Chapter 3.
Functions

Topic: Ignore

The function interface is a set of Application Procedural Interface (API) and Direct Interface (DI) functions that an application can invoke to interact with ACIS. API functions, which combine modeler functionality with application support features such as argument error checking and roll back, are the main interface between applications and ACIS. The DI functions provide access to modeler functionality, but do not provide the additional application support features, and, unlike APIs, are not guaranteed to remain consistent from release to release. Refer to the *3D ACIS Online Help User's Guide* for a description of the fields in the reference template.

api_convert_pipes

Function: Local Operations

Action: Replaces the geometry of any faces in the supplied array which currently are pipe_spl_sur splines with rb_blend_spl_sur splines.

Prototype:

```
outcome api_convert_pipes (
    int const nface,         // number of faces
    FACE* face[],            // faces being converted
    AcisOptions* ao = NULL   // ACIS options
    );

outcome api_convert_pipes (
        BODY* body,          // body to convert
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Replaces the geometry of any faces in the supplied array which currently are pipe_spl_sur splines with rb_blend_spl_sur splines.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: Refer to the Errors listed for the function, api_tweak_faces.

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_edge_taper_faces

Function: Local Operations

Action: Tapers an array of faces about an array of corresponding edges and a supplied draft direction by a given draft angle.

Prototype:

```
outcome api_edge_taper_faces (
    int const nface,                // number of
                                    // faces
    FACE* face[],                   // faces being
                                    // tapered
    EDGE* edge[],                   // edges to be
                                    // tapered about
    SPAunit_vector const& draft_dir,
// draft
                                    // direction
    double const& draft_angle,      // draft angle
    SPAposition box_low,            // start of
                                    // intersection
                                    // box to be used
    SPAposition box_high,           // end of
                                    // intersection
                                    // box to be used
    AcisOptions* ao = NULL          // ACIS options
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/edge.hxx″
#include ″kernel/kerndata/top/face.hxx″
#include ″lop_husk/api/lop_api.hxx″
#include ″baseutil/vector/position.hxx″
#include ″baseutil/vector/unitvec.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description:

The purpose of edge taper is to facilitate extraction from a mould rather than changing the general appearance of a model. So typically the angle supplied would be very small.

Replaces surfaces of supplied faces with surfaces tapered by draft angle about the corresponding edge and draft vector. Note that the corresponding edge must lie on the face to be tapered.

The taper operation on the faces is determined by the number of faces, the faces, the edges, the draft directions and the draft angle.

If a face is to be tapered about several edges, these edges must all be supplied, which means that some faces may have to be supplied more than once. This will lead to a topology change (see below).

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

If the option “validate_lop” is changed to “validate_only” then api_edge_taper_faces does only a validation of the edge taper. If the validation fails, the faces and edges which have caused it to fail are returned using the error return mechanism.

If the option is set to “validate_and_lop”, validation is carried out. If the validation fails, edge and faces are returned in the error return mechanism, and the edge–taper is not attempted. If the validation is successful, it is followed by the actual edge taper.

Note that successful validation is no guarantee of a successful edge–taper. No validation is carried out with the final setting of the option, which is “lop_only”. The default is “validate_and_lop”.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function, api_tweak_faces.

If a face is being tapered about more than one edge, a new edge will be introduced that splits the face into several bits (one per edge about which the original face is to be tapered).

Vent faces are added between mergeable faces, when one of them is not being tapered. Vent faces can only be added if there is a face on the original model that shares a vertex with the mergeable edge, does not have the mergeable edge in its boundary, and that will be adjacent to the vent faces after the taper. Vent faces can also be added at tangent edges when just one of the two faces that share the edge is being tapered. Note however, that a bent face will only be added if there is no intersection between the surface of the face that is being tapered and the surface of the face that isn't.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Valid angle (between –90 and +90 degrees, and non zero) must be supplied or error;–
LOP_TAP_BAD_ANGLE
Valid direction must be supplied or error;–
LOP_TAP_BAD_NORMAL
Surface must be able to be tapered as requested, or error;–
LOP_TAP_NO_SURF
Supplied edges must lie on corresponding faces, or error;–
LOP_TWK_NO_EDGE
Supplied edges must not be parallel to draft vector at any point, or error;–
LOP_TWK_NO_EDGE
Normal of face to be tapered should not be parallel to draft vector, or error;–
LOP_TAP_NO_SURF (FACE*)
Must be able to find a face adjacent to every vent face, or error;–
LOP_TWK_NO_ADJ_FACE

Limitations: In addition to the following, refer to the Limitations listed for the function, api_tweak_faces.

Only planes, cones, ruled surfaces, and previously edge tapered faces (provided the same edge and the same draft direction is used) may be tapered.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_initialize_local_ops

Function: Local Operations, Modeler Control

Action: Initializes the local operations library.

Prototype:
```
outcome api_initialize_local_ops ();
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”lop_husk/api/lop_api.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: System routine

api_move_faces

Function: Local Operations

Action: Moves an array of faces through a transform.

Prototype:
```
outcome api_move_faces (
    int const nface,         // number of faces
    FACE* face[],            // faces being moved
    SPAtransf const& tr,     // transform used
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/face.hxx”
#include ”lop_husk/api/lop_api.hxx”
#include ”baseutil/vector/position.hxx”
#include ”baseutil/vector/transf.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: Replaces surfaces of supplied faces with surfaces moved by a transform.

The move operation on the faces is determined by two arguments, the number of faces and which faces.

The movement is defined by the transform.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Mergeable edges will be retained provided they have a NO_MERGE_ATTRIB.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Translation vector must be non–zero, or error;–
LOP_MOVE_BAD_TRANSFORM

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_offset_body

Function: Local Operations

Action: Offsets all faces of a body based on a given distance.

Prototype:

```
outcome api_offset_body (
    BODY* body,              // body to be offset
    double offset,           // distance to offset
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/body.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Offsets the faces of the supplied body by an offset distance. Faces with radial surfaces which cannot be so offset are removed and the resulting wound healed by the surrounding face surfaces.

The offset body operation is defined by the supplied body and a given distance.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Mergeable edges will be retained provided they have a NO_MERGE_ATTRIB.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function, api_tweak_faces.

Faces with radial surfaces which cannot be offset by the distance without degenerating or inverting (i.e., spheres, cones, blended edges, blended vertices and tori) are removed and the resulting wound healed by the surrounding face surfaces, before the offset.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Valid offset (greater than minus half the body box max side), and not a zero offset (magnitude greater than twice SPAresabs) or error;–
LOP_OFF_BAD_OFFSET

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_offset_faces

Function: Local Operations

Action: Offsets an array of faces based on a given distance.

Prototype:
```
outcome api_offset_faces (
    int const nface,         // number of faces
    FACE* face[],            // faces being tweaked
    double offset,           // distance to offset
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Replaces surfaces of supplied faces with surfaces offset by offset distance. Radial faces with surfaces which cannot be so offset are removed and the resulting wound healed by the surrounding face surfaces.

The offset operation on the faces is determined by two arguments, the number of faces and which faces.

The offset distance is defined by a double.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Mergeable edges will be retained provided they have a NO_MERGE_ATTRIB.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function, api_tweak_faces.

Faces with radial surfaces which cannot be offset by the distance without degenerating or inverting (spheres, cones, blended edges, blended vertices, and tori) are removed and the resulting wound healed by the surrounding face surfaces,before the offset.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Valid offset (greater than minus the body box max side), and not a zero offset (magnitude greater than twice SPAresabs) or error;–
LOP_OFF_BAD_OFFSET

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_offset_faces_specific

Function: Local Operations

Action: Offsets an array of faces, each face as required.

Prototype:

```
outcome api_offset_faces_specific (
    int const& n_def_face,  // number of faces offset
    FACE* def_face[],       // faces being offset
    double def_offset,      // default distance to
                            // offset
    int const& n_spec_face, // number of faces with
                            // specific offsets
    FACE* spec_face[],      // faces with specific
                            // offsets
    double spec_offset[],   // specific offset values
    SPAposition box_low,    // start of intersection
                            // box to be used
    SPAposition box_high,   // end of intersection
                            // box to be used
    AcisOptions* ao = NULL  // ACIS options
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Replaces surfaces of supplied faces with surfaces offset. Unless in the specific array, each face in the face array is offset by the default offset distance. If a face is in the specific array, the face is offset by the amount in the corresponding entry in the specific distance array. Note that faces with specific offsets must be in both face arrays.

Radial faces with surfaces which cannot be so offset are removed and the resulting wound healed by the surrounding face surfaces.

The offset operation on the faces is determined by four arguments, the number of faces and which faces, the number of faces with specific offsets and the faces with specific offsets.

The default offset distance is defined by a double. The specific offset distances are defined by an array of doubles.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function, api_tweak_faces.

Faces with radial surfaces which cannot be offset by the distance without degenerating or inverting (spheres, cones, blended edges, blended vertices, and tori) are removed and the resulting wound healed by the surrounding face surfaces,before the offset.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Valid offset (greater than minus the body box max side), and not a zero offset (magnitude greater than twice SPAresabs) or error;–
LOP_OFF_BAD_OFFSET

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_remove_lop_attribs

Function: Local Operations

Action: Removes all attributes that a local operation attached to EDGEs and VERTEXs contained in an entity.

Prototype:
```
outcome api_remove_lop_attribs (
    ENTITY* entity,          // entity to be cleaned
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/data/entity.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Removes all ATTRIB_LOP_EDGE and ATTRIB_LOP_VERTEX attributes attached to EDGEs and VERTEXs on an ENTITY. These attributes are attached when using the APIs api_tweak_faces_init, api_tweak_fix_edge, and api_tweak_fix_vertex.

Errors: None

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_shadow_taper_faces

Function: Local Operations

Action: Inserts a ruled face which tangentially meets the supplied face, and which covers the region in the shadow cast from a light source.

Prototype:

```
outcome api_shadow_taper_faces (
    int const nface,            // number of faces
    FACE* face[],               // faces being
                                // tapered
    SPAunit_vector const&       // draft
        draft_dir,              // direction
    double const& draft_angle,  // draft angle
    SPAposition box_low,        // start of
                                // intersection box
                                // to be used
    SPAposition box_high,       // end of
                                // intersection box
                                // to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/face.hxx”
#include ”lop_husk/api/lop_api.hxx”
#include ”baseutil/vector/position.hxx”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description:

The purpose of shadow taper is to facilitate extraction from a mold rather than changing the general appearance of a model. So typically the angle supplied would be very small.

Inserts a ruled face which tangentially meets the supplied face, and which covers the region that is in the shadow that is cast from a light source which is directed from an infinite point and is at a given angle from the supplied draft direction.

Replaces regions of the face to be shadow tapered where the angle between the normal and the draft vector is greater than the complement of the given angle (i.e., greater than 90 – given angle) by ruled faces where the angle between the draft vector and the normal is exactly the complement of the given angle. Thus the ruled faces meet the face to be tapered tangentially and are bounded by the intersections with either neighboring faces or the same face. The regions to be replaced are referred to as ”being in shadow” with respect to the draft vector and the given angle.

The complement was chosen rather than the given angle itself in order to keep shadow tapering in line with taper faces and edge taper, where a small angle provokes a smaller change than a large angle.

The shadow taper operation on the faces is determined by the number of faces, the faces, the direction and the angle.

The draft direction is defined by a vector and the draft angle is defined by a double.

The box_low and box_high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function, api_tweak_faces.

Almost every shadow taper will result in a topology change as the parts of the body that are ”in shadow” are replaced by ruled faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

Valid direction must be supplied, or error;–
LOP_TAP_BAD_NORMAL
Valid angle (at least 0 and less than 90 degrees) must be supplied or error;–
LOP_TAP_BAD_ANGLE
Surface must be able to be tapered as requested, or error;–
LOP_TAP_NO_SURF (outcome points to standard_error_info containing a pointer to a FACE*)

Limitations: In addition to the following, refer to the Limitations listed for the function, api_tweak_faces.

More than one face that is in the shadow cannot be deleted.

In general the model must contain a face that is adjacent to the face with the silhouettes and that is intersected by the new face.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_sweep_more

Function: Local Operations, Sweeping

Action: Sweeps an array of faces along a path defined by the faces adjacent the given faces.

Prototype:
```
outcome api_sweep_more (
    int nfaces,              // number of faces
    FACE* faces[],           // faces being moved
    double dist,             // distant to sweep faces
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/top/face.hxx″
#include ″lop_husk/api/lop_api.hxx″
#include ″baseutil/vector/position.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: Replaces surfaces of supplied faces with surfaces moved along the sweep path. The sweep path is determined by a face that is not in that set of faces given but is adjacent to at least one of those faces in the set. This face must determine the sweep path. This means that it must be a cone, cylinder, plane, torus, sweep_spl_sur, rot_spl_sur, sum_spl_sur, or an offset of one of these surfaces.

The optional intersection box limits the size of intersections between surfaces which might otherwise be very long. It cannot be used to choose faces. Its main purpose is to speed up complicated cases where the intersection curves might be very long, thus improving performance. The box must contain the final faces and if omitted defaults to the size box.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the errors listed for the function, api_tweak_faces.

Note, some of the following errors will return a standard_error_info object in the API outcome. In these cases, the type of ENTITY returned, in parentheses, follows the error message below.

Distance must be non–zero, or error;–
LOP_MOVE_BAD_TRANSFORM.

Rail and path information on adjacent swept face must be good, or error;–
LOP_SWP_BAD_RAIL_OR_PATH (FACE *).

Limitations: If the first face is not planar and does not intersect the sweep path, ambiguous results may occur.

Refer to api_tweak_faces for more limitations.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_taper_faces

Function: Local Operations

Action: Tapers each given face about the intersection between the surface underlying the face and a given plane by a given draft angle.

Prototype:

```
outcome api_taper_faces (
    int const nface,            // number of faces
    FACE* face[],               // faces being
                                // tapered
    SPAposition const& point,       // draft plane
point
    SPAunit_vector const& normal,   // draft plane
normal
    double const& draft_angle,  // draft angle
    SPAposition box_low,        // start of
                                // intersection box
                                // to be used
    SPAposition box_high,       // end of
                                // intersection box
                                // to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/top/face.hxx”
#include ”lop_husk/api/lop_api.hxx”
#include ”baseutil/vector/position.hxx”
#include ”baseutil/vector/unitvec.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description:

The purpose of plane taper is to facilitate extraction from a mold rather than changing the general appearance of a model. So typically the angle supplied would be very small. Replaces surfaces of supplied faces with surfaces tapered by draft angle about the intersection between the draft plane and the surface. Note that the intersection between the draft plane and the surface need not lie on the face to be tapered, and that even when it does, this intersection need not be an edge of the body (unlike in the command edge_taper_faces).

The taper operation on the faces is determined by two arguments, the number of faces and the faces.

The draft plane is defined by a position and a vector. The draft angle is defined by a double.

Whether the taper is executed or just validated is controlled by the option validate_lop. One of its setting is lop_only in which the taper is executed. When set to validate_only the taper is validated and if validation fails, the faces which failed to validate are returned in the error return mechanism. The default is validate_and_lop in which validation is carried out. If the validation fails, faces are returned in the error return mechanism, and the taper is not attempted. If the validation is successful, it is followed by the actual taper operation.

Note that successful validation is no guarantee of a successful taper.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Topology Changes:

In addition to the following, refer to the topology changes listed for the function api_tweak_faces.

Vent faces are added between mergeable faces, when one of them is not being tapered. Vent faces can only be added if there is a face on the original model that shares a vertex with the mergeable edge, does not have the mergeable edge in its boundary, and that will be adjacent to the vent faces after the taper. Vent faces can also be added at tangent edges when just one of the two faces that share the edge is being tapered. Note however, that a bent face will only be added if there is no intersection between the surface of the face that is being tapered and the surface of the face that isn't.

Geometry Changes:

Refer to the geometry changes listed for the function api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function api_tweak_faces.

Valid normal must be supplied, or error;–
LOP_TAP_BAD_NORMAL
Valid angle (between –90 and +90, and non zero) must be supplied or error;–
LOP_TAP_BAD_ANGLE
Surface must be able to be tapered as requested, or error;–
LOP_TAP_NO_SURF (outcome points to standard_error_info containing a pointer to a FACE*)
Normal of face to be tapered should not be parallel to draft vector, or error;–
LOP_TAP_NO_SURF (FACE*)
Must be able to find a face adjacent to vent face if a vent face is inserted, or error;–
LOP_TAP_NO_ADJ_FACE

Limitations: In addition to the following, refer to the Limitations listed for the function api_tweak_faces.

Only planes, cones, ruled surfaces and previously plane tapered surfaces (provided the same taper plane is used) may be plane tapered.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_terminate_local_ops

Function: Local Operations, Modeler Control

Action: Terminates the local operations library.

Prototype:
```
outcome api_terminate_local_ops ();
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”lop_husk/api/lop_api.hxx”
```

Description: Refer to Action.

Errors: None

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: System routine

api_tweak_extend_faces

Function: Local Operations

Action: Extends an array of surfaces for use in api_tweak_faces.

Prototype:
```
outcome api_tweak_extend_faces (
    int const nface,         // number of faces
    FACE* face[],            // faces being tweaked
    SURFACE* tool_surface[], // new surfaces
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/geom/surface.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Given an array of surfaces and the an array of faces that the surfaces will be tweaked into, the surfaces are automatically extended so that the tweak has a greater chance of being successful.

The curves underlying edges between two faces that are not directly involved in the tweak but adjacent to a face being tweaked may also be extended.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: In addition to the following, refer to the Errors listed for the function, api_tweak_faces.

At least one face must be supplied, or error;–
LOP_TWK_NO_FACE
Faces must be valid, non–duplicate and from the same body, or error;–
LOP_TWK_BAD_FACE (FACE*)
Tool surfaces must be valid or error;–
LOP_TWK_BAD_SURFACE(FACE*)
Box, if supplied must be valid, or error;–
LOP_BAD_BOX.

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_faces

Function: Local Operations

Action: Tweaks an array of faces to new surfaces.

Prototype:
```
outcome api_tweak_faces (
    int const nface,          // number of faces
    FACE* face[],             // faces being tweaked
    SURFACE* tool_surface[], // new surfaces
    int reverse[],            // flags to reverse new
                              // surfaces; zero = leave
                              // as–is,
                              // nonzero = reverse
    SPAposition box_low,      // start of intersection
                              // box to be used
    SPAposition box_high,     // end of intersection
                              // box to be used
    AcisOptions* ao = NULL    // ACIS options
    );
```

Includes:
```
#include ”kernel/acis.hxx”
#include ”kernel/kernapi/api/api.hxx”
#include ”kernel/kerndata/geom/surface.hxx”
#include ”kernel/kerndata/top/face.hxx”
#include ”lop_husk/api/lop_api.hxx”
#include ”baseutil/vector/position.hxx”
#include ”kernel/kernapi/api/acis_options.hxx”
```

Description: Replaces surfaces of supplied faces with supplied surfaces, taking the supplied reverse flags into consideration. Choose reverse flags carefully, as resulting body is not checked.

The box low and high arguments are used to limit any geometrical intersections and so improve performance. Limiting the intersection improves performance. The box must contain the final faces and if omitted defaults to the size box.

Topology Changes:

The topology changes performed by tweak_faces on the tweaked faces and their neighbors are:

Mergeable edges and vertices on the supplied faces will be merged out.

Edges and vertices between faces that are tweaked to the same geometry will also be merged out.

Two edge vertices on the supplied faces between edges on the same geometrical non analytic intersection curve, which are not presently mergeable elsewhere, will be merged out by lop:tweak–faces if the option lop_merge_vertex is set on, which is the default setting.

Vertices with more than three edges may split into two or more vertices during a local operation. Single inverted faces needed to fill any gap in the body will be made (if two or more inverted faces are needed the operation will fail).

Isolated single edge loops may degenerate exactly to a point or be removed altogether and their faces will be deleted when appropriate.

Edges in multi edge loops may degenerate exactly to a point and will be removed.

Where an edge exactly degenerates to a point in a multi edge loop, the edges either side will be detected becoming fully or partially coincident, and the loop repaired by splitting it into valid and degenerate loops. Degenerate loops will be deleted, and their faces when appropriate.

When faces have been deleted, their shells and lumps will be repartitioned and split if necessary.

The option lop_ff_int switches on face face intersection checking on the result body. The default for the option is off. If the option is switched on and a face face intersection is found in the body, the operation fails.

The option lop_repair_self_int switches on the repairing of self intersections on the resulting body. The default for the option is off. If the option is switched on api_repair_body_self_ints is called at the end of successful tweaks and supplied with the changed body faces.

If the repair of self intersections is required, the options lop_check_invert and lop_ff_int are disabled internally so that self–intersection checks are avoided and possibly an invalid body is passed to api_repair_body_self_ints with no error raised.

Geometry Changes:

The geometry changes performed by tweak_faces on the tweaked faces and their neighbors are:

pipe_spl_sur spline surfaces will be converted to rb_blend_spl_sur spline surfaces in the supplied faces and their neighbors if the option lop_convert_pipe is set on, which is the default setting.

Errors: Note, some of the following errors will return a standard_error_info object in the API outcome. In these cases, the type of ENTITY returned, in parentheses, follows the error message below. (When the ENTITY is new and will be lost on rollback, no standard_error_info object is returned.)

At least one face must be supplied, or error;–
LOP_TWK_NO_FACE
Faces must be valid, non–duplicate and from the same body, or error;–
LOP_TWK_BAD_FACE (FACE*)
Tool surfaces must be valid or error;–
LOP_TWK_BAD_SURFACE (FACE*)
Body must be manifold or error;–
LOP_TWK_NON_MANIFOLD (COEDGE*)
Body must be solid or error;–
LOP_TWK_FREE_EDGE (COEDGE*)
New surface geometries must intersect consistently, or errors
LOP_TWK_NO_EDGE (EDGE*) no solution for an edge
LOP_TWK_NO_VERT (VERTEX*) no solution for a vertex
Internal algorithmic problems produce error;–
LOP_TWK_INTERNAL
Unpermitted topology changes, if detected, will produce the error;–
LOP_TWK_TOPOL_CHANGE (FACE*)
Box, if supplied must be valid, or error;–
LOP_BAD_BOX

Limitations: Body must be manifold and solid.

No other topology changes than those documented above are permitted. Tweak_faces does not extend the geometry of the tool surfaces or their neighbors (unlike the other local operations which do).

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_faces_init

Function: Local Operations

Action: Adds attributes to edges completing the initial stage of tweak.

Prototype:

```
outcome api_tweak_faces_init (
    int const nface,         // number of faces
    FACE* face[],            // faces being tweaked
    SURFACE* tool_surface[], // new surfaces
    int reverse[],           // flags to reverse new
                             // surfaces; zero = leave
                             // as-is,
                             // nonzero = reverse
    ENTITY_LIST&             // edges with multiple
        multiple_sols,       // solutions
    SPAposition box_low,     // start of intersection
                             // box to be used
    SPAposition box_high,    // end of intersection
                             // box to be used
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/geom/surface.hxx"
#include "kernel/kerndata/top/face.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "baseutil/vector/position.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Completes the initial stage of a tweak where attributes are added to edges. Returns a list of edges that have more than one possible solution so the user may pick which solution the tweak uses in the next stage.

To complete the tweak, use api_tweak_faces, which will continue the tweak without duplicating the work already done.

The box low and high arguments are used to limit any geometrical intersections. Limiting the intersection improves performance. The box must contain the final faces. If omitted, it defaults to the size box.

Refer to the function api_tweak_faces for a complete description.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: Refer to the Errors listed for the function api_tweak_faces.

Limitations: Refer to the Limitations listed for the function, api_tweak_faces.

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_fix_edge

Function: Local Operations

Action: Fixes an edge prior to a tweak.

Prototype:

```
outcome api_tweak_fix_edge (
    EDGE* ed,                // edge to be fixed
    CURVE* cu,               // new geometry (may be
                             // NULL)
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:

```
#include ″kernel/acis.hxx″
#include ″kernel/kernapi/api/api.hxx″
#include ″kernel/kerndata/geom/curve.hxx″
#include ″kernel/kerndata/top/edge.hxx″
#include ″lop_husk/api/lop_api.hxx″
#include ″kernel/kernapi/api/acis_options.hxx″
```

Description: Places ATTRIB_LOP_EDGE on edge containing new geometry for the edge to be used during the next tweak. NULL new geometry implies that the geometry must not change during the tweak. Data for an edge from the most recent call referring to it applies.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: None

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_fix_vertex

Function: Local Operations

Action: Fixes a vertex prior to a tweak.

Prototype:
```
outcome api_tweak_fix_vertex (
    VERTEX* vt,             // vertex to be fixed
    APOINT* ap,             // new geometry (may be
                            // NULL)
    AcisOptions* ao = NULL  // ACIS options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "kernel/kerndata/geom/point.hxx"
#include "kernel/kerndata/top/vertex.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: Places ATTRIB_LOP_VERTEX on vertex containing new geometry for the vertex to be used during the next tweak. NULL new geometry implies that the geometry must not change during the tweak. Data for an vertex from the most recent call referring to it applies.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: None

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_pick_edge_solution

Function: Local Operations

Action: Specifies an edge solution that a subsequent tweak will use.

Prototype:
```
outcome api_tweak_pick_edge_solution (
    EDGE* edge,              // edge being specified
    int soln_no,             // the solution curve to
                             // use when curve is
                             // non-NULL
    CURVE* curve = NULL,     // the new edge geometry
    AcisOptions* ao = NULL   // ACIS options
    );
```

Includes:
```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kerndata/geom/curve.hxx"
#include "kernel/kerndata/top/edge.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: After a call to api_tweak_faces_init, attributes containing EDGE solution curves are attached to EDGEs involved in the tweak. This function allows the user to specify which of multiple solutions a subsequent tweak will use. Note, it is possible to choose a solution that forces the tweak to fail or make a bad model. It is also possible to specify incompatible solutions. Therefore, the user must take care when using this function, particularly when specifying solutions on more than one EDGE.

If a curve is supplied, this curve will be used as the new edge geometry (if the curve is one of the possible solutions in the edge attribute) and the variable solution_no is ignored. If no curve is supplied, solution_no represents the index of the curve to be used.

To query the possible solutions on an edge, use api_tweak_query_edge_solutions.

To complete the tweak after any number of edge solutions have been specified, use api_tweak_faces.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: If the edge does not have a LOP_EDGE_ATTRIB attribute, i.e. the edge was not involved in the initial stages of a tweak, or the solution number is not valid LOP_TWK_NO_EDGE.

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model

api_tweak_query_edge_solutions

Function: Local Operations

Action: Returns an ENTITY_LIST of CURVEs that are possible solutions for an EDGE's geometry in a tweak.

Prototype:

```
outcome api_tweak_query_edge_solutions (
    EDGE* edge,              // edge being queried
    ENTITY_LIST& curve_list, // list of possible
                             // solutions
    AcisOptions* ao = NULL   // ACIS options
     );
```

Includes:

```
#include "kernel/acis.hxx"
#include "kernel/kernapi/api/api.hxx"
#include "lop_husk/api/lop_api.hxx"
#include "kernel/kerndata/lists/lists.hxx"
#include "kernel/kerndata/top/edge.hxx"
#include "kernel/kernapi/api/acis_options.hxx"
```

Description: After a call to api_tweak_faces_init, attributes containing edge solution curves are attached to edges involved in the tweak. This function allows the user to obtain a list of the possible solutions for an EDGE.

Note, the CURVEs returned in curve_list are created on the heap and it is the caller's responsibility to lose them.

To select a possible solution on the edge, use api_tweak_pick_edge_solution.

To complete the tweak after any number of edge solutions have been specified, use api_tweak_faces.

Topology Changes:

Refer to the topology changes listed for the function, api_tweak_faces.

Geometry Changes:

Refer to the geometry changes listed for the function, api_tweak_faces.

Errors: If there is no LOP attribute on the EDGE, LOP_TWK_NO_EDGE.

Limitations: None

Library: lop_husk

Filename: lop/lop_husk/api/lop_api.hxx

Effect: Changes model