

Chapter 1.

Advanced Blending Component

Component: *Blending

The Advanced Blending Component (ABL), in the abl directory, performs blends that are more complex than found in the standard ACIS Blending Component (BLND).

The sharp edges and vertices in models must often be replaced by faces in order to improve the appearance of a part, reduce stress points, or assist manufacture. This operation is called *blending*. For example, a woodworker may use a router to physically round the sharp edges of a table. This is analogous to blending edges in an ACIS model.

ABL offers the following types of blending operations:

- Constant radius blends
- Edge blends
- Elliptical blends
- Fixed width radius blends
- Variable radius blends
- Entity–entity blends
- Face–edge blends
- Rounded chamfer cross sections
- Blending in non–tangent environments
- A flexible range of radius functions and cross section shapes

ABL uses intuitive blend processing instruction sequencing to simplify the automatic transition between blend types, variable radius specifications, and other parameters.

Some *standard blending* functionality is embedded in ACIS (in the Blending Component). The application developer should be familiar with this standard blending functionality before working with the Advanced Blending Component (ABL). Refer to the *Blending Component Manual* for discussions of the standard blending concepts and terminology which are the foundation for the advanced blending discussions in this manual. This manual assumes that the reader is familiar with ACIS standard blending.

Note *This manual assumes that the reader is familiar with ACIS standard blending.*

This chapter is an overview of advanced blending concepts and the functionality provided with ABL. Differences between standard blending and advanced blending are highlighted where appropriate.

Blending Functionality

Topic: *Blending

In the most general sense, a *blend* is a transition surface between regions of a model. Standard blending, which is embedded within ACIS, offers blends to transition between faces, but ABL provides more options.

Advanced blending with ABL falls broadly into two categories:

Edge sequence-following . . . Augments the range of face-face geometries available to standard blending.

Entity-entity Provides a wholly general framework for blends between faces, edges, and vertices.

Note *The standard blending of the Blending Component provides some support for entity-entity blends to resolve certain difficult blending configurations. However, it does not provide the functionality for defining arbitrary entity-entity blends.*

In ABL, variable radius blends may be defined with a more flexible range of radius functions and cross section shapes, and blends between faces and edges (entity-entity blending) may be performed.

ABL handles many more geometric and topological cases than standard blending and also provides many new blend shapes, augmenting the round and chamfer blends available in standard blending. However, the geometry classes (e.g., new curve and surface types) needed for this functionality are implemented within the Kernel Component, because ACIS based applications that are not linked with ABL must be able to read, write, and evaluate (but not create) the geometries produced by advanced blending operations.

Blend Cross Section

Topic: *Blending

A blend surface between two entities has a characteristic shape which is defined by its cross section. The *cross section* is a line in the blend surface which connects the two entities being blended.

In standard blending, a blend surface bridges from one face to another face (e.g., does not blend from a face to an edge or an edge to an edge). The cross section shape of the blend is seen by the shortest line along the blend surface from one face to the other.

A *round*, or *circular*, cross section creates a pipe-shaped blend surface that is represented as a spline, in general, but under special circumstances is represented as a cylinder or a torus. In standard blending, the blend surface is tangent to each of the faces being blended, and the plane of the cross section is always perpendicular to both of the faces.

A *chamfer* cross section is a straight line which is controlled by offset distance parameters along each of the surfaces being blended. Under special circumstances, the blend surface is created as a plane or a cone.

In addition to the round and chamfer cross sections described above, the following cross section shapes are also available with ABL:

Elliptical Cross section shape is a rotated ellipse controlled by major and minor radius and rotation angle parameters.

Rounded chamfer Cross section is like a chamfer with a round bulge added. Like the chamfer, it is not tangent to the side surfaces. The "bulge" may be zero in order to create an ordinary chamfer.

Thumbweights The normally circular cross section can be given more or less bulge. The surface is still tangent to the side surfaces. Left and right thumbweights can be applied separately.

ABL provides geometry extensions that allow a variety of cross section shapes beyond those available in standard blending.

In standard blending, the plane of the cross section is always perpendicular to both of the faces being blended, but with advanced blending, the plane of the cross section may be lofted to align with a given orientation.

Cross Section Radius

Topic:

*Blending

All types of blend surface cross sections are controlled by *blend radius* parameter values. Varying the parameter values at different positions along the blend surface allows the size of the cross section to change.

In standard blending, the blend radius may be either a constant radius value or an implicit variable radius across the blend, which is specified by providing two radius values at the ends of a smooth edge sequence.

In addition to these two radius specification methods, advanced blending offers the following methods:

- Radius specified by a series of radius values at arbitrary positions along the blend
- Radius specified by a series of parameters along the defining curve for the blend and corresponding radius values
- Radius varies implicitly, depending on the angle between the side faces, to maintain a constant, or fixed, cross sectional width along the blend
- Radius specified by an arbitrary radius function

Standard and advanced blending perform automatic smoothing of consecutive two-end functions. Additionally, advanced blending offers automatic stabilizing of discrete-valued radius functions. If parameter values are specified at a few locations, the resulting blend forces the parameter values to change smoothly between the specified points.

Topology

Topic:

*Blending

Standard blending supports blends between both faces that are adjacent to the blend edge (the blend is tangent to both faces) and has some limited support for blending faces and edges that are remote from the edge.

Advanced blending supports blends between:

- Both faces that are adjacent to an edge
- Two nonadjacent faces
- A face and an edge
- A vertex and a face
- A vertex and an edge
- Two edges

Mitering

Topic:

*Blending

Mitering is available only in standard blending and the geometry extensions provided by ABL. For entity-entity blends, the effect of mitering is only possible when distinct sequences can be fixed separately.

Automatic Transitions

Topic:

*Blending

With entity-entity blending, blend processing instructions are placed on the model to direct the automatic transition between blend types: from face-face to edge-face or from edge-face to face-face. **This only applies to entity-entity blending.**

Single Stepping

Topic:

*Blending

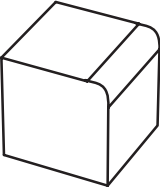
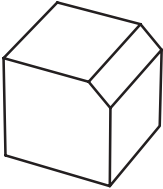
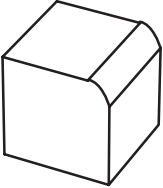
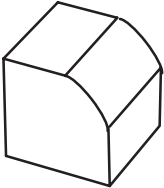
Blends using ABL may be produced in the same single-step manner as standard blending (using `api_init_blend_ss`, `api_do_one_blend_ss`, and `api_concl_blend_ss`).

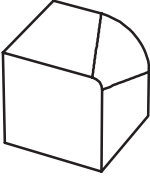
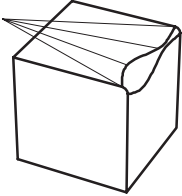
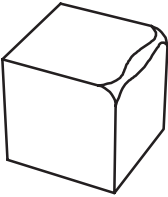
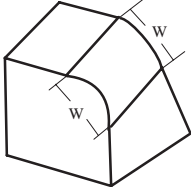
Blending Comparison

Topic: ^{*Blending}

The following tables summarize blending functionality and compare the standard blending functionality embedded in ACIS with the advanced blending functionality provided in ABL.

Table 1-1. Blending Comparison

Cross Section Geometry	Standard	Advanced
Circular 	Yes	Yes
Chamfer 	Yes	Yes
Elliptical 	No	Yes
Rounded chamfer 	No	Yes
Thumbweight designed	No	Yes

Cross Section Radius	Standard	Advanced
Constant radius	Yes	Yes
Radius values set at ends 	Yes	Yes
Radius values set at arbitrary positions 	No	Yes
Radius values set by arbitrary radius functions 	No	Yes
Radius varies implicitly to maintain constant cross sectional width 	No	Yes

Cross Section Orientation	Standard	Advanced
Perpendicular to blend surfaces	Yes	Yes
Lofted cross section <div data-bbox="539 318 770 493" data-label="Image"> </div>	No	Yes

Topology	Standard	Advanced
Face–face	Yes	Yes
Edge–face	Some	Yes
Edge–edge	Some	Yes
Vertex–face	No	Yes

Automatic Transitions	Standard	Advanced
Face to face (if faces smooth)	Yes	Yes
Edge to edge (if edges tangent)	Some	Yes
Face to edge	Some	Yes
Edge to face	Some	Yes
Edge to vertex	No	Yes
Vertex to edge	No	Yes

Blend Attributes

Topic: [*Blending, Attributes, *Blending](#)

Attributes are used to set up blends and blend sequences on edges and vertices. A blend is assigned by picking an edge or vertex and describing the blend characteristics, such as its radius. Attributes of class ATTRIB_BLEND are attached to the picked entities.

There are two types of blend attribute classes:

- Those derived from ATTRIB_BLEND, which are attached to the body to be blended, and indicate the blend operations requested
- Those derived from ATTRIB_BLINFO, which are attached to the blend sheet during blending to record its relationship with the blank body

Refer to Appendix C, *Derivations*, for the derivation of the blend attribute classes defined in ABL, and to the *Blending Component Manual* for information on the blending classes defined in standard blending (Blending Component).

Blend Processing

Topic: *Blending

All blend processing starts with the assignment of a variety of blend attributes to an ACIS body. Some of these attributes directly drive blend processing and might be termed *primary* blend attributes. Examples include a face–face or vertex blend, or an initial entity–entity blend. Other *secondary* blend attributes offer detailed guidance on the kind of blend to be made at particular places on the body.

Blend processing begins with a list of primary blend attributes, the attributes being considered one at a time in list order. Successful processing of a blend attribute creates one or more blend faces. At the same time, new primary blend attributes may be made and attached to entities of the blend body and are also placed in the list. Blend processing may also cause attributes already in the list—but not yet considered—to be removed from the list. If the processing of a blend attribute is unsuccessful, the attribute is moved to the end of the list for later reconsideration, or removed from the list.

In edge sequence–following blending, the user places blend attributes on a network of edges and vertices. These attributes form the initial list. In entity–entity blending, the user places a single primary and any number of secondary blend attributes on the body. The primary blend attribute constitutes the initial blend list.

Edge Sequence–Following Blends

Topic: *Blending

Edge sequence–following blends are driven by following a selected sequence of blended edges. These blends are essentially between pairs of faces only (face–face blends). They include all of the available standard blending functionality. ABL extends the standard edge sequence–following blend functionality by broadening the means of specifying the radius function and cross section shapes.

In addition to the constant radius and implicitly varying radius (specifying values at two ends) available in standard blending, the radius function may be specified in ABL by:

- A series of parameter values (along the defining curve for the edge sequence) and radius values
- A series of positions (near the defining curve for the sequence) and radius values
- A constant (fixed) cross sectional width to produce an implicitly varying radius function
- An explicitly constructed radius function

Surface extensions are supported by variable radius edge sequence-following blends.

Entity–Entity Blends

Topic: *Blending

The ability to specify completely arbitrary entity–entity blends is available only with ABL. Entity–entity blending can be used for blends other than face–face blends or for face–face blends between two faces that are not adjacent to one another (they may even be in disjoint shells). The standard blending of the Blending Component provides some support for entity–entity blends to resolve certain difficult blending configurations. However, it does not provide the functionality for defining arbitrary entity–entity blends.

These blends do not follow a sequence of blended edges. Instead, a pair of entities is given as a starting point and then the blend propagates itself as the ball rolls. The blend determines how to propagate itself by reading instructions from the entities which the spring curves intercept, rather than from a sequence of blended edges. Entity–entity blends may switch arbitrarily between face–face, edge–face, face–edge, edge–edge and even vertex–face and vertex–edge blends. The blend implicitly follows a sequence of entity pairs in the sense that new entity pairs are blended together as the blend propagates. These pairs are not known in advance, but are computed as the blend progresses.

The default behavior when the contact point encounters another topological entity is described in the following table:

Table 1-2. Default Behaviors

Currently On	Encountered	Default Action
Face	Smooth edge	Roll over the edge onto the next face
Face	Sharp edge	Cap the blend with a side or end cap
Face	Vertex with smooth face	Roll onto the smooth face
Face	Vertex without smooth face	Cap the blend
Edge	Face	Roll onto the face
Edge	Vertex with tangent edge	Roll onto the tangent edge
Edge	Vertex without tangent edge	Cap the blend

Instruction attributes may be explicitly placed on entities to force certain behavior. Because an entity–entity blend has no sequence of blended edges, the only place from which the instructions can be read is the entities that the contact points of the rolling ball actually encounter. The instruction used is either *roll on* or *cap*, depending on which behavior is to be forced. For example, a blend on a face that runs into a sharp edge may be forced to roll on to that edge rather than to form a cap. The roll on instruction would actually be placed on the encountered edge.

Instructions may have positions associated with them to allow different behaviors at different points along the blended edge. In this case, the instruction nearest to the point of intercept applies.

Both constant and variable radius entity–entity blends may be defined. Variable radius blends require a single smooth defining curve and may have arbitrary cross sections.

These blends do not interact with edge sequence–following blends. Only a single entity–entity sequence may be blended at a time, as entity–entity blends do not occur in miters or vertex blends.

Blend Transitions

Topic: *Blending

The transitions from one entity to another allowed with entity–entity blending depend on whether the blend is running along a face, along an edge, or around a vertex. In general, the blend comes off or away from the entity it is on, and it must compute what entity it will be blending next. The following sections describe the transition cases. In each case, the possible transitions are listed along with a description of how to obtain such a transition using instruction attributes. The default transitions do not need an instruction attribute.

Blends on Faces

Topic: Blending

Transitions onto new entities are only allowed when the spring curve of the blend leaves the face, having been inside it. Therefore, transitions are not supported at places at which the face boundary is merely grazed, either from the inside or outside. In some cases, an edge or vertex exists in a face only to support modeling functions, although it does not really need to be there in the pure boundary representation. This includes an edge in a periodic face that exists only to break the periodicity, such as the edge that splits the periodic face in a cylinder from one end cap to the other. In such a case, the edge is embedded in the face; i.e., the same face exists on both sides of the edge. Transitions onto new entities are not supported at such edges or vertices.

Table 1-3. Blends on Faces

Run Into	Transition	How to Obtain
smooth edge	roll onto next face	place rollon on face (default)
smooth edge	start capping	place cap on face
sharp edge	start capping	place cap on edge (default)
sharp edge	roll onto edge	place rollon on edge
smooth vertex	roll onto smooth face	place rollon on face (default)
smooth vertex	start capping	place cap on face
sharp vertex	start capping	place cap on vertex (default)

A *smooth vertex* is a vertex at which there is a smoothly joined face into which the blend will definitely be able to travel. This excludes vertices at which there is a smooth face and the direction of travel of the blend lies along one of that face's boundaries at the vertex.

A *sharp vertex* is a vertex at which there is definitely no smoothly connected face, or if there is, the direction of travel of the blend definitely excludes it from going into this face. This excludes vertices at which there is a smooth face and the direction of travel of the blend lies along one of that face's boundaries at the vertex.

Any blend leaving the face at a vertex where there is a smoothly connected face, but the direction of travel of the blend lies along one of that face's boundaries at that vertex, is not currently supported.

Blends on Edges

Topic: Blending

The majority of blends running against edges are moving either forward or backward along the edge, without doubling back on that edge. Such a blend may leave the edge either by reaching one of its vertices or by coming away from the edge onto one of the adjoining faces. In this case, the blend always has an *interior* face, which is the face on the side that will be "swallowed" when the blend is fixed, and an *exterior* face, which is the face on the side that will remain.

Table 1-4. Blends on Edges

Run Into	Transition	How to Obtain
smooth vertex	roll onto next edge	place rollon on edge (default)
smooth vertex	start capping	place cap on next edge
sharp vertex	start capping	place cap on vertex (default)
sharp vertex	roll onto vertex	place rollon on vertex

Run Into	Transition	How to Obtain
interior face	roll onto face	no other choice
exterior face	roll onto face	place rollon on face (default)
exterior face	stay on edge	place cap on face

A *smooth vertex* is one at which there is another tangent edge incident that can be followed. A *sharp vertex* is one at which there is no other tangent edge to follow.

Transitions directly onto a face that occur at one of the ends of the edge are not supported.

A blend is disallowed from following the edge when it has the option of rolling onto the interior face. Not rolling onto the interior face would violate the rule that *concave* blends add material and that *convex* ones remove it.

Some blends do not move along the edge, but remain wholly stationary at a point on the edge. In such cases the faces cannot be distinguished as either interior or exterior and the blend is forced to roll onto both. The only such blend allowed is the *osculating torus* produced by blending a plane against a straight line normal to it.

Blends on Vertices

Topic: Blending

For blends on vertices, the only transitions allowed are onto the edges of the vertex.

Transitions directly from a vertex onto a face are not supported. Transitions onto edges are forced, and they cannot be avoided.

Capping Restrictions

Topic: *Blending

Entity–entity blending uses the same capping algorithms as standard blending, so all the standard blending capping restrictions apply to entity–entity blending. The entity–entity blending algorithms also impose some additional restrictions.

In order to process caps successfully and to determine the correct extent of an entity–entity blend sequence, it is necessary to distinguish side caps (which do not define the end of the sequence) from end caps (which do). The algorithms for doing this rely on the fact that the end points of any caps will lie on the boundaries of the underlying blended entities. If this does not hold for any particular situation, then the cap may be identified incorrectly. Even if these stipulations do hold, but the number of body edges involved between the start and end points of the cap is large (typically around half a dozen or more), then the cap may be identified incorrectly. Periodic blend surfaces pose a problem since end and side caps may be particularly difficult to distinguish. It may sometimes be apparent that a cap has been identified incorrectly if an error message mentions a particular kind of cap (end or side) when the other type is actually desired.

Entity–Entity Blending Limitations

Topic: Blending

Entity–entity blends are designed to be used in conjunction with ordinary edge sequence following blends, and to provide a number of topologically new blends (such as edge–face blends). Although they will blend faces against one another, they are not intended as a substitute to standard blending, as they have some limitations. These include:

- Only one entity–entity blend attribute may be applied to the body and processed at a time. If multiple blend sequences are to be performed, the first attribute must be applied and fixed before moving on to the next.
- Entity–entity blends cannot be mitered against any other blend. In some cases, similar effects may be achieved by fixing sequences of entity–entity blends separately. However, this is not possible in all cases, such as when the faces to be mitered form part of the same sequence.
- Entity–entity blends can never form the boundaries of any vertex blend. Even entity–entity blends which are face–face blends are disallowed from doing so.

Spring Curve Design

Topic: *Blending

Application developers may need to define blends that follow particular spring curves. The desired spring curves must first be imprinted on the body, then an entity–entity blend set on the body which is capable of following a smooth edge sequence. The user can create a tool body with which to slice the body to be blended. Particular sections of this slice can then be imprinted on the blend body.

ABL provides utilities for designing and imprinting these spring curves. These include the APIs `api_abh_slice`, and `api_abh_imprint`.