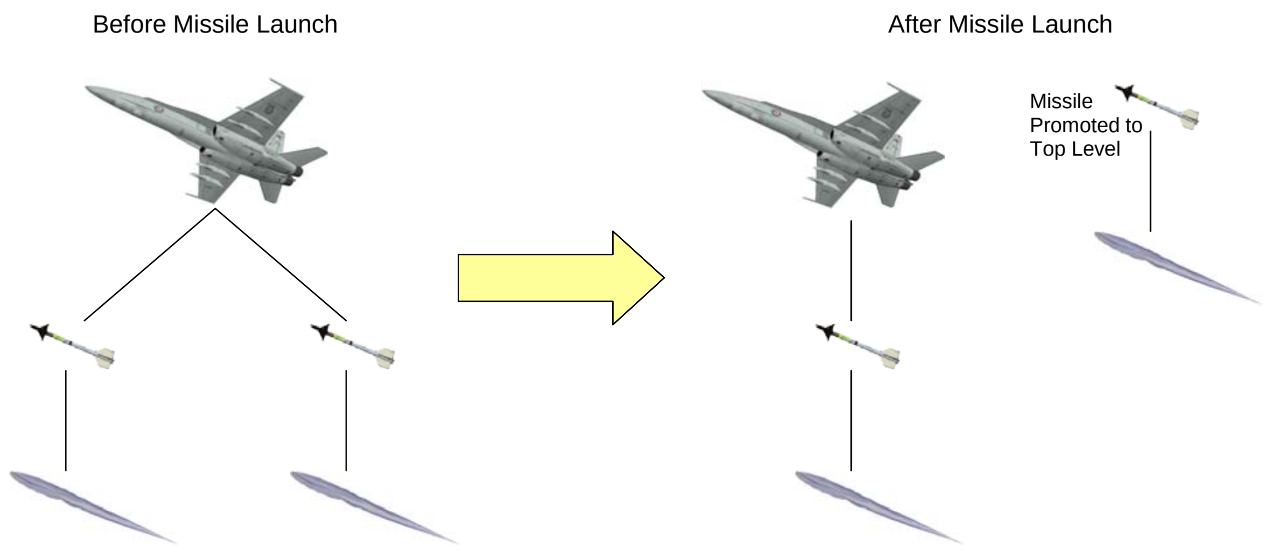

Figure 26 – Example of Child Entity Detachment

The *Attach State* parameter of the **Entity Control** packet determines whether an entity is attached to a parent. If this parameter is set to Attach (1), the entity is attached to the entity specified by the *Parent ID* parameter.

The *Entity State* field is used to control when an entity is visible and when its geometry is loaded and unloaded. When an entity is created, the *Entity State* field can be set to Active to specify that the entity should be added to the scene as soon as the model geometry is loaded. The entity and any children can be made invisible at any time by setting *Entity State* to Inactive/Standby. When the entity is no longer needed, *Entity State* can be set to Destroyed to direct the IG to unload the geometry and free any memory allocated for the entity. Any children attached to the entity are also destroyed.

Models can be preloaded to increase the speed at which they can be initially displayed. For example, when an aircraft fires a missile, a new entity would need to be created for that missile. Unless the missile geometry is cached, the IG must load the model from its hard disk. Because of its tremendous speed, the missile might fly a significant distance (and possibly beyond visual range) before the disk I/O can be completed. By preloading the entity, the geometry can already exist in memory and be instantly activated within the scene graph when needed. To accomplish this, the *Entity State* flag could be set to Inactive/Standby when the missile is created. Later, when the missile is needed, an **Entity Control** packet for that entity would be sent containing the proper positional data and with the *Entity State* flag set to Active.

An entity can also be made invisible by setting the *Alpha* parameter to zero (0). This parameter specifies an alpha value to be applied to the entity's geometry. The *Inherit Alpha* parameter indicates whether a child entity's alpha value is combined with that of its parent. For example, a missile attached to the wing of an aircraft would typically be made invisible when the aircraft is destroyed, so its *Inherit Alpha* attribute would be set to Inherited (1). An explosion or similar animation attached to that aircraft, however, would typically linger after the aircraft's destruction, so its *Inherit Alpha* attribute would be set to Not Inherited (0).

Note that setting the *Entity State* parameter to Inactive/Standby is not equivalent to setting the *Alpha* parameter to zero (0). The *Entity State* parameter enables or disables the entity geometry in the scene graph. The entity would not be included in line of sight and collision testing, nor would any transformations be applied. Any children would also be disabled. The *Alpha* parameter, on the other hand, merely affects the opacity of the specified entity.

The positions of top-level entities (i.e., those entities that are not children) are always specified as a geodetic latitude, longitude, and altitude (see Section 3.3.1.1). The positions of child entities are always specified with respect to the parents' NED body coordinate system (see Section 3.3.2).

In certain instances, it is desirable for the IG to “clamp” the entity to the surface of the terrain. This can be used as an alternative to using HOT requests and responses to determine ground elevation and slope below the entity. If the *Ground/Ocean Clamp* parameter is set to Non-Conformal (1) or Conformal (2), the *Altitude* parameter specifies an offset above the ground or sea surface height. This is useful for specifying the vertical distance from an automobile’s reference point to its wheels, for instance, or from a ship’s reference point to its waterline. Similarly, *Roll* and *Pitch* specify rotational offsets if ground or ocean clamping is enabled.

The *Animation State* parameter is used to control the playback state of animation entities. To start the animation sequence, the Host will send an **Entity Control** packet with its *Entity State* set to Active and its *Animation State* parameter to either Play or Resume. The Host may explicitly stop the animation at any time by setting the *Animation State* to Stop. Setting the parameter to Pause freezes the animation sequence at the current frame. Setting the parameter to Resume in a subsequent frame will resume the animation from its paused state; setting it to Play will play the animation again from its initial state. Setting *Animation State* to Play during playback will restart the animation.

Note that setting the *Animation State* parameter to Stop will have different effects on different types of animations. Frame-based animations may simply stop, or begin a termination sequence if such a sequence has been defined, at the current frame. Emitter-based animations (e.g., missile trails and particle systems) will stop producing new particles or segments; however, existing particles or segments will continue to decay normally. Stopping an animation does not implicitly remove it from the scene unless the *Entity State* parameter is set to Inactive/Standby or Destroyed.

The following table summarizes the animation behavior:

Table 5 – Animation State Summary

Current State	New *Animation State* Value	Effect
Stopped	Stop	None
	Pause	None
	Play	Plays from beginning
	Continue	Plays from beginning
Playing	Stop	Stops at current frame; Stops emitting particles/segments
	Pause	Pauses at current frame; Freezes all particles
	Play	Restarts from beginning
	Continue	No action, continues playing
Paused	Stop	Stops
	Pause	None
	Play	Plays from beginning
	Continue	Plays from current frame

If an animation has been built with a limited duration, and if the *Animation Loop Mode* parameter is set to One-Shot, the animation will stop automatically upon its completion. The IG will indicate this condition by sending an **Animation Stop Notification** packet (Section 4.2.15) to the Host. If the *Animation Loop Mode* parameter is set to Loop, the animation will immediately restart from the beginning and no **Animation Stop Notification** packet will be sent.

Once an **Entity Control** packet describing an entity is sent to the IG, the state of that entity will not change until another **Entity Control** packet specifying that entity ID is received. For example, packets describing the Ownship and a wingman may be sent every frame to indicate continuous positional changes, while a packet describing an inactive SAM site may be sent once during mission initialization.

The contents of the **Entity Control** packet are as follows:

7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Packet ID = 2								Packet Size = 48								*Entity ID*															
*6	*5		*4	*3	*2	*1		Reserved				*9		*8	*7	*Alpha*								Reserved							
Entity Type																*Parent ID*															
Roll																															
Pitch																															
Yaw																															
Latitude/X Offset																															
Longitude/Y Offset																															
Altitude/Z Offset																															

*1	*Entity State*
*2	*Attach State*
*3	*Collision Detection Request*
*4	*Inherit Alpha*
*5	*Ground/Ocean Clamp*
*6	Reserved
*7	*Animation Direction*
*8	*Animation Loop Mode*
*9	*Animation State*

Figure 27 – Entity Control Packet Structure

Table 6 defines each parameter's data type, units, and usage. If a default value and/or reference datum are applicable for a parameter, or if the domain differs from the range of values listed in Table 2, those values are also listed.

Table 6 – Entity Control Parameter Definitions

Parameter	Description
Packet ID Type: unsigned int8 Units: N/A Value: 2	This parameter identifies this data packet as the **Entity Control** packet. The value of this parameter must be 2.
Packet Size Type: unsigned int8 Units: Bytes Value: 48	This parameter indicates the number of bytes in this data packet. The value of this parameter must be 48.
Entity ID Type: unsigned int16 Units: N/A	This parameter specifies the entity to which this packet is applied. A value of zero (0) corresponds to the Ownship.

Parameter	Description
Entity State Type: unsigned 2-bit field Units: N/A Values: 0 Inactive/Standby 1 Active 2 Destroyed	This parameter specifies whether the entity should be active or destroyed. It may be set to one of the following values: **Inactive/Standby** – The entity is loaded, but its tree is not enabled in the scene graph. The entity is invisible, and no transformations are applied. Additionally, the entity is excluded from line of sight and collision testing. **Active** – The entity's tree is enabled in the scene graph. Transformations are applied to the entity and it is included in line of sight and collision testing. **Destroyed** – The entity's tree is removed from the scene graph. Any children are also destroyed. All other parameters in this packet are ignored.
Attach State Type: 1-bit field Units: N/A Values: 0 Detach 1 Attach	This parameter specifies whether the entity should be attached as a child to a parent. If this parameter is set to Detach (0), the entity becomes or remains a top-level (non-child) entity. The *Parent ID* parameter is ignored. The *Yaw*, *Pitch*, *Roll*, *Latitude*, *Longitude*, and *Altitude* parameters all specify the entity's position relative to the geodetic coordinate system (see Section 3.3.1). If this parameter is set to Attach (1), the entity becomes or remains attached to the entity specified by the *Parent ID* parameter. The parent must already exist, having been created in a prior frame or earlier in the current frame. The *Yaw*, *Pitch*, *Roll*, *X Offset*, *Y Offset*, and *Z Offset* parameters all specify the entity's position relative to the parent's coordinate system (see Section 3.3.2). This parameter may be changed for a given entity at any time. The attachment or detachment takes place immediately and remains in effect until changed with another **Entity Control** packet.

Parameter	Description
Collision Detection Request Type: 1-bit field Units: N/A Values: 0 No Request 1 Request	This parameter determines whether any collision detection segments and volumes associated with this entity are used as the source in collision testing. If this parameter is set to Request (1), each collision detection segment is tested for intersections with polygons not associated with this entity and each collision detection volume is tested pair-wise with every other volume that is not associated with the entity. If this parameter is set to No Request (0), any collision detection segments defined for the entity are ignored and any collision detection volumes are only tested (as the destination) against volumes defined for entities whose collision detection is enabled. See Sections 4.1.22 and 4.1.23 for details on creating collision detection segments and volumes, respectively.
Inherit Alpha Type: 1-bit field Units: N/A Values: 0 Not Inherited 1 Inherited	This parameter specifies whether the entity's alpha is combined with the apparent alpha of its parent. The following formula will be used to combine the alpha values: $$\alpha = \frac{\alpha_0 \cdot \alpha_p}{255}$$ where α is the apparent alpha of the child, α_0 is the explicit alpha of the child, and α_p is the apparent alpha of the parent. Note that a change in an entity's alpha affects the entities below it in the hierarchy if those entities inherit their parents' alphas. If *Attach State* is set to Detach (0), this parameter is ignored.

Parameter	Description
Ground/Ocean Clamp Type: unsigned 2-bit field Units: N/A Values: 0 No Clamp 1 Non-Conformal 2 Conformal	This parameter specifies whether the entity should be clamped to the ground or water surface. If *Attach State* is set to Attach (1), this parameter is ignored. **No Clamp** – The entity is not clamped. The *Altitude* parameter specifies the entity's height above Mean Sea Level. The *Pitch* and *Roll* parameters specify the entity's pitch and roll relative to the geodetic reference plane (Figure 17, page 22). **Non-Conformal** – The entity is clamped. The *Altitude* parameter specifies an offset above the terrain or sea level. The *Pitch* and *Roll* parameters specify the entity's pitch and roll relative to the geodetic reference plane (Figure 17, page 22). **Conformal** – The entity is clamped and its attitude conforms to the terrain. The *Altitude* parameter specifies an offset above the terrain or sea level. The *Pitch* and *Roll* parameters specify the entity's pitch and roll relative to the slope of the terrain or water.
Animation Direction Type: 1-bit field Units: N/A Values: 0 Forward 1 Backward	This parameter specifies the direction in which an animation plays. This parameter applies only when the value of the *Entity Type* parameter corresponds to an animation.
Animation Loop Mode Type: 1-bit field Units: N/A Values: 0 One-Shot 1 Continuous	This parameter specifies whether an animation should be a one-shot (i.e., should play once and stop) or should loop continuously.

Parameter	Description
Animation State Type: unsigned 2-bit field Units: N/A Values: 0 Stop 1 Pause 2 Play 3 Continue	This parameter specifies the state of an animation. This parameter applies only when the value of the *Entity Type* parameter corresponds to an animation. **Stop** – Stops the animation sequence. If the animation has a termination sequence or decay behavior, the animation will switch to that behavior. Has no effect if the animation is currently stopped. **Pause** – Pauses playback of an animation. The entity's geometry will remain visible, provided *Entity State* is set to Active. **Play** – Begins or restarts playback from the first animation frame. **Continue** – Continues a playing animation from the current frame of the animation sequence. If the animation is paused, playback is resumed from the current frame. If the animation is stopped, playback is restarted from the first frame of the sequence.
Alpha Type: unsigned int8 Units: N/A	This parameter specifies the explicit alpha to be applied to the entity's geometry. A value of zero (0) corresponds to fully transparent; a value of 255 corresponds to fully opaque.
Entity Type Type: unsigned int16 Units: N/A	This parameter specifies the type for the entity. A value of zero (0) indicates a "null" type with no associated geometry. Such an entity might be used to represent the Ownship or a floating camera. When changing entity types, the Host should first delete the entity by setting the *Entity State* parameter to Deactivate (2) and then recreate the entity and any children during a subsequent frame. If the specified type is undefined, the data packet will be disregarded.
Parent ID Type: unsigned int16 Units: N/A	This parameter specifies the parent for the entity. If the *Attach State* parameter is set to Detach (0), this parameter is ignored. The value of this parameter may be changed without first detaching the entity from its existing parent. If the specified parent entity is invalid, no change in the attachment will be made.

Parameter	Description
Roll Type: single float Units: degrees Values: -180.0 – 180.0 Datum: If *Attach State* = 0 and *Ground/Ocean Clamp* = 0 or 1: Geodetic reference coordinate system (see Section 3.3.1.2) If *Attach State* = 0 and *Ground/Ocean Clamp* = 2: Terrain surface polygon orientation If *Attach State* = 1: Entity reference coordinate system (see Section 3.3.2.2)	This parameter specifies the roll angle of the entity. For top-level entities for which *Ground/Ocean Clamp* is set to No Clamp (0) or Non-Conformal (1), this angle is measured from the reference plane described in Section 3.3.1.2. For top-level entities for which *Ground/Ocean Clamp* is enabled, this angle specifies an angular offset from the terrain surface polygon's orientation. For child entities, roll is measured from the parent entity's reference plane as described in Section 3.3.2.2.
Pitch Type: single float Units: degrees Values: -90.0 – 90.0 Datum: If *Attach State* = 0 and *Ground/Ocean Clamp* = 0 or 1: Geodetic reference plane (see Section 3.3.1.2) If *Attach State* = 0 and *Ground/Ocean Clamp* = 2: Terrain surface polygon orientation If *Attach State* = 1: Entity reference plane (see Section 3.3.2.2)	This parameter specifies the pitch angle of the entity. For top-level entities for which *Ground/Ocean Clamp* is set to No Clamp (0) or Non-Conformal (1), this angle is measured from the reference plane described in Section 3.3.1.2. For top-level entities for which *Ground/Ocean Clamp* is enabled, this angle specifies an angular offset from the terrain surface polygon's orientation. For child entities, pitch is measured with respect to the entity's reference plane as described in Section 3.3.2.2.

Parameter	Description
Yaw Type: single float Units: degrees Values: 0.0 – 360.0 Datum: If *Attach State* = 0: True North If *Attach State* = 1: Entity Reference Plane's +**X** axis	For top-level (non-child) entities, this parameter specifies the instantaneous heading of the entity. This angle is measured from a line parallel to the Prime Meridian as described in Section 3.3.1.2. For child entities, this parameter specifies yaw relative to the parent entity and is measured with respect to the +**X** axis of the entity's reference plane as described in Section 3.3.2.2.
Latitude (Top-Level Entities) Type: double float Units: degrees Values: -90 – 90 Datum: Equator	For top-level (non-child) entities, this parameter specifies the entity's geodetic latitude.
X Offset (Child Entities) Type: double float Units: meters Datum: Parent's reference point	For child entities, this parameter specifies the distance from the parent's reference point along its parent's **X** axis.
Longitude (Top-Level Entities) Type: double float Units: degrees Values: -180.0 – 180.0 Datum: Prime Meridian	For top-level (non-child) entities, this parameter specifies the entity's geodetic longitude.
Y Offset (Child Entities) Type: double float Units: meters Datum: Parent's reference point	For child entities, this parameter specifies the distance from the parent's reference point along its parent's **Y** axis.

Parameter	Description
Altitude (Top-Level Entities) Type: double float Units: meters Datum: If *Ground/Ocean Clamp* = 0: Mean Sea Level If *Ground/Ocean Clamp* = 1 or 2: Ground/sea surface elevation	For top-level (non-child) entities, this parameter specifies the entity's altitude. If the *Ground/Ocean Clamp* parameter is set to No Clamp (0), this value is the height above Mean Sea Level. If the *Ground/Ocean Clamp* parameter is set to Non-Conformal (1) or Conformal (2), this value specifies an offset above the terrain height or local sea level (see Section 4.1.13) at the entity's latitude and longitude.
Z Offset (Child Entities) Type: double float Units: meters Datum: Parent's reference point	For child entities, this parameter specifies the distance from the parent's reference point along its parent's **Z** axis.

3.3.3 Submodel Coordinate Systems

A submodel is a hierarchy of geometry nodes within a model (entity) for which a coordinate system is defined. Position and rotation of submodels are defined with respect to this coordinate system. Transformations performed on the coordinate system affect the submodel geometry as a whole. The order of rotation is as shown in Figure 18.

The submodel coordinate system may be defined with an arbitrary position and orientation relative to the entity model's coordinate system in a way that makes sense for the submodel. For example, a leading-edge flap might have a submodel coordinate system defined as shown in Figure 20a so that applying a positive pitch angle will rotate the flap above the wing. A trailing-edge flap's submodel coordinate system, however, might need to be rotated to achieve a positive pitch above the wing as shown in Figure 20b.

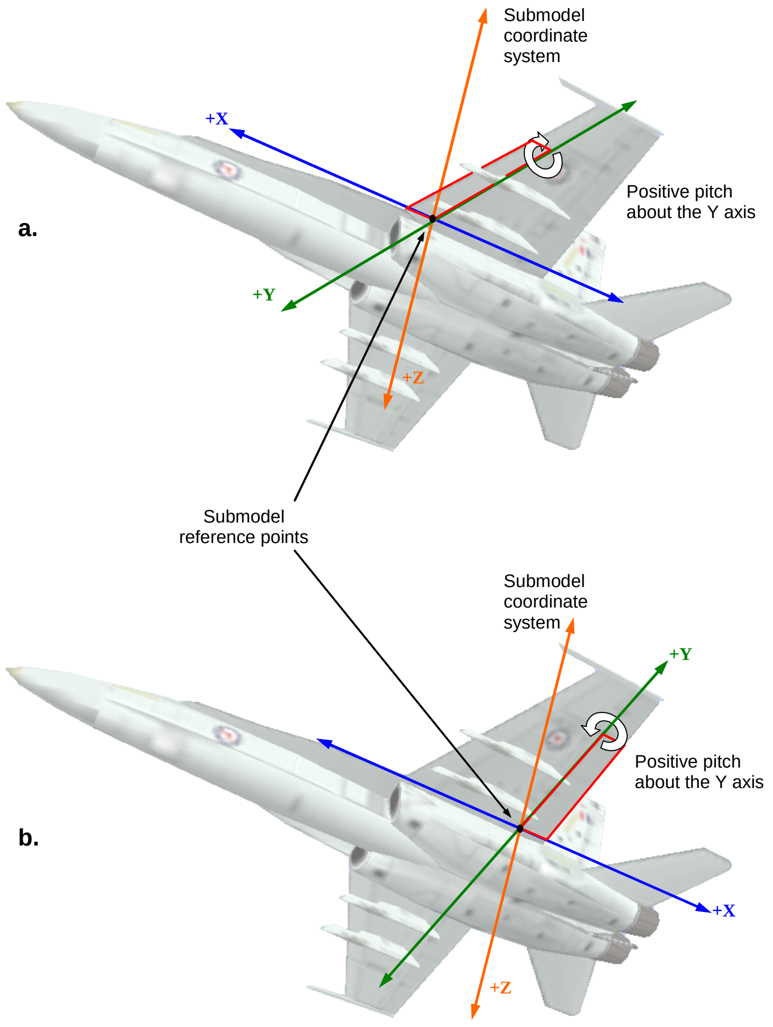
Submodel coordinate system
+X
Positive pitch about the Y axis
a.
+Y
+Z
Submodel reference points
Submodel coordinate system
+Y
Positive pitch about the Y axis
b.
+X
+Z

Figure 20 – Examples of Submodel Coordinate Systems

Note: Regardless of its orientation, the submodel coordinate system must be a right-handed coordinate system.

Rotations applied to a submodel are not cumulative. In other words, specifying a rotation and translation will override any previous values.

Section 4.1.6 describes the use of the **Articulated Part Control** packet in manipulating submodels.