Chapter 3.
Options

Topic: Ignore

Options may be set to modify the behavior of ACIS. An option's value may be a flag (indicating an on/off state), a number (integer or real number), or a string. Options may be set in a Scheme application (such as Scheme AIDE) using the Scheme extension option:set; in the ACIS Test Harness using the command option; or in a C++ application using one of several API functions. Refer to the *3D ACIS Online Help User's Guide* for a description of the fields in the reference template.

lop_error_return_all

Option: Modeler Control, Local Operations

Action: Controls whether or not all entities (including "dead" entities) are returned in "error info" objects.

Name String: **`lop_error_return_all`**

Scheme:	boolean	#f, #t	#f
Test Harness:	integer	0, 1	0
C++:	logical	FALSE, TRUE	FALSE

Description: When this option is off (false), only entities that will remain "alive" after a roll back are returned in standard_error_info objects and ff_error_info objects. Turning this option on (true) allows all entities, even those created in the API (i.e., "dead" entities after the roll back), to be returned. Entities created during the API remain accessible until the next API call, when the roll back occurs and they are deleted and their memory is freed.

The user is warned to exercise caution when using this option.

Great care must be taken when dealing with dead entities. To help the user avoid problems (such as entities unexpectedly disappearing), flags have been added to standard_error_info and ff_error_info objects that indicate which entities are dead.

Example:

```
; lop_error_return_all
; Turn on options so all entities returned at roll
; back
(define profile (wire–body:points (list
    (position 0 0 0) (position –5 0 0)
    (position –5 –5 0) (position 5 –5 0)
    (position 5 2 0) (position –2 2 0))))
;; profile
(define sheet (sweep:along–vector profile #f
    (gvector 0 0 10)))
;; sheet
(option:set ”lop_ff_int” #t)
;; #f
(option:set ”lop_error_return_all” #t)
;; #f
; thicken so there are improper intersections in the
; newly created side faces which are lost after
; rollback
(shell:sheet–thicken sheet 4 #t)
;; entid 1434840 entid 1455656: Error: edge
;; intersection
;; entid 1421120 entid 1436448: Error: edge
;; intersection
;; entid 1438424: Warning: invalid face
;; entid 1435488 entid 1455752: Error: edge
;; intersection
;; entid 1454984 entid 1419104: Error: edge
;; intersection
;; entid 1441504: Warning: invalid face
;; entid –56856 entid 1441552: Error: face
;; intersection
;; entid 1438472 entid 1440592: Error: face
;; intersection
;; Warning: shell entid –58248 not used in
;; containment check
;; Warning: lump entid –83264 not used in containment
;; check
;; *** Error shell:sheet–thicken: improper edge/edge
;; intersection
```

lop_ff_error_prevent_roll

Option: Modeler Control, Local Operations

Action:	Controls whether or not errors in local operations face/face checks cause a roll back.		
Name String:	**`lop_ff_error_prevent_roll`**		
Scheme:	boolean	#f, #t	#f
Test Harness:	integer	0, 1	0
C++:	logical	FALSE, TRUE	FALSE

Description: If this option is off (false), roll back will occur when errors are found during the face/face checking stage of a local operation. When this option is on (true), roll back is prevented. Errors found during the face/face checking stage of a local operation (including remove faces and shelling) will not cause a roll back; however, they will still be returned in the API outcome error_info object.

When this option is on, bad models can be returned. The user is warned to exercise caution when using this option.

Example:

```
; lop_ff_error_prevent_roll
; Turn option on so roll back is prevented when error
; occurs
(define block (solid:block (position –10 –10 –10)
     (position 10 10 10)))
;; block
(define hole (solid:sphere (position 0 0 0) 5))
;; hole
(bool:subtract block hole)
;; #[entity 2 1]
(option:set ”lop_ff_int” #t)
;; #f
(option:set ”lop_ff_error_prevent_roll” #t)
;; #f
(lop:move–faces (pick:face (ray ( position 8 0 0)
    (gvector 1 0 0))) (gvector –10 0 0))
;; entid –58248 entid –56856: Error: face
;; intersection
;; entid –57000 entid –56808: Warning: shell
;; intersection
;; Warning: lump entid 1425696 not used in
;; containment check
;; *** Warning
;;  (boolean/sg_husk/sanity:CHECK_BAD_FF_INT)
;; improper face/face intersection
;; #[entity 2 1]
```

validate_lop

Option: Modeler Control, Local Operations

Action: Controls whether or not a taper local operation is checked (validated).

Name String: **validate_lop**

Scheme:	string	See *Description*	“validate_and_lop”
Test Harness:	string	See *Description*	“validate_and_lop”
C++:	char*	See *Description*	“validate_and_lop”

Description: This option controls whether or not a taper local operation is checked (validated) to determine if will work before actually performing the operation. If validation fails, the taper operation will not work. However, if validation succeeds, this does *not* guarantee that the taper will work. If validation fails, the entities (faces or edges) at which validation failed are returned using the error return mechanism.

This option only applies to edge taper and plane taper operations.

The argument to this option is a string. Possible values are:

“validate_and_lop”	Basic validation checks on the taper operation are performed. If these checks succeed, the taper is attempted. If the validation fails, the taper is not attempted.
“validate_only”	Some basic checks on the taper operation are performed, but the operation is not attempted.
“lop_only”	No validation is performed. If the taper is attempted, it may or may not succeed.

Example:

```
; validate_lop
; First example shows the use of “validate_only”
; with an edge taper that will not validate.
(define std (view:dl 0 0 500 500))
;; std
(view:set–perspective OFF std)
;; #f
(define b (solid:block (position –5 –5 –5)
    (position 5 5 5)))
;; b
; Set the option
(option:set ”validate_lop” ”validate_only”)
;; “validate_and_lop”
; Attempt an illegal edge–taper
(lop:edge–taper–faces (pick:face (ray
    (position 0 0 0) (gvector 1 0 0)))
    (pick:edge (ray (position 5 0 0)
    (gvector 0 0 –1))) (gvector 0 1 0) 10))
;; Returned edge has been highlighted.
;; *** Error lop:edge–taper–faces: edge tangent is at
;; some point parallel to draft direction
```

```
; example 2: will validate and work
; Second example shows ”validate_and_lop” with an
; edge taper that will validate and be executed.
(define std (view:dl 0 0 500 500))
;; std
(view:set–perspective OFF std)
;; #f
(define b (solid:block (position –5 –5 –5)
    (position 5 5 5)))
; switch the option on
(option:set ”validate_lop” ”validate_and_lop” )
;; “validate_and_lop”
; This should work now
(lop:edge–taper–faces (pick:face (ray
    (position 0 0 0) (gvector 1 0 0)))
    (pick:edge (ray (position 5 0 0)
    (gvector 0 0 –1))) (gvector 0 0 1) 10))
;; #[entity 2 1]
```