Chapter 2.
Scheme Extensions

Topic: Ignore

Scheme is a public domain programming language, based on the LISP language, that uses an interpreter to run commands. ACIS provides extensions (written in C++) to the native Scheme language that can be used by an application to interact with ACIS through its Scheme Interpreter. The C++ source files for ACIS Scheme extensions are provided with the product. *Spatial*'s Scheme based demonstration application, Scheme ACIS Interface Driver Extension (Scheme AIDE), also uses these Scheme extensions and the Scheme Interpreter. Refer to the *3D ACIS Online Help User's Guide* for a description of the fields in the reference template.

ihl:clean

Scheme Extension: Interactive Hidden Line

Action: Removes interactive hidden line data attributes from the model.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: api_ihl_clean

Syntax: `(ihl:clean [token=0] body-list)`

Arg Types:

token	integer
body–list	body | (body ...)

Returns: unspecified

Errors: None

Description: This extension removes and deletes any IHL attributes with a matching view token from the list of bodies.

When bodies are cleaned, attributes are removed and are not stored. These attributes cannot be retrieved using the ihl:retrieve extension.

The optional token is an integer index; the default is 0. When token is 0, no action is taken. A nonzero token causes hidden line data to be removed from attributes attached to the bodies for the view matching that view token; attributes for other views are not affected.

body–list is the list of bodies that are to be cleaned.

Limitations: None

Example:

```
; ihl:clean
; Compute the hidden lines for two different views,
; storing the data in attributes attached to the
; body. Then retrieve and display each view. Clean
; the attributes and redraw.
(define block1
    (solid:block (position –10 –25 –35)
    (position 10 25 35)))
;; block1
; Set a color for the block.
(entity:set–color block1 2)
;; ()
(define cyl1
    (solid:cylinder (position 0 0 –20)
    (position 0 0 20) 30))
;; cyl1
; Set a color for the cylinder.
(entity:set–color cyl1 3)
;; ()
; OUTPUT Original

(define combo (bool:unite cyl1 block1))
;; combo
(define ihldata1 (ihl:compute 1 combo))
;; ihldata1
(ihl:draw ihldata1)
;; #[ihl–data 40248e60]
(ihl:clean 1 combo)
;; ()
(define ihldata1 (ihl:retrieve 1 combo))
;; ihldata1
(ihl:draw ihldata1)
;; #[ihl–data 402471b8]
; OUTPUT Result
```

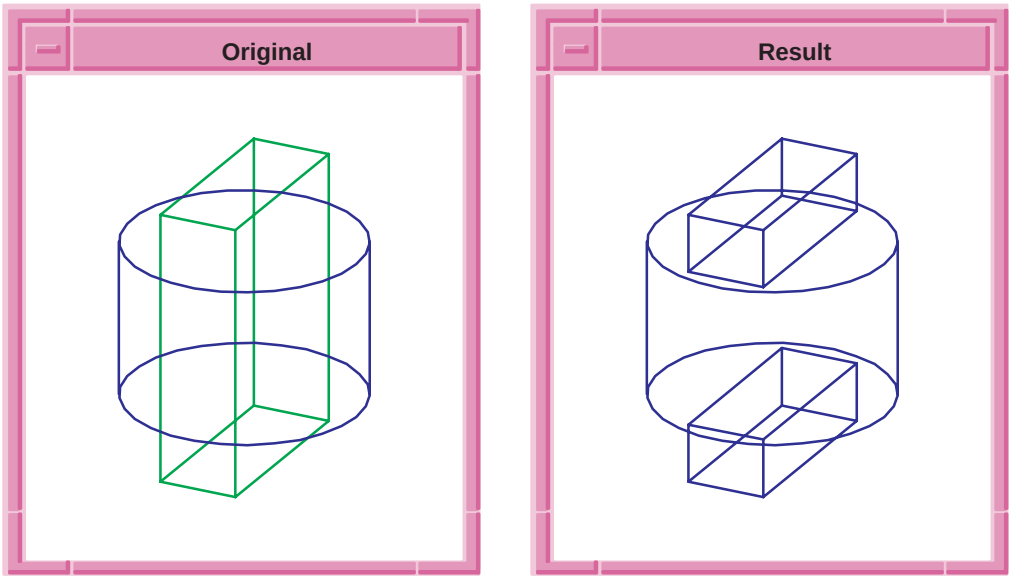

Figure 2-1. ihl:clean

ihl:compute

Scheme Extension: Interactive Hidden Line

Action: Computes interactive hidden line data and optionally stores it on the model as attributes.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: api_ihl_compute

Syntax:
```
(ihl:compute [token=0] body–list [view=active]
    [unfaceted=#f])
```

Arg Types:		
	token	integer
	body–list	body | (body ...)
	view	view
	unfaceted	boolean

Returns: ihl–data

Errors: None

Description: This extension computes the interactive hidden line data for the given viewing parameters and list of bodies.

The optional token is an integer index to a view; the default is 0. When token is 0, hidden line data is computed for drawing, but it is not stored in attributes attached to the bodies. If token is nonzero, the data is stored on the model as attributes and identified by token. Existing attributes identified by the same token are replaced.

The body–list supplies a list of bodies that participate in the computation.

The view argument supplies a view object.

If the unfaceted argument is #t the body will not be faceted. If it is #f (the default), the body will be refaceted prior to hidden line computation and display.

The ihl–data returned is a list of line segments with attached visibility information.

Limitations: None

Example:

```
; ihl:compute
; Compute and draw hidden lines for a view.
(define block1
    (solid:block (position –10 –25 –35)
    (position 10 25 35)))
;; block1
(define cyl1
    (solid:cylinder (position 0 0 –20)
    (position 0 0 20) 30))
;; cyl1
(define combo (bool:unite block1 cyl1))
;; combo
(define ihldata1 (ihl:compute 1 combo #f))
;; ihldata1
(ihl:draw ihldata1)
;; #[ihl–data 40248e60]
```

ihl:display

Scheme Extension: Interactive Hidden Line

Action: Computes and displays hidden line drawings.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: api_ihl_compute

Syntax: (**ihl:display** body–list [unfaceted=#f])

Arg Types:	body–list unfaceted	body | (body ...) boolean

Returns: unspecified

Errors: None

Description: This extension computes and displays the interactive hidden line data for the list of bodies. The default views are the views defined in the current viewport. The hidden line data is not attached to the bodies. If no bodies are specified, all bodies in the currently active part are processed.

If the unfaceted argument is #t the body will not be refaceted. If it is #f (the default), the body will be refaceted prior to hidden line computation and display.

Limitations: None

Example:

```
; ihl:display
; Display the hidden lines for the current view.
(define block1
    (solid:block (position –10 –25 –35)
    (position 10 25 35)))
;; block1
(define cyl1
    (solid:cylinder (position 0 0 –20)
    (position 0 0 20) 30))
;; cyl1
(define unite (bool:unite cyl1 block1))
;; unite
(ihl:display cyl1 #f)
;; ()
```

ihl:draw

Scheme Extension: Interactive Hidden Line

Action: Draws interactive hidden line data.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: None

Syntax: (**ihl:draw** ihl–data [view=active] [clear])

Arg Types:	ihl–data view clear	ihl–data view boolean

Returns: ihl–data

Errors: None

Description: This extension draws ihl–data returned from the ihl:compute or ihl:retrieve extensions.

The optional view specifies the view to print. Prior to the image being written out to a file, the eye, target, and projection mode of the specified (or, active if none is specified) view are internally changed by the system to agree with those of the view in the context of which ihl–data was generated.

If the optional clear is #t, the screen is cleared before drawing. If clear is not specified or is #f, the screen is not cleared before drawing.

Limitations: None

Example:
```
; ihl:draw
; Compute and draw hidden lines for a view.
(define block1
    (solid:block (position –10 –25 –35)
    (position 10 25 35)))
;; block1
(define cyl1
    (solid:cylinder (position 0 0 –20)
    (position 0 0 20) 30))
;; cyl1
(define combo (bool:unite block1 cyl1))
;; combo
(define ihldata1 (ihl:compute 1 combo))
;; ihldata1
(ihl:draw ihldata1)
;; #[ihl–data 40248f40]
```

ihl:postscript

Scheme Extension: Interactive Hidden Line

Action: Prints an interactive hidden line image to a PostScript file.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: None

Syntax:
```
(ihl:postscript ihl–data [filename=plotfile.ps]
    [color=#f] [x–size=195.9 y–size=259.4]
    [view=active])
```

Arg Types:	ihl–data	ihl–data
	filename	string
	color	boolean
	x–size	real
	y–size	real
	view	view

Returns: string

Errors: None

Description: ihl–data specifies IHL data from ihl:compute or ihl:retrieve.

The optional filename specifies the filename where images are sent. The default filename is plotfile.ps.

The optional color is a logical flag (#t or #f) that indicates whether the color is to be sent to the file.

The optional x–size and y–size specify the dimensions, in millimeters, of a rectangle into which the printed image of the viewport is to fit as tightly as possible without distortion. The default is x–size = 195.9mm (7.75 in) by y–size = 259.4mm (10.25 in).

The optional view specifies the view to print. Prior to the image being written out to the file, the eye, target, and projection mode of the specified (or, active, if none is specified) view are internally changed by the system to agree with those of the view in the context of which ihl–data was generated.

This extension returns the filename.

Limitations: None

Example:

```
; ihl:postscript
; Create a cylinder.
(define cyl1
    (solid:cylinder (position –20 –25 –15)
    (position 25 35 20) 17.5))
;; cyl1
; Render the cylinder with ihl
(ihl:postscript (ihl:compute cyl1))
;; ”plotfile.ps”
```

ihl:retrieve

Scheme Extension: Interactive Hidden Line

Action: Retrieves interactive hidden line data from attributes attached to the bodies.

Filename: ihl/ihl_scm/ihl_scm.cxx

APIs: api_ihl_retrieve

Syntax: `(ihl:retrieve body–list [token=0])`

Arg Types:

token	integer
body–list	body | (body ...)

Returns: ihl–data

Errors: None

Description: This extension retrieves the data stored in IHL attributes with a matching view token and returns the list of hidden line segments as an ihl–data object.

All data returned is a copy of the data stored in attributes on the body.

The optional token is an integer index to a view; the default is 0. When token is 0, no action is taken. A nonzero token retrieves interactive hidden line data from attributes corresponding to the views attached to the bodies.

Limitations: None

Example:

```
; ihl:retrieve
; Compute the hidden lines for two different views,
; storing the data in attributes attached to the
; body. Then retrieve and display each view.
(define view1 (view:dl))
;; view1
(define view2 (view:dl))
;; view2
(view:set–eye (position 100 0 100) view2)
;; #[position 0 0 500]
(define block1 (solid:block (position –10 –25 –35)
    (position 10 25 35)))
;; block1
(define cyl1
    (solid:cylinder (position 0 0 –20)
    (position 0 0 20) 30))
;; cyl1
(define combo (bool:unite cyl1 block1))
;; combo
(define ihldata1 (ihl:compute 1 combo view1))
;; ihldata1
(define ihldata2 (ihl:compute 2 combo view2))
;; ihldata2
(define ihldata3 (ihl:retrieve 1 combo))
;; ihldata3
(ihl:draw ihldata3 view1)
;; #[ihl–data 40227c20]
(define ihldata4 (ihl:retrieve 2 combo))
;; ihldata4
(ihl:draw ihldata4 view2)
;; #[ihl–data 40227e50]
```