Linear Path Solid Models

Convert the frames model to a solid volume model

If you are using AutoCAD version 2007 or greater, a solid model version of this form can be developed by using the same computed points as the surface version required for each frame.

AutoCAD's LOFT command can accept multiple sections, our frames, and then create a solid model by connecting these sections to each other and closing the ends.

Each section will be created as a closed 3DPOLY, converted to a REGION and then placed in a selection list. Once all the sections have been accumulated, the selection list will be given to the LOFT command for conversion to a solid.

Two methods are available to connect each section; one is to smooth fit the sections, and the other is to rule fit the sections. The more sections that are used to define the loft, the smoother the form will appear. For rendering purposes, both version will give you a smooth surface.

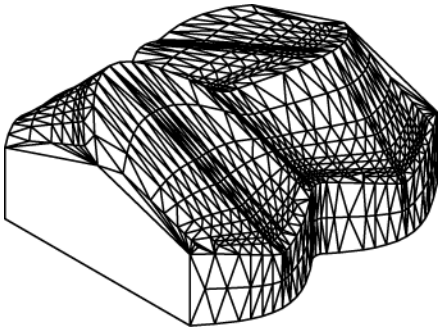

Figure 5.27a: PROG21, converting frames model to a solid volume, smooth fit

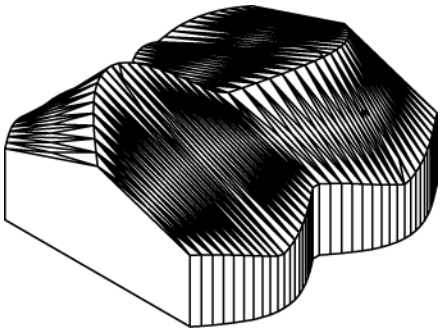

Figure 5.27b: PROG21, converting frames model to a solid volume, ruled fit

The outline for PROG21 is as follows:

```
;-----------------------------------
(defun prog21 ()
 ; input and initial computations
  .
 (repeat numtimes
  ; compute frame points
   .
  ; create a 3DPOLY of the section
   .
  ; convert section to a REGION
   .
  ; add section to LOFT selection list
   .
  ; inc for next section
   .
 )
 ; LOFT sections
  .
)
;-----------------------------------
```

Copy PROG18 to PROG21. Remove the code sections for adding the front and back closing points and remove the frame counter.

Change the 3DMESH command from:

```
; start 3DMESH
(command ".3DMESH" (+ numtimes 2) "6")
; frame counter
(setq cnt 0)
```

To starting a new selection list:

```
; selection list of frames
(setq flist (ssadd))
```

Change the adding of the 3DMESH points from:

```
; add points to 3DMESH
(command pnt1 pnt2 pnt3 pnt4 pnt5 pnt1)
```

To creating the section and adding it to the selection list:

```
; make section
(command ".3DPOLY" pnt1 pnt2 pnt3
 pnt4 pnt5 pnt1 "")
(command ".REGION" "last" "")
; add to list
(setq flist (ssadd (entlast) flist))
```

At the completion of the loop, execute the LOFT command on the sections in the selection list, add:

```
(command ".ZOOM" "e")
; turn off history
(setvar "SOLIDHIST" 0)
; set DELOBJ to delete the sections
(setvar "DELOBJ" 1)
; set LOFTNORMALS =1 for smooth
; =0 for ruled
(setvar "LOFTNORMALS" 0)
; set LOFTPARAM to default
(setvar "LOFTPARAM" 7)
; loft sections
(command ".LOFT" flist "" "")
; turn on history
(setvar "SOLIDHIST" 1)
```

A number of system variables are set for the loft of these sections.

The SOLIDHIST variable is set to turn off the recording of the history of the solid. This saves a less complex solid model and in our case we do not need to revert to the original sections that created the solid. The DELOBJ variable is set to delete the sections once the solid is created; this is the normal default. Set DELOBJ to 0 if the sections are not to be deleted.

The LOFTNORMALS variable is set to 1 for a smooth fit and 0 for a ruled fit. The LOFTPARAM variable is set to its default value.

If other settings are required, check AutoCAD Help for a complete description of all the system variables that pertain to the LOFT command.

Sample script file for PROG21:

```
;-----------------------------------
```

```
(prog21)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
40
; keep line factor
0.75
; cycles
1.0
;-----------------------------------
```

[NOT FOR CLASS]

Completed function PROG21:

```
;-----------------------------------
(defun prog21 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG21:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; selection list of frames
 (setq flist (ssadd))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
  (setq linepart2
   (* (* linelen (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newlinelen
   (+ linepart1 linepart2))
  ; compute endpoint
  (setq epnt1
   (polar xpnt (dtr 90) newlinelen))
  (setq epnt2
   (polar xpnt (dtr 270) newlinelen))
  ; compute edge height
  (setq linepart1 (* zedge linefact))
  (setq linepart2
   (* (* zedge (- 1.0 linefact))
   (abs (cos (dtr ang)))))
  (setq newzedge
   (+ linepart1 linepart2))
  ; compute midpoint height
  (setq linepart1 (* zmid linefact))
  (setq linepart2
   (* (* zmid (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newzmid
   (+ linepart1 linepart2))
  ; compute midpoint offset
  (setq newyoff
   (* yoff (sin (dtr ang))))
  ; compute frame points
  (setq pnt1 epnt2)
  (setq pnt2
   (list (nth 0 epnt2)
   (nth 1 epnt2)
   (+ (nth 2 epnt2) newzedge)))
  (setq pnt3
   (list (nth 0 xpnt)
   (+ (nth 1 xpnt) newyoff)
   (+ (nth 2 xpnt) newzmid)))
  (setq pnt4
   (list (nth 0 epnt1) (nth 1 epnt1)
   (+ (nth 2 epnt1) newzedge)))
  (setq pnt5 epnt1)
  ; make section
  (command ".3DPOLY" pnt1 pnt2 pnt3
   pnt4 pnt5 pnt1 "")
  (command ".REGION" "last" "")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
  ; inc x coord
  (setq xpt (+ xpt xinc))
  (setq xpnt
   (list xpt (nth 1 xpnt)
   (nth 2 xpnt)))
  ; inc ang
  (setq ang (+ ang anginc))
 )
 (command ".ZOOM" "e")
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 (setvar "DELOBJ" 1)
 ; set LOFTNORMALS =1 for smooth
 ; =0 for ruled
 (setvar "LOFTNORMALS" 0)
 ; set LOFTPARAM to default
 (setvar "LOFTPARAM" 7)
 ; loft sections
 (command ".LOFT" flist "" "")
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Convert the frames model to a partially closed solid surface model

Instead of creating a solid volume model from the frame sections, create a solid model where the top and sides have a thickness.

The frame points are used to create an inside and outside section as a REGION. The inside section is created as in PROG21. The outside section is created using the OFFSET command.

The sections are made into REGIONs and the inside is subtracted from the outside. The bottom edge is then subtracted leaving the top and sides in the section. A rectangular REGION is temporarily created to subtract from the section.

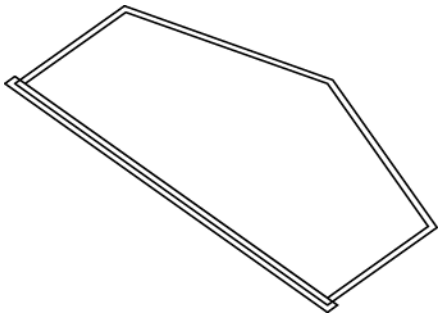

Figure 5.27c1: PROG22, converting frames model to a solid surface model, typical section

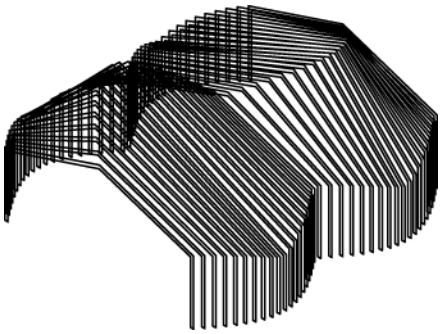

Figure 5.27c2: PROG22, converting frames model to a solid surface model, frames only

In this example, the frame section is created in the XY plane as a simple PLINE and then OFFSET before making it into a REGION. Once the bottom is subtracted, the section is rotated into the YZ plane and added into a selection list for lofting.

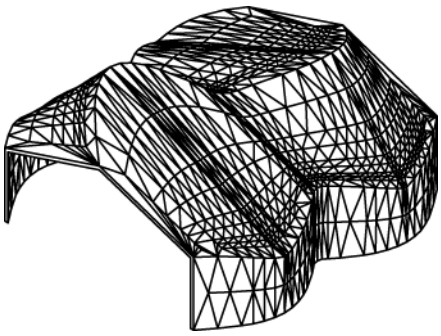

Figure 5.27d: PROG22, converting frames model to a solid surface model, smooth fit

Start function PROG22 with a copy of PROG21.

The outline for PROG22 is as follows:

```
;-----------------------------------
(defun prog22 ()
 ; input and initial computations as
 ; in PROG21 plus input for thickness
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
   ; create a PLINE on the inside edge
    .
   ; place the PLINE in a selection list
    .
   ; OFFSET the PLINE for the thickness
    .
   ; place the OFFSET in a selection list
    .
   ; make each selection a REGION
    .
   ; subtract the inside from the outside
    .
   ; make a rectangular REGION at the base
    .
   ; subtract rectangle from the section
    .
   ; rotate section into the YZ plane
    .
   ; add section to LOFT selection list
    .
   ; inc for next section
    .
 )
 ; LOFT sections
  .
)
;-----------------------------------
```

Review this sequence of commands to create the section in the completed PROG22.

Sample script file for PROG22:

```
;-----------------------------------
(prog22)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
40
; keep line factor
0.75
; cycles
1.0
; thickness
6"
;-----------------------------------
```

Completed function PROG22:

```
;-----------------------------------
(defun prog22 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG22:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
```

```
 "\nEnter keep line factor:"))
(setq numcycles
 (getreal
 "\nNumber of curve cycles:"))
(setq sthick
 (getdist
 "\nEnter surface thickness:"))
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; selection list of frames
(setq flist (ssadd))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
 (setq pnt2
  (list (+ (nth 0 epnt2) newzedge)
  (nth 1 epnt2) 0.0))
 (setq pnt3
  (list (+ (nth 0 xpnt) newzmid)
  (+ (nth 1 xpnt) newyoff) 0.0))
 (setq pnt4
  (list (+ (nth 0 epnt1) newzedge)
  (nth 1 epnt1) 0.0))
 (setq pnt5
  (list (nth 0 epnt1)
   (nth 1 epnt1) 0.0))
 ; create inside edge
 (command ".PLINE" pnt1 pnt2 pnt3
  pnt4 pnt5 "c")
 (setq obj1 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; offset
 (command ".OFFSET" sthick pnt1
  (list (nth 0 pnt5)
    (+ (nth 1 pnt5) sthick) 0.0) "")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; make regions and subtract
  (command ".REGION" obj1 "")
  (setq obj1 (ssadd (entlast)))
  (command ".REGION" obj2 "")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj2 "" obj1 "")
  (setq obj3 (ssadd (entlast)))
  ; remove base
  (command ".RECTANGLE"
   (list
   (nth 0 pnt1)
   (- (nth 1 pnt1) (* sthick 2))
   (nth 2 pnt1))
   (list
   (- (nth 0 pnt5) (* sthick 2))
   (+ (nth 1 pnt5) (* sthick 2))
   (nth 2 pnt5)))
  (command ".REGION" "last" "")
  (setq obj4 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj3 "" obj4 "")
  (setq obj3 (ssadd (entlast)))
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj3 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
  ; inc x coord
  (setq xpt (+ xpt xinc))
  (setq xpnt
   (list xpt (nth 1 xpnt)
   (nth 2 xpnt)))
  ; inc ang
  (setq ang (+ ang anginc))
 )
 (command ".ZOOM" "e")
 ; loft sections
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 (setvar "DELOBJ" 1)
 ; set LOFTNORMALS =1 for smooth
 ; =0 for ruled
 (setvar "LOFTNORMALS" 1)
 ; set LOFTPARAM to default
 (setvar "LOFTPARAM" 7)
 ; loft sections
 (command ".LOFT" flist "" "")
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Convert the frames model to a totally enclosed solid surface model

A simple variation of function PROG22, partial section, is to include the bottom surface also. Since the LOFT command will not replicate the edge only, account for the void in the center, another approach needs to be taken.

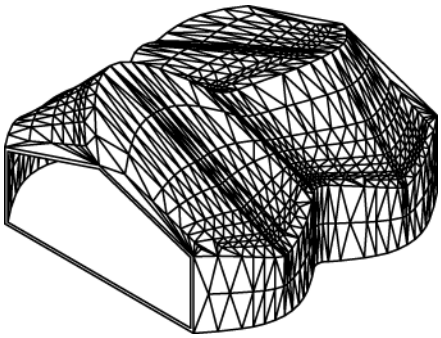

Figure 5.27e: PROG23, converting frames model to a solid surface model, bottom surface included

The sections are created as in PROG22, including the offset, but each section, inside and outside is placed into their own selection list. Each list is then LOFTed into a solid volume and the inside solid is subtracted from the outside solid.

Start function PROG23 with a copy of PROG22.

The outline for PROG23 is as follows:

```
;-----------------------------------
(defun prog23 ()
 ; input and initial computations as
 ; in PROG22
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE on the inside edge
   .
  ; place the PLINE in a selection list
   .
  ; OFFSET the PLINE for the thickness
   .
  ; place the OFFSET in a selection list
   .
  ; make inside section a REGION
   .
  ; rotate section into the YZ plane
   .
  ; add to LOFT inside selection list
   .
  ; make outside section a REGION
   .
  ; rotate section into the YZ plane
   .
  ; add to LOFT outside selection list
   .
  ; inc for next section
   .
 )
 ; LOFT inside sections
  .
 ; place inside solid into a selection
 ;  list
  .
 ; LOFT outside sections
  .
 ; place outside solid into a selection
 ;  list
  .
 ; subtract inside solid from outside
 ;  solid
  .
)
;-----------------------------------
```

Review this sequence of commands to create each section in the completed PROG23.

The script file for PROG23 is the same as for PROG22.

Completed function PROG23:

```
;-----------------------------------
(defun prog23 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG23: ")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 (setq sthick
  (getdist
  "\nEnter surface thickness:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; selection list of frames
 (setq flist1 (ssadd))
 (setq flist2 (ssadd))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
  (setq linepart2
   (* (* linelen (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newlinelen
   (+ linepart1 linepart2))
  ; compute endpoint
  (setq epnt1
   (polar xpnt (dtr 90) newlinelen))
  (setq epnt2
   (polar xpnt (dtr 270) newlinelen))
  ; compute edge height
  (setq linepart1 (* zedge linefact))
  (setq linepart2
   (* (* zedge (- 1.0 linefact))
   (abs (cos (dtr ang)))))
  (setq newzedge
   (+ linepart1 linepart2))
  ; compute midpoint height
  (setq linepart1 (* zmid linefact))
  (setq linepart2
   (* (* zmid (- 1.0 linefact))
```

```
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
 (setq pnt2
  (list (+ (nth 0 epnt2) newzedge)
  (nth 1 epnt2) 0.0))
 (setq pnt3
  (list (+ (nth 0 xpnt) newzmid)
  (+ (nth 1 xpnt) newyoff) 0.0))
 (setq pnt4
  (list (+ (nth 0 epnt1) newzedge)
  (nth 1 epnt1) 0.0))
 (setq pnt5
  (list (nth 0 epnt1)
   (nth 1 epnt1) 0.0))
 ; create inside edge
 (command ".PLINE" pnt1 pnt2 pnt3
  pnt4 pnt5 "c")
 (setq obj1 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; offset
 (command ".OFFSET" sthick pnt1
  (list (nth 0 pnt5)
  (+ (nth 1 pnt5) sthick) 0.0) "")
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; inside
 (command ".REGION" obj1 "")
 (setq obj1 (ssadd (entlast)))
 ; rotate in Y axis
 (command ".ROTATE3D"
  obj1 "" "y" pnt1 "-90")
 (command ".ZOOM" "e")
 ; add to list
 (setq flist1
  (ssadd (entlast) flist1))
 ; outside
 (command ".REGION" obj2 "")
 (setq obj2 (ssadd (entlast)))
 ; rotate in Y axis
 (command ".ROTATE3D"
  obj2 "" "y" pnt1 "-90")
 (command ".ZOOM" "e")
 ; add to list
 (setq flist2
  (ssadd (entlast) flist2))
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
(command ".ZOOM" "e")
; loft sections
; turn on history
(setvar "SOLIDHIST" 0)
; set DELOBJ to delete the sections
(setvar "DELOBJ" 1)
; set LOFTNORMALS =1 for smooth
; =0 for ruled
(setvar "LOFTNORMALS" 1)
; set LOFTPARAM to default
(setvar "LOFTPARAM" 7)
; loft sections
(command ".LOFT" flist1 "" "")
(setq obj1 (ssadd (entlast)))
(command ".LOFT" flist2 "" "")
(setq obj2 (ssadd (entlast)))
; subtract
 (command
  ".SUBTRACT" obj2 "" obj1 "")
; turn on history
(setvar "SOLIDHIST" 1)
; set extrude height
(command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Convert the frames model to a totally enclosed solid surface model with curved corners

The corners in this closed section come to sharp point. A variation of this connection would be to curve or fillet these corners.

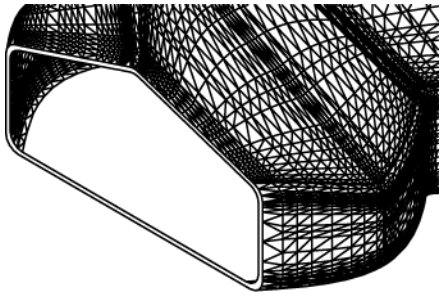

Figure 5.27f: PROG23a, converting frames model to a solid surface model, bottom surface included, section filleted

Start function PROG23a with a copy of PROG23.

The outline for PROG23a is as follows:

```
;-----------------------------------
(defun prog23a ()
 ; input and initial computations as
 ; in PROG23
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE on the inside edge
   .
  ; place the PLINE in a selection list
   .
  ; OFFSET the PLINE for the thickness
   .
  ; place the OFFSET in a selection list
   .
  ; FILLET the inside section
   .
  ; FILLET the outside section
   .
  ; make inside section a REGION
   .
  ; rotate section into the YZ plane
   .
  ; add to LOFT inside selection list
   .
  ; make outside section a REGION
   .
  ; rotate section into the YZ plane
   .
  ; add to LOFT outside selection list
   .
  ; inc for next section
   .
 )
 ; LOFT inside sections
  .
 ; place inside solid into a selection
 ;  list
  .
 ; LOFT outside sections
  .
 ; place outside solid into a selection
 ;  list
  .
 ; subtract inside solid from outside
 ;  solid
  .
```

```
)
;-----------------------------------
```

The only addition to PROG23 is the FILLET commands. The FILLET command first sets the radius then converts the corners to arcs with the polyline option.

Change:

```
; offset
(command ".OFFSET" sthick pnt1
(list (nth 0 pnt5)
 (+ (nth 1 pnt5) sthick) 0.0) "")
(setq obj2 (ssadd (entlast)))
(command ".ZOOM" "e")
; inside
(command ".REGION" obj1 "")
(setq obj1 (ssadd (entlast)))
```

To:

```
; offset
(command ".OFFSET" sthick pnt1
(list (nth 0 pnt5)
 (+ (nth 1 pnt5) sthick) 0.0) "")
(setq obj2 (ssadd (entlast)))
(command ".ZOOM" "e")
; fillet inside
(setvar "FILLETRAD" crad)
(command ".FILLET" "p" obj1)
; fillet outside
(command ".FILLET" "p" obj2)
; inside
(command ".REGION" obj1 "")
(setq obj1 (ssadd (entlast)))
```

Note that after the FILLETs are made, a new selection list is not needed. A new object is not created, the current one is just redefined.

Sample script file for PROG23a:

```
;-----------------------------------
(prog23a)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
40
; keep line factor
0.75
; cycles
1.0
; thickness
6"
; corner radius
18"
;-----------------------------------
```

The fillet radius will have some limits as related to the surface thickness. If the solid is not created correctly, a void is not included, lower the radius value or set it to zero.

Convert the frames model to a solid volume segmented model

Another approach to representing the frames as a solid model is to extrude each computed section, developing a solid segmented model.

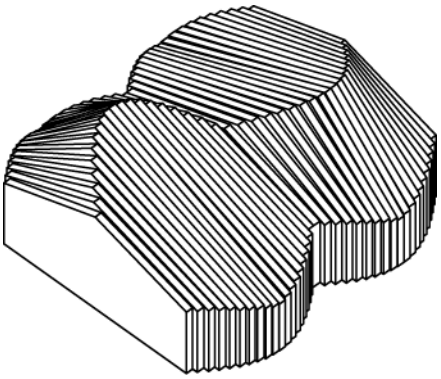

Figure 5.27g: PROG24, converting frames model to a solid segmented model, 40 segments

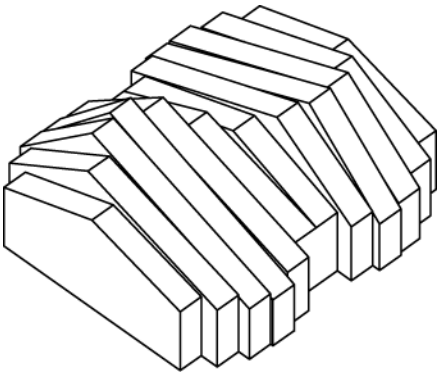

Figure 5.27h: PROG24, converting frames model to a solid segmented model, 12 segments

Sample script file for PROG24:

```
;-----------------------------------
(prog24)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
40
; keep line factor
0.75
; cycles
1.0
;-----------------------------------
```

Start function PROG24 with a copy of PROG22.

The outline for PROG24 is as follows:

```
;-----------------------------------
(defun prog24 ()
 ; input and initial computations as
 ; in PROG22
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; rotate section into the YZ plane
   .
  ; extrude
   .
  ; add to UNION list
```

```
   .
  ; inc for next section
   .
 )
 ; UNION sections
  .
)
;-----------------------------------
```

The OFFSET, REGION and SUBTRACT commands are removed and replace by a simple EXTRUDE command. All the EXTRUDEs are placed in a selection list so they can be UNIONed. The LOFT commands are removed.

Completed function PROG24:

```
;-----------------------------------
(defun prog24 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG24:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; selection list of frames
 (setq flist (ssadd))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
  (setq linepart2
   (* (* linelen (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newlinelen
   (+ linepart1 linepart2))
  ; compute endpoint
  (setq epnt1
   (polar xpnt (dtr 90) newlinelen))
  (setq epnt2
   (polar xpnt (dtr 270) newlinelen))
  ; compute edge height
  (setq linepart1 (* zedge linefact))
  (setq linepart2
   (* (* zedge (- 1.0 linefact))
   (abs (cos (dtr ang)))))
  (setq newzedge
   (+ linepart1 linepart2))
  ; compute midpoint height
  (setq linepart1 (* zmid linefact))
  (setq linepart2
   (* (* zmid (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newzmid
   (+ linepart1 linepart2))
  ; compute midpoint offset
  (setq newyoff
   (* yoff (sin (dtr ang))))
  ; compute frame points
  (setq pnt1
   (list (nth 0 epnt2)
    (nth 1 epnt2) 0.0))
  (setq pnt2
   (list (+ (nth 0 epnt2) newzedge)
   (nth 1 epnt2) 0.0))
  (setq pnt3
   (list (+ (nth 0 xpnt) newzmid)
   (+ (nth 1 xpnt) newyoff) 0.0))
  (setq pnt4
   (list (+ (nth 0 epnt1) newzedge)
   (nth 1 epnt1) 0.0))
  (setq pnt5
   (list (nth 0 epnt1)
    (nth 1 epnt1) 0.0))
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj1 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; extrude
  (command ".EXTRUDE" obj1 ""
   xinc)
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" obj3 ""
  ; xinc "0")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
  ; inc x coord
  (setq xpt (+ xpt xinc))
  (setq xpnt
   (list xpt (nth 1 xpnt)
   (nth 2 xpnt)))
  ; inc ang
  (setq ang (+ ang anginc))
 )
 ; union segments
 (command ".UNION" flist "")
 (command ".ZOOM" "e")
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Instead of extruding the entire section, a partial section can be extruded.

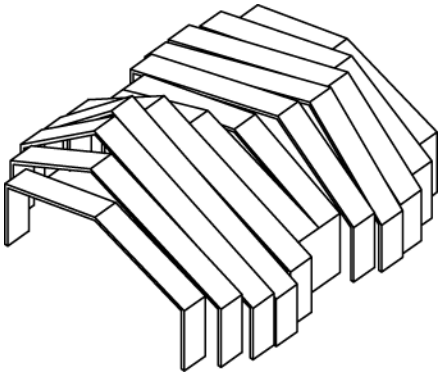

Figure 5.27i: PROG24a, converting frames model to a solid surface segmented model, 12 segments

Sample script file for PROG24a:

```
;-----------------------------------
(prog24a)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; thickness
6"
;-----------------------------------
```

Start function PROG24a with a copy of PROG22.

The outline for PROG24a is as follows:

```
;-----------------------------------
(defun prog24a ()
 ; input and initial computations as
 ; in PROG22
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; place the PLINE in a selection list
   .
  ; OFFSET the PLINE for the thickness
   .
  ; place the OFFSET in a selection list
   .
  ; make each selection a REGION
   .
  ; subtract the inside from the outside
   .
  ; make a rectangular REGION at the base
   .
  ; subtract rectangle from the section
   .
  ; rotate section into the YZ plane
   .
  ; extrude
   .
  ; add to UNION list
   .
  ; inc for next section
   .
 )
 ; UNION sections
  .
)
;-----------------------------------
```

A simple EXTRUDE command is added to the section. All the EXTRUDEs are placed in a selection list so they can be UNIONed. The LOFT commands are removed.

Completed function PROG24a:

```
;-----------------------------------
(defun prog24a ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG24a:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 (setq sthick
  (getdist
  "\nEnter surface thickness:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; selection list of frames
 (setq flist (ssadd))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
  (setq linepart2
   (* (* linelen (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newlinelen
   (+ linepart1 linepart2))
  ; compute endpoint
  (setq epnt1
   (polar xpnt (dtr 90) newlinelen))
  (setq epnt2
   (polar xpnt (dtr 270) newlinelen))
  ; compute edge height
  (setq linepart1 (* zedge linefact))
  (setq linepart2
```

```
 (* (* zedge (- 1.0 linefact))
 (abs (cos (dtr ang)))))
(setq newzedge
 (+ linepart1 linepart2))
; compute midpoint height
(setq linepart1 (* zmid linefact))
(setq linepart2
 (* (* zmid (- 1.0 linefact))
 (abs (sin (dtr ang)))))
(setq newzmid
 (+ linepart1 linepart2))
; compute midpoint offset
(setq newyoff
 (* yoff (sin (dtr ang))))
; compute frame points
(setq pnt1
 (list (nth 0 epnt2)
  (nth 1 epnt2) 0.0))
(setq pnt2
 (list (+ (nth 0 epnt2) newzedge)
 (nth 1 epnt2) 0.0))
(setq pnt3
 (list (+ (nth 0 xpnt) newzmid)
 (+ (nth 1 xpnt) newyoff) 0.0))
(setq pnt4
 (list (+ (nth 0 epnt1) newzedge)
 (nth 1 epnt1) 0.0))
(setq pnt5
 (list (nth 0 epnt1)
  (nth 1 epnt1) 0.0))
; create inside edge
(command ".PLINE" pnt1 pnt2 pnt3
 pnt4 pnt5 "c")
(setq obj1 (ssadd (entlast)))
(command ".ZOOM" "e")
; offset
(command ".OFFSET" sthick pnt1
 (list (nth 0 pnt5)
 (+ (nth 1 pnt5) sthick) 0.0) "")
(setq obj2 (ssadd (entlast)))
(command ".ZOOM" "e")
; make regions and subtract
(command ".REGION" obj1 "")
(setq obj1 (ssadd (entlast)))
(command ".REGION" obj2 "")
(setq obj2 (ssadd (entlast)))
(command
 ".SUBTRACT" obj2 "" obj1 "")
(setq obj3 (ssadd (entlast)))
; remove base
(command ".RECTANGLE"
 (list
 (nth 0 pnt1)
 (- (nth 1 pnt1) (* sthick 2))
 (nth 2 pnt1))
 (list
 (- (nth 0 pnt5) (* sthick 2))
 (+ (nth 1 pnt5) (* sthick 2))
 (nth 2 pnt5)))
(command ".REGION" "last" "")
(setq obj4 (ssadd (entlast)))
(command
 ".SUBTRACT" obj3 "" obj4 "")
(setq obj3 (ssadd (entlast)))
; rotate in Y axis
(command ".ROTATE3D"
 obj3 "" "y" pnt1 "-90")
(command ".ZOOM" "e")
; extrude
(command ".EXTRUDE" obj3 ""
 xinc)
; for version prior to 2007 use
;(command ".EXTRUDE" obj3 ""
; xinc "0")
; add to list
(setq flist
 (ssadd (entlast) flist))
; inc x coord
(setq xpt (+ xpt xinc))
(setq xpnt
 (list xpt (nth 1 xpnt)
 (nth 2 xpnt)))
; inc ang
(setq ang (+ ang anginc))
 )
 ; union segments
 (command ".UNION" flist "")
 (command ".ZOOM" "e")
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Convert the frames model to a solid model with section converted to curves and splines

By defining the section as a polyline, we can use the endpoints of the line segments as control points for a curve fit or a spline. The PEDIT command can convert line polylines to curves.

The number of control points will determine the shape of the curve created.

In this diagram the normally computed section is converted to a curve and two types of splines,

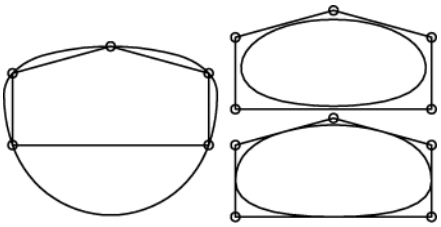

Figure 5.27i1: five control points as curve fit and splines

The curve fit conversion creates a smooth curve with a series of arcs placed between the given points.

For the spline conversion, the SPLINETYPE system variable can be set to 6 for a Cubic B-spline and 5 for a Quadratic B-spline; 6 is the default value. The Cubic B-spline uses three adjacent points to determine curvature; the Quadratic B-spline uses four.

The curve fit and the Cubic B-spline do not engage the edge of the section.

If a midpoint at the bottom edge is added, then they all engage the section.

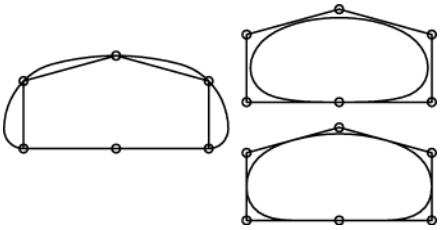

Figure 5.27i2: six control points for curve fit and splines

If other points are added the curves can be made to further engage the edges of the section. Try others.

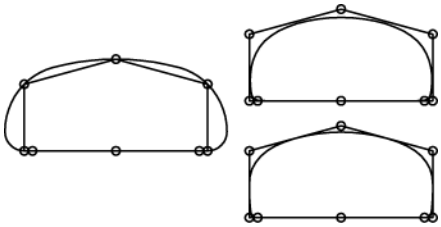

Figure 5.27i3: eight control points as curve fit and splines

For function PROG25, this conversion of the straight line frame to a curve, either fit or spline, an extra point is specified in the polyline at the bottom midpoint so to make sure the bottom edge is not effected by the curve conversion.

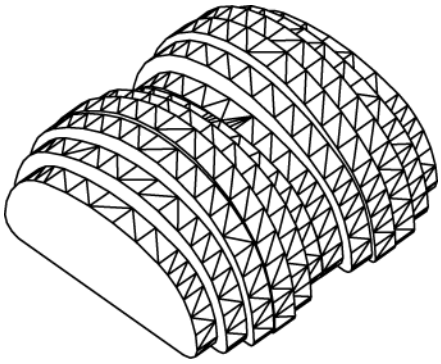

Figure 5.27j: PROG25, converting frames model to a solid segmented model, 12 segments, curve fit

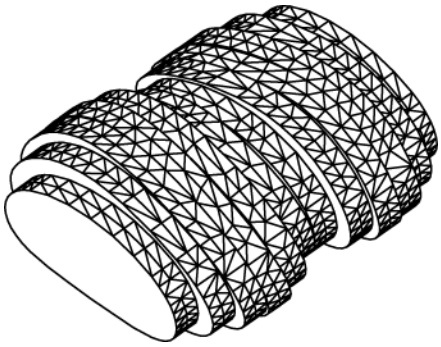

Figure 5.27k: PROG25, converting frames model to a solid segmented model, 12 segments, spline fit

The conversion to a curve is made with the PEDIT command:

```
; convert polyline to a curve
; or spline
(setvar "SPLINETYPE" 6)
(command ".PEDIT" obj1 "f" "")
```

When the "f" option, curve fit, is changed to a "s", spline, the SPLINETYPE can then be set to a value of 6 or 5.

Once the section is converted to a curve and extruded, any portion below the ground is subtracted. This would be required if a midpoint is not added to the bottom edge or in some other section configuration that is generated.

Start function PROG25 with a copy of PROG24.

The outline for PROG25 is as follows:

```
;-----------------------------------
(defun prog25 ()
 ; input and initial computations as
 ; in PROG24
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; convert polyline to a curve
  ; or spline
   .
  ; rotate section into the YZ plane
   .
  ; extrude
   .
  ; remove base
   .
  ; add to UNION list
   .
  ; inc for next section
   .
 )
 ; UNION sections
  .
)
;-----------------------------------
```

Sample script file for PROG25:

```
;-----------------------------------
(prog25)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
;-----------------------------------
```

Completed function PROG25:

```
;-----------------------------------
(defun prog25 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG25:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
```

```
 (getreal
 "\nNumber of curve cycles:"))
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; selection list of frames
(setq flist (ssadd))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
 (setq pnt2
  (list (+ (nth 0 epnt2) newzedge)
  (nth 1 epnt2) 0.0))
 (setq pnt3
  (list (+ (nth 0 xpnt) newzmid)
  (+ (nth 1 xpnt) newyoff) 0.0))
 (setq pnt4
  (list (+ (nth 0 epnt1) newzedge)
  (nth 1 epnt1) 0.0))
 (setq pnt5
  (list (nth 0 epnt1)
   (nth 1 epnt1) 0.0))
 ; create inside edge
 (command ".PLINE" pnt1 pnt2 pnt3
  pnt4 pnt5 xpnt "c")
 (setq obj1 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; convert polyline to a curve
 ; or spline
 (setvar "SPLINETYPE" 6)
 (command ".PEDIT" obj1 "f" "")
 ; rotate in Y axis
 (command ".ROTATE3D"
  obj1 "" "y" pnt1 "-90")
 (command ".ZOOM" "e")
  ; extrude
  (command ".EXTRUDE" obj1 ""
   xinc)
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" obj3 ""
  ; xinc "0")
  (setq obj1 (ssadd (entlast)))
  ; subtract base
  (command ".POLYGON" "4" xpnt "c"
   (* linelen 3))
  (command ".EXTRUDE" "last" ""
   (* linelen -3))
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" "last" ""
  ; (* linelen -3) "0")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj1 "" obj2 "")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
  ; inc x coord
  (setq xpt (+ xpt xinc))
  (setq xpnt
   (list xpt (nth 1 xpnt)
   (nth 2 xpnt)))
  ; inc ang
  (setq ang (+ ang anginc))
 )
 ; union segments
 (command ".UNION" flist "")
 (command ".ZOOM" "e")
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

The partially extruded section can also be converted to a curve.

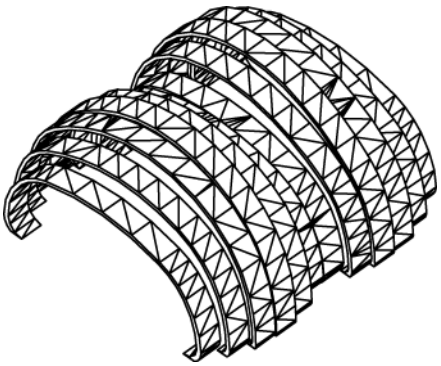

Figure 5.27l: PROG25a, converting frames model to a solid surface segmented model, 12 segments, curve fit

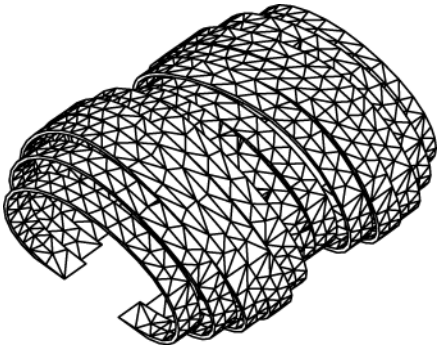

Figure 5.27m: PROG25a, converting frames model to a solid surface segmented model, 12 segments, spline fit

Start function PROG25a with a copy of PROG24a.

The outline for PROG25a is as follows:

```
;-----------------------------------
(defun prog25a ()
 ; input and initial computations as
 ; in PROG24a
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; place the PLINE in a selection list
   .
  ; OFFSET the PLINE for the thickness
   .
  ; place the OFFSET in a selection list
   .
  ; convert each polyline to a curve
  ; or spline
   .
  ; make each selection a REGION
   .
  ; subtract the inside from the outside
   .
  ; make a rectangular REGION at the base
   .
  ; subtract rectangle from the section
   .
  ; rotate section into the YZ plane
   .
  ; extrude
   .
  ; add to UNION list
   .
  ; inc for next section
   .
 )
 ; UNION sections
  .
)
;-----------------------------------
```

The conversion of the polyline to a curve is completed in the same fashion as outlined in PROG25:

```
; convert polyline to a curve
; or spline
(setvar "SPLINETYPE" 6)
(command ".PEDIT" obj1 "f" "")
(command ".PEDIT" obj2 "f" "")
```

The subtraction of the base is modified from:

```
; remove base
(command ".RECTANGLE"
 (list
  (nth 0 pnt1)
  (- (nth 1 pnt1) (* sthick 10))
  (nth 2 pnt1))
 (list
  (- (nth 0 pnt5) (* linelen 10))
  (+ (nth 1 pnt5) (* sthick 10))
  (nth 2 pnt5)))
```

To:

```
; remove base
(command ".RECTANGLE"
 (list
  (+ (nth 0 pnt1) (* sthick 0.1))
  (- (nth 1 pnt1) (* sthick 10))
  (nth 2 pnt1))
 (list
  (- (nth 0 pnt5) (* linelen 10))
  (+ (nth 1 pnt5) (* sthick 10))
  (nth 2 pnt5)))
```

Sample script file for PROG25a:

```
;-----------------------------------
(prog25a)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; thickness
6"
;-----------------------------------
```

Completed function PROG25a:

```
;-----------------------------------
(defun prog25a ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG25a:")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 (setq sthick
  (getdist
  "\nEnter surface thickness:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; selection list of frames
 (setq flist (ssadd))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
```

```
(setq linepart2
 (* (* linelen (- 1.0 linefact))
 (abs (sin (dtr ang)))))
(setq newlinelen
 (+ linepart1 linepart2))
; compute endpoint
(setq epnt1
 (polar xpnt (dtr 90) newlinelen))
(setq epnt2
 (polar xpnt (dtr 270) newlinelen))
; compute edge height
(setq linepart1 (* zedge linefact))
(setq linepart2
 (* (* zedge (- 1.0 linefact))
 (abs (cos (dtr ang)))))
(setq newzedge
 (+ linepart1 linepart2))
; compute midpoint height
(setq linepart1 (* zmid linefact))
(setq linepart2
 (* (* zmid (- 1.0 linefact))
 (abs (sin (dtr ang)))))
(setq newzmid
 (+ linepart1 linepart2))
; compute midpoint offset
(setq newyoff
 (* yoff (sin (dtr ang))))
; compute frame points
(setq pnt1
 (list (nth 0 epnt2)
  (nth 1 epnt2) 0.0))
(setq pnt2
 (list (+ (nth 0 epnt2) newzedge)
 (nth 1 epnt2) 0.0))
(setq pnt3
 (list (+ (nth 0 xpnt) newzmid)
 (+ (nth 1 xpnt) newyoff) 0.0))
(setq pnt4
 (list (+ (nth 0 epnt1) newzedge)
 (nth 1 epnt1) 0.0))
(setq pnt5
 (list (nth 0 epnt1)
  (nth 1 epnt1) 0.0))
; create inside edge
(command ".PLINE" pnt1 pnt2 pnt3
 pnt4 pnt5 xpnt "c")
(setq obj1 (ssadd (entlast)))
(command ".ZOOM" "e")
; offset
(command ".OFFSET" sthick pnt1
 (list (nth 0 pnt5)
 (+ (nth 1 pnt5) sthick) 0.0) "")
(setq obj2 (ssadd (entlast)))
(command ".ZOOM" "e")
; convert polyline to a curve
; or spline
(setvar "SPLINETYPE" 6)
(command ".PEDIT" obj1 "f" "")
(command ".PEDIT" obj2 "f" "")
; make regions and subtract
(command ".REGION" obj1 "")
(setq obj1 (ssadd (entlast)))
(command ".REGION" obj2 "")
(setq obj2 (ssadd (entlast)))
(command
 ".SUBTRACT" obj2 "" obj1 "")
(setq obj3 (ssadd (entlast)))
; remove base
(command ".RECTANGLE"
 (list
  (+ (nth 0 pnt1) (* sthick 0.1))
  (- (nth 1 pnt1) (* sthick 10))
 (nth 2 pnt1))
 (list
 (- (nth 0 pnt5) (* linelen 10))
 (+ (nth 1 pnt5) (* sthick 10))
 (nth 2 pnt5)))
(command ".REGION" "last" "")
(setq obj4 (ssadd (entlast)))
(command
 ".SUBTRACT" obj3 "" obj4 "")
(setq obj3 (ssadd (entlast)))
; rotate in Y axis
(command ".ROTATE3D"
  obj3 "" "y" pnt1 "-90")
 (command ".ZOOM" "e")
 ; extrude
 (command ".EXTRUDE" obj3 ""
  xinc)
 ; for version prior to 2007 use
 ;(command ".EXTRUDE" obj3 ""
 ; xinc "0")
 ; add to list
 (setq flist
  (ssadd (entlast) flist))
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
; union segments
(command ".UNION" flist "")
(command ".ZOOM" "e")
; set extrude height
(command ".ELEV" 0.0 0.0)
(princ)
)
;-----------------------------------
```

Convert the frames model to a solid volume model with curved sections

The conversion of the frame sections to a curved section can also be used in the original example of creating a complete solid with the use of the LOFT command.

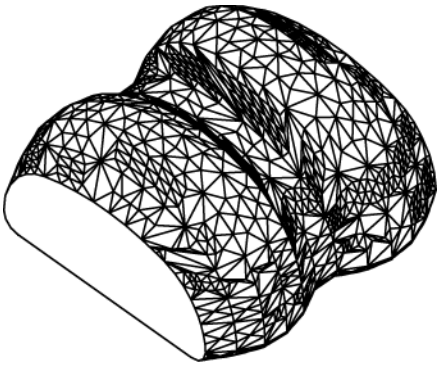

Figure 5.27n: PROG26, converting frames model to a solid model, 12 segments, curve fit

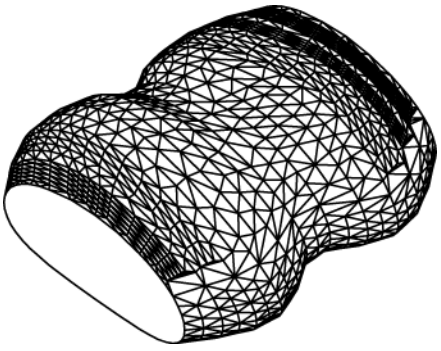

Figure 5.27o: PROG26, converting frames model to a solid model, 12 segments, spline fit

Start function PROG26 with a copy of PROG25.

The outline for PROG26 is as follows:

```
;-----------------------------------
```

```
(defun prog26 ()
 ; input and initial computations as
 ; in PROG25
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; convert polyline to a curve
  ; or spline
   .
  ; rotate section into the YZ plane
   .
  ; make into a REGION
   .
  ; remove base
   .
  ; add to LOFT selection list
   .
  ; inc for next section
   .
 )
 ; LOFT sections
  .
)
;-----------------------------------
```

The differences include creating the section into a REGION instead of EXTRUDEing it and replacing the UNION of the sections with the previously used LOFT sequence of commands from PROG21.

Change:

```
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 xpnt "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; convert polyline to a curve
  ; or spline
  (setvar "SPLINETYPE" 6)
  (command ".PEDIT" obj1 "f" "")
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj1 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; extrude
  (command ".EXTRUDE" obj1 ""
   xinc)
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" obj3 ""
  ; xinc "0")
  (setq obj1 (ssadd (entlast)))
  ; subtract base
  (command ".POLYGON" "4" xpnt "c"
   (* linelen 3))
  (command ".EXTRUDE" "last" ""
   (* linelen -3))
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" "last" ""
  ; (* linelen -3) "0")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj1 "" obj2 "")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
```

To:

```
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 xpnt "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; convert polyline to a curve
  ; or spline
  (setvar "SPLINETYPE" 6)
  (command ".PEDIT" obj1 "f" "")
  ; make regions
  (command ".REGION" obj1 "")
  (setq obj1 (ssadd (entlast)))
  ; remove base
  (command ".RECTANGLE"
   (list
   (nth 0 pnt1)
   (- (nth 1 pnt1) (* sthick 2))
   (nth 2 pnt1))
   (list
   (- (nth 0 pnt5) (* linelen 2))
   (+ (nth 1 pnt5) (* sthick 2))
   (nth 2 pnt5)))
  (command ".REGION" "last" "")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj1 "" obj2 "")
  (setq obj1 (ssadd (entlast)))
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj1 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
```

Change:

```
; union segments
 (command ".UNION" flist "")
 (command ".ZOOM" "e")
```

To:

```
 (command ".ZOOM" "e")
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 (setvar "DELOBJ" 1)
 ; set LOFTNORMALS =1 for smooth
 ; =0 for ruled
 (setvar "LOFTNORMALS" 0)
 ; set LOFTPARAM to default
 (setvar "LOFTPARAM" 7)
 ; loft sections
 (command ".LOFT" flist "" "")
 ; turn on history
 (setvar "SOLIDHIST" 1)
```

Optionally, change the PEDIT curve type from "f" to "s", and when a spline, the spline type system variable can be changed to 6 or 5.

Sample script file for PROG26:

```
;-----------------------------------
(prog26)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
;-----------------------------------
```

Completed function PROG26:

```
;-----------------------------------
(defun prog26 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG26:")
```

```
(setq spnt
 (getpoint "\nPick start point:"))
(setq cpnt
 (getcorner spnt
 "\nPick end point:"))
(setq zedge
 (getdist spnt
 "\nEnter edge height:"))
(setq zmid
 (getdist spnt
 "\nEnter midpoint height:"))
(setq yoff
 (getdist spnt
 "\nEnter midpoint offset:"))
(setq numtimes
 (getint
 "\nEnter number of lines:"))
(setq linefact
 (getreal
 "\nEnter keep line factor:"))
(setq numcycles
 (getreal
 "\nNumber of curve cycles:"))
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; selection list of frames
(setq flist (ssadd))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
 (setq pnt2
  (list (+ (nth 0 epnt2) newzedge)
  (nth 1 epnt2) 0.0))
 (setq pnt3
   (list (+ (nth 0 xpnt) newzmid)
   (+ (nth 1 xpnt) newyoff) 0.0))
 (setq pnt4
  (list (+ (nth 0 epnt1) newzedge)
  (nth 1 epnt1) 0.0))
 (setq pnt5
  (list (nth 0 epnt1)
   (nth 1 epnt1) 0.0))
 ; create inside edge
 (command ".PLINE" pnt1 pnt2 pnt3
  pnt4 pnt5 xpnt "c")
 (setq obj1 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; convert polyline to a curve
 ; or spline
 (command ".PEDIT" obj1 "f" "")
 ; make regions
 (command ".REGION" obj1 "")
 (setq obj1 (ssadd (entlast)))
 ; remove base
 (command ".RECTANGLE"
  (list
  (nth 0 pnt1)
  (- (nth 1 pnt1) (* sthick 2))
  (nth 2 pnt1))
  (list
  (- (nth 0 pnt5) (* linelen 2))
  (+ (nth 1 pnt5) (* sthick 2))
  (nth 2 pnt5)))
 (command ".REGION" "last" "")
 (setq obj2 (ssadd (entlast)))
 (command
  ".SUBTRACT" obj1 "" obj2 "")
 (setq obj1 (ssadd (entlast)))
 ; rotate in Y axis
 (command ".ROTATE3D"
  obj1 "" "y" pnt1 "-90")
 (command ".ZOOM" "e")
 ; add to list
 (setq flist
  (ssadd (entlast) flist))
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
(command ".ZOOM" "e")
; turn off history
(setvar "SOLIDHIST" 0)
; set DELOBJ to delete the sections
(setvar "DELOBJ" 1)
; set LOFTNORMALS =1 for smooth
; =0 for ruled
(setvar "LOFTNORMALS" 0)
; set LOFTPARAM to default
(setvar "LOFTPARAM" 7)
; loft sections
(command ".LOFT" flist "" "")
; turn on history
(setvar "SOLIDHIST" 1)
; set extrude height
(command ".ELEV" 0.0 0.0)
(princ)
)
;-----------------------------------
```

The solid curved volume model can also be represented as a solid surface model.

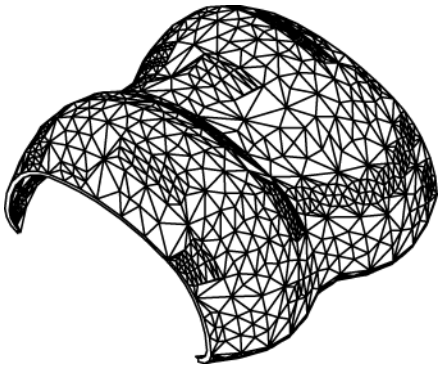

Figure 5.27p: PROG26a, converting frames model to a solid surface model, 12 segments, curve fit

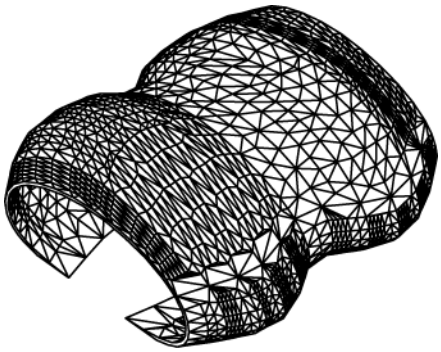

Figure 5.27q: PROG26a, converting frames model to a solid surface model, 12 segments, as a spline

Start function PROG26a with a copy of PROG25a.

The outline for PROG26a is as follows:

```
;-----------------------------------
(defun prog26a ()
 ; input and initial computations as
 ; in PROG25a
  .
 (repeat numtimes
  ; compute frame points in the XY plane
   .
  ; create a PLINE of the inside edge
   .
  ; place the PLINE in a selection list
   .
  ; OFFSET the PLINE for the thickness
   .
  ; place the OFFSET in a selection list
   .
  ; convert each polyline to a curve
  ; or spline
   .
  ; make each selection a REGION
   .
  ; subtract the inside from the outside
   .
  ; make a rectangular REGION at the base
   .
  ; subtract rectangle from the section
   .
  ; rotate section into the YZ plane
   .
  ; add to LOFT selection list
   .
  ; inc for next section
   .
 )
 ; LOFT sections
  .
)
;-----------------------------------
```

Change the creation of the section from:

```
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 xpnt "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; offset
  (command ".OFFSET" sthick pnt1
   (list (nth 0 pnt5)
   (+ (nth 1 pnt5) sthick) 0.0) "")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; convert polyline to a curve
  ; or spline
  (setvar "SPLINETYPE" 6)
  (command ".PEDIT" obj1 "f" "")
  (command ".PEDIT" obj2 "f" "")
  ; make regions and subtract
  (command ".REGION" obj1 "")
  (setq obj1 (ssadd (entlast)))
  (command ".REGION" obj2 "")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj2 "" obj1 "")
  (setq obj3 (ssadd (entlast)))
  ; remove base
  (command ".RECTANGLE"
   (list
    (+ (nth 0 pnt1) (* sthick 0.1))
    (- (nth 1 pnt1) (* sthick 10))
   (nth 2 pnt1))
   (list
   (- (nth 0 pnt5) (* linelen 10))
   (+ (nth 1 pnt5) (* sthick 10))
   (nth 2 pnt5)))
  (command ".REGION" "last" "")
  (setq obj4 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj3 "" obj4 "")
  (setq obj3 (ssadd (entlast)))
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj3 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; extrude
  (command ".EXTRUDE" obj3 ""
   xinc)
  ; for version prior to 2007 use
  ;(command ".EXTRUDE" obj3 ""
  ; xinc "0")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
```

To:

```
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 xpnt "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; offset
  (command ".OFFSET" sthick pnt1
   (list (nth 0 pnt5)
   (+ (nth 1 pnt5) sthick) 0.0) "")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; convert polyline to a curve
  ; or spline
  (setvar "SPLINETYPE" 6)
  (command ".PEDIT" obj1 "f "")
  (command ".PEDIT" obj2 "f" "")
  ; make regions and subtract
  (command ".REGION" obj1 "")
  (setq obj1 (ssadd (entlast)))
  (command ".REGION" obj2 "")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj2 "" obj1 "")
  (setq obj3 (ssadd (entlast)))
```

```
  ; remove base
  (command ".RECTANGLE"
   (list
   (+ (nth 0 pnt1) (* sthick 0.1))
   (- (nth 1 pnt1) (* sthick 2))
   (nth 2 pnt1))
   (list
   (- (nth 0 pnt5) (* linelen 2))
   (+ (nth 1 pnt5) (* sthick 2))
   (nth 2 pnt5)))
  (command ".REGION" "last" "")
  (setq obj4 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj3 "" obj4 "")
  (setq obj3 (ssadd (entlast)))
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj3 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
```

Change the UNION to a LOFT:

```
 ; union segments
 (command ".UNION" flist "")
 (command ".ZOOM" "e")
```

To:

```
 (command ".ZOOM" "e")
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 (setvar "DELOBJ" 1)
 ; set LOFTNORMALS =1 for smooth
 ; =0 for ruled
 (setvar "LOFTNORMALS" 0)
 ; set LOFTPARAM to default
 (setvar "LOFTPARAM" 7)
 ; loft sections
 (command ".LOFT" flist "" "")
 ; turn on history
 (setvar "SOLIDHIST" 1)
```

Optionally, change the PEDIT curve type from "f" to "s", and when a spline, the spline type system variable can be changed to 6 or 5.

Sample script file for PROG26a:

```
;-----------------------------------
(prog26a)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; thickness
6"
;-----------------------------------
```

Completed function PROG26a:

```
;-----------------------------------
(defun prog26a ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
(prompt "\nPROG26a:")
(setq spnt
 (getpoint "\nPick start point:"))
(setq cpnt
 (getcorner spnt
 "\nPick end point:"))
(setq zedge
 (getdist spnt
 "\nEnter edge height:"))
(setq zmid
 (getdist spnt
 "\nEnter midpoint height:"))
(setq yoff
 (getdist spnt
 "\nEnter midpoint offset:"))
(setq numtimes
 (getint
 "\nEnter number of lines:"))
(setq linefact
 (getreal
 "\nEnter keep line factor:"))
(setq numcycles
 (getreal
 "\nNumber of curve cycles:"))
(setq sthick
 (getdist
 "\nEnter surface thickness:"))
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; selection list of frames
(setq flist (ssadd))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
```

```
  (setq pnt2
   (list (+ (nth 0 epnt2) newzedge)
   (nth 1 epnt2) 0.0))
  (setq pnt3
   (list (+ (nth 0 xpnt) newzmid)
   (+ (nth 1 xpnt) newyoff) 0.0))
  (setq pnt4
   (list (+ (nth 0 epnt1) newzedge)
   (nth 1 epnt1) 0.0))
  (setq pnt5
   (list (nth 0 epnt1)
    (nth 1 epnt1) 0.0))
  ; create inside edge
  (command ".PLINE" pnt1 pnt2 pnt3
   pnt4 pnt5 xpnt "c")
  (setq obj1 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; offset
  (command ".OFFSET" sthick pnt1
   (list (nth 0 pnt5)
   (+ (nth 1 pnt5) sthick) 0.0) "")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; convert polyline to a curve
  ; or spline
  (setvar "SPLINETYPE" 6)
  (command ".PEDIT" obj1 "f" "")
  (command ".PEDIT" obj2 "f" "")
  ; make regions and subtract
  (command ".REGION" obj1 "")
  (setq obj1 (ssadd (entlast)))
  (command ".REGION" obj2 "")
  (setq obj2 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj2 "" obj1 "")
  (setq obj3 (ssadd (entlast)))
  ; remove base
  (command ".RECTANGLE"
   (list
   (+ (nth 0 pnt1) (* sthick 0.1))
   (- (nth 1 pnt1) (* sthick 2))
   (nth 2 pnt1))
   (list
   (- (nth 0 pnt5) (* linelen 2))
   (+ (nth 1 pnt5) (* sthick 2))
   (nth 2 pnt5)))
  (command ".REGION" "last" "")
  (setq obj4 (ssadd (entlast)))
  (command
   ".SUBTRACT" obj3 "" obj4 "")
  (setq obj3 (ssadd (entlast)))
  ; rotate in Y axis
  (command ".ROTATE3D"
   obj3 "" "y" pnt1 "-90")
  (command ".ZOOM" "e")
  ; add to list
  (setq flist
   (ssadd (entlast) flist))
  ; inc x coord
  (setq xpt (+ xpt xinc))
  (setq xpnt
   (list xpt (nth 1 xpnt)
   (nth 2 xpnt)))
  ; inc ang
  (setq ang (+ ang anginc))
 )
 (command ".ZOOM" "e")
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 (setvar "DELOBJ" 1)
 ; set LOFTNORMALS =1 for smooth
 ; =0 for ruled
 (setvar "LOFTNORMALS" 0)
 ; set LOFTPARAM to default
 (setvar "LOFTPARAM" 7)
 ; loft sections
 (command ".LOFT" flist "" "")
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
```

```
;-----------------------------------
```

Convert the frames model to a solid members model

The same points that are computed to generate mesh points or section points of each frame can be converted to a continuous line and that made into a solid member by sweeping a section across it.

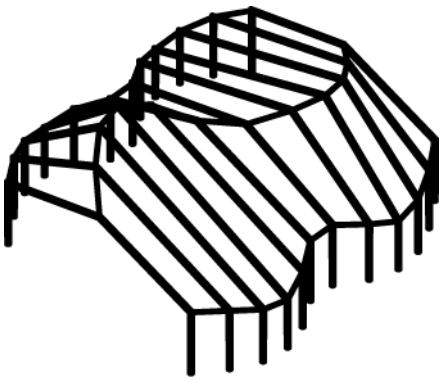

Figure 5.27r: PROG27, converting frames model to a solid members model, 12 segments

The SWEEP command is used on each frame and the horizontal connecting member with a circular section. Review the use of this command and the structure of function PROG21.

Copy PROG21 to PROG27. Remove the code section for lofting the frames at the end of the function.

Add input for frame and member section radius.

```
(setq sectrad1
 (getdist spnt
 "\nEnter frame section radius:"))
(setq sectrad2
 (getdist spnt
 "\nEnter member section radius:"))
```

Change the loft selection list from:

```
; selection list of frames
(setq flist (ssadd))
```

To starting a new point list:

```
; list of frame points
(setq flist (list ))
```

Change the adding of the 3DPOLY points from:

```
; make section
(command ".3DPOLY" pnt1 pnt2 pnt3
 pnt4 pnt5 "c")
(command ".REGION" "last" "")
; add to list
(setq flist
 (ssadd (entlast) flist))
```

To creating the section, sweeping a circular member profile and adding it to the points list:

```
; make section
(command ".3DPOLY" pnt1 pnt2
 pnt3 pnt4 pnt5 "")
(setq obj1 (ssadd (entlast)))
; add cross section
(command ".CIRCLE" pnt1 sectrad1)
(setq obj2 (ssadd (entlast)))
(command ".ZOOM" "e")
```

```
; SWEEP frame section
; turn off history
(setvar "SOLIDHIST" 0)
; set DELOBJ to delete the sections
; and path
(setvar "DELOBJ" 2)
; sweep sections
(command ".SWEEP" obj2 "" obj1)
; turn on history
(setvar "SOLIDHIST" 1)
; add points to list
(setq flist
 (append (list (list pnt1 pnt2 pnt3 pnt4
  pnt5)) flist))
```

A 3DPOLY is created from each set of frames points and a circular section is swept over it.

Once the frames are generated, the points in the list are traversed and the horizontal connecting members are generated in the same fashion as the individual frames members.

Review how 3DPOLYs are generated from the points extracted from the frame points list and then swept.

Add the following:

```
; horizontal members
(setq ipt 1)
(repeat 3
 (setq iframe 0)
 ; start 3DPOLY
 (command ".3DPOLY")
 (repeat numtimes
   ; add pt to 3DPOLY
   (setq npt
    (nth ipt (nth iframe flist)))
   (command npt)
   ; next frame
   (setq iframe (+ iframe 1))
 )
 ; close
 (command "")
 (setq obj1 (ssadd (entlast)))
 ; add cross section
 (command ".CIRCLE" npt sectrad2)
 (command ".ROTATE3D"
  "last" "" "y" npt "-90")
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; SWEEP members across frames
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
  ; and path
  (setvar "DELOBJ" 2)
  ; sweep sections
  (command ".SWEEP" obj2 "" obj1)
  ; turn on history
  (setvar "SOLIDHIST" 1)
  (command ".ZOOM" "e")
  ; next pt
  (setq ipt (+ ipt 1))
 )
```

Sample script file for PROG27:

```
;-----------------------------------
(prog27)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; frame section radius
4.5"
; member section radius
3.0"
;-----------------------------------

Completed function PROG27:

;-----------------------------------
(defun prog27 ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
 (prompt "\nPROG27: Tubular members")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 (setq sectrad1
  (getdist spnt
  "\nEnter frame section radius:"))
 (setq sectrad2
  (getdist spnt
   "\nEnter member section radius:"))
 ; compute line length,
 ; width of boundary
 (setq linelen
  (- (nth 1 cpnt) (nth 1 spnt)))
 ; compute increment across
 ; boundary
 (setq xinc
  (/ (- (nth 0 cpnt)
  (nth 0 spnt)) (- numtimes 1)))
 ; computer ang inc
 (setq anginc
  (/ (* 360.0 numcycles)
  (- numtimes 1)))
 ; set first point and
 ; first x coord
 (setq xpnt spnt)
 (setq xpt (nth 0 spnt))
 ; start ang
 (setq ang 0)
 ; list of frame points
 (setq flist (list ))
 ; loop to repeat lines
 (repeat numtimes
  ; compute line length
  (setq linepart1
   (* linelen linefact))
  (setq linepart2
   (* (* linelen (- 1.0 linefact))
   (abs (sin (dtr ang)))))
  (setq newlinelen
   (+ linepart1 linepart2))
  ; compute endpoint
  (setq epnt1
   (polar xpnt (dtr 90) newlinelen))
  (setq epnt2
```

```
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1 epnt2)
 (setq pnt2 (list
   (nth 0 epnt2)
   (nth 1 epnt2)
   (+ (nth 2 epnt2) newzedge)))
 (setq pnt3 (list
   (nth 0 xpnt)
   (+ (nth 1 xpnt) newyoff)
   (+ (nth 2 xpnt) newzmid)))
 (setq pnt4 (list
   (nth 0 epnt1)
   (nth 1 epnt1)
   (+ (nth 2 epnt1) newzedge)))
 (setq pnt5 epnt1)
 ; make section
 (command ".3DPOLY" pnt1 pnt2
  pnt3 pnt4 pnt5 "")
 (setq obj1 (ssadd (entlast)))
 ; add cross section
 (command ".CIRCLE" pnt1 sectrad1)
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; SWEEP frame section
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 ; and path
 (setvar "DELOBJ" 2)
 ; sweep sections
 (command ".SWEEP" obj2 "" obj1)
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; add to list
 (setq flist
  (append (list (list pnt1 pnt2 pnt3 pnt4
   pnt5)) flist))
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
(command ".ZOOM" "e")
; horizontal members
(setq ipt 1)
(repeat 3
 (setq iframe 0)
 ; start 3DPOLY
 (command ".3DPOLY")
 (repeat numtimes
   ; add pt to 3DPOLY
   (setq npt
    (nth ipt (nth iframe flist)))
   (command npt)
   ; next frame
   (setq iframe (+ iframe 1))
 )
 ; close
 (command "")
 (setq obj1 (ssadd (entlast)))
 ; add cross section
 (command ".CIRCLE" npt sectrad2)
 (command ".ROTATE3D"
   "last" "" "y" npt "-90")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; SWEEP members across frames
  ; turn off history
  (setvar "SOLIDHIST" 0)
  ; set DELOBJ to delete the sections
  ; and path
  (setvar "DELOBJ" 2)
  ; sweep sections
  (command ".SWEEP" obj2 "" obj1)
  ; turn on history
  (setvar "SOLIDHIST" 1)
  (command ".ZOOM" "e")
  ; next pt
  (setq ipt (+ ipt 1))
 )
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

Additional members can be added to any straight section of the frame.

Add function PROG27a. Using a copy of function PROG027, compute two midpoints in the top section of the frame.

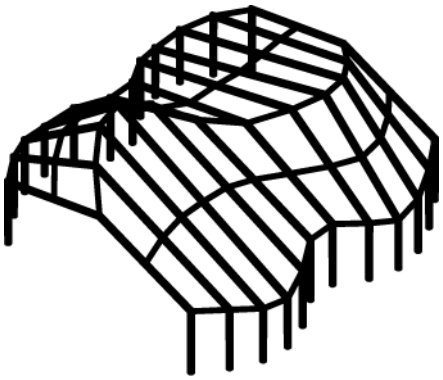

Figure 5.27s: PROG27a, converting frames model to a solid members model, 12 segments, with midpoint members

Add a midpoint computation, for example:

```
; midpt
(setq xpt (+ (nth 0 pnt2)
 (/ (- (nth 0 pnt3)
 (nth 0 pnt2)) 2)))
(setq ypt (+ (nth 1 pnt2)
 (/ (- (nth 1 pnt3)
 (nth 1 pnt2)) 2)))
(setq zpt (+ (nth 2 pnt2)
 (/ (- (nth 2 pnt3)
 (nth 2 pnt2)) 2)))
(setq pnt3a
 (list xpt ypt zpt))
```

Do the same for pnt4a, midpoint of pnt3 and pnt4. Add these two new points to the frame point list and modify the horizontal member loop from 3 times to 5 times.

The script file for PROG27a is the same as for PROG27.

Completed function PROG27a:

```
;-----------------------------------
(defun prog27a ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
  parameters
```

```
(prompt
 "\nPROG27a: Tubular members")
(setq spnt
 (getpoint "\nPick start point:"))
(setq cpnt
 (getcorner spnt
 "\nPick end point:"))
(setq zedge
 (getdist spnt
 "\nEnter edge height:"))
(setq zmid
 (getdist spnt
 "\nEnter midpoint height:"))
(setq yoff
 (getdist spnt
 "\nEnter midpoint offset:"))
(setq numtimes
 (getint
 "\nEnter number of lines:"))
(setq linefact
 (getreal
 "\nEnter keep line factor:"))
(setq numcycles
 (getreal
 "\nNumber of curve cycles:"))
(setq sectrad1
 (getdist spnt
 "\nEnter frame section radius:"))
(setq sectrad2
 (getdist spnt
 "\nEnter member section radius:"))
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; list of frame points
(setq flist (list ))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1 epnt2)
 (setq pnt2 (list
   (nth 0 epnt2)
   (nth 1 epnt2)
   (+ (nth 2 epnt2) newzedge)))
 (setq pnt3 (list
   (nth 0 xpnt)
   (+ (nth 1 xpnt) newyoff)
   (+ (nth 2 xpnt) newzmid)))
 ; midpt
 (setq xpt (+ (nth 0 pnt2)
  (/ (- (nth 0 pnt3)
  (nth 0 pnt2)) 2)))
 (setq ypt (+ (nth 1 pnt2)
  (/ (- (nth 1 pnt3)
  (nth 1 pnt2)) 2)))
 (setq zpt (+ (nth 2 pnt2)
  (/ (- (nth 2 pnt3)
  (nth 2 pnt2)) 2)))
 (setq pnt3a
  (list xpt ypt zpt))
 (setq pnt4 (list
   (nth 0 epnt1)
   (nth 1 epnt1)
   (+ (nth 2 epnt1) newzedge)))
 ; midpt
 (setq xpt (+ (nth 0 pnt3)
  (/ (- (nth 0 pnt4)
  (nth 0 pnt3)) 2)))
 (setq ypt (+ (nth 1 pnt3)
  (/ (- (nth 1 pnt4)
  (nth 1 pnt3)) 2)))
 (setq zpt (+ (nth 2 pnt3)
  (/ (- (nth 2 pnt4)
  (nth 2 pnt3)) 2)))
 (setq pnt4a
  (list xpt ypt zpt))
 (setq pnt5 epnt1)
 ; make section
 (command ".3DPOLY" pnt1 pnt2
  pnt3a pnt3 pnt4a pnt4 pnt5 "")
 (setq obj1 (ssadd (entlast)))
 ; add cross section
 (command ".CIRCLE" pnt1 sectrad1)
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; SWEEP frame section
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 ; and path
 (setvar "DELOBJ" 2)
 ; sweep sections
 (command ".SWEEP" obj2 "" obj1)
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; add to point list
 (setq flist
  (append (list (list pnt1 pnt2
   pnt3a pnt3 pnt4a pnt4 pnt5))
   flist))
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
(command ".ZOOM" "e")
; horizontal members
(setq ipt 1)
(repeat 5
 (setq iframe 0)
 ; start 3DPOLY
 (command ".3DPOLY")
 (repeat numtimes
   ; add pt to 3DPOLY
   (setq npt
    (nth ipt (nth iframe flist)))
   (command npt)
   ; next frame
   (setq iframe (+ iframe 1))
```

```
  )
  ; close
  (command "")
  (setq obj1 (ssadd (entlast)))
  ; add cross section
  (command ".CIRCLE" npt sectrad2)
  (command ".ROTATE3D"
   "last" "" "y" npt "-90")
  (setq obj2 (ssadd (entlast)))
  (command ".ZOOM" "e")
  ; SWEEP members across frames
  ; turn off history
  (setvar "SOLIDHIST" 0)
  ; set DELOBJ to delete the sections
  ; and path
  (setvar "DELOBJ" 2)
  ; sweep sections
  (command ".SWEEP" obj2 "" obj1)
  ; turn on history
  (setvar "SOLIDHIST" 1)
  (command ".ZOOM" "e")
  ; next pt
  (setq ipt (+ ipt 1))
 )
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

The member section can be circular, elliptical, a polygon, or a rectangular shape. More complex sections can be created by polylines or regions.

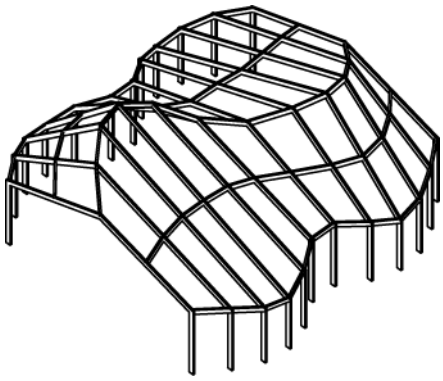

Figure 5.27t: PROG27b, converting frames model to a solid members model, 12 segments, rectangular member

Add function PROG27b, using a copy of PROG237a. Replace the circular section with a rectangular shape.

Replace section radius with section dimensions:

```
(setq sectrad1
  (getdist spnt
  "\nEnter frame section radius:"))
 (setq sectrad2
  (getdist spnt
  "\nEnter member section radius:"))
```

With:

```
(setq sectz
  (getdist spnt
  "\nEnter member height:"))
 (setq sectw
  (getdist spnt
  "\nEnter member width:"))
```

Replace the circle section with a rectangle for the frames:

```
  ; add cross section
  (command ".CIRCLE" pnt1 sectrad1)
```

With:

```
  ; add cross section
  (setq rpt1 (list
   (- (nth 0 pnt1) (/ sectw 2))
   (- (nth 1 pnt1) (/ sectz 2)) 0))
  (setq rpt2 (list
   (+ (nth 0 pnt1) (/ sectw 2))
   (+ (nth 1 pnt1) (/ sectz 2)) 0))
  (command ".RECTANGLE" rpt1 rpt2)
```

Replace the circle section with a rectangle for the horizontal members:

```
  ; add cross section
  (command ".CIRCLE" npt sectrad2)
```

With:

```
  ; add cross section
  (setq rpt1 (list
   (- (nth 0 npt) (/ sectz 2))
   (- (nth 1 npt) (/ sectw 2))
   (nth 2 npt)))
  (setq rpt2 (list
   (+ (nth 0 npt) (/ sectz 2))
   (+ (nth 1 npt) (/ sectw 2))
   (nth 2 npt)))
  (command ".RECTANGLE" rpt1 rpt2)
```

Sample script file for PROG27b:

```
;-----------------------------------
(prog27b)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; member height
8.0"
; member width
3.0"
;-----------------------------------
```

The frames in this series have all been generated from straight line segments. As demonstrated previously, the frames can be converted to arcs or splines.

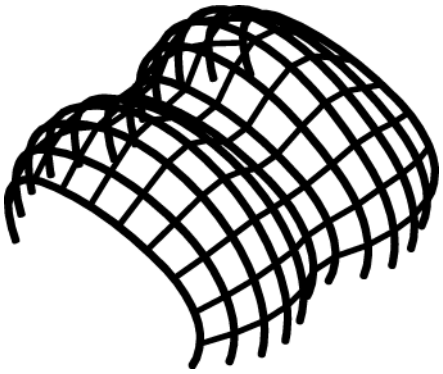

Figure 5.27u: PROG27c, converting frames model to a solid members model, 12 segments, curve fit

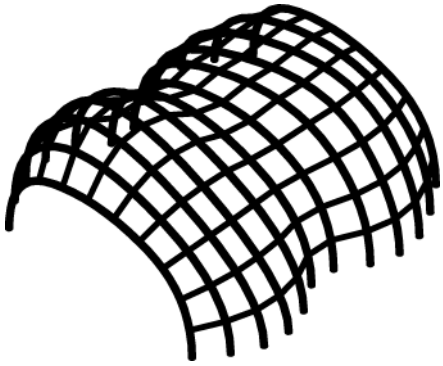

Figure 5.27v: PROG27c, converting frames model to a solid members model, 12 segments, spline fit

Using the methods outlined in function PROG25, convert the frames into a curved section.

Add function PROG27c, using a copy of PROG27 and sections from PROG25.

A PLINE is used since it can be converted to arcs and splines, a 3DPOLY only a spline. As in PROG25, the PLINE will be created in the XY plane first and then rotated into the YZ plane. The PEDIT command will then be used to curve each frame based on the input specified.

The horizontal members considered for the simple frame were based on the vertices and midpoints. In these curved versions, such points are not available, so the DIVIDE command is used to created equal distanced points. These points can be gathered and placed into a list for use in generating the horizontal members.

The points can computed using:

```
; get pts from frame
(command ".DIVIDE"
 obj1 (+ nmembers 1))
; get points
(setq spts (ssget "_p"))
; make pt list
(setq plist (list ))
(setq cnt 0)
; get pt locations
(repeat (sslength spts)
 ; entity name
 (setq objname (ssname spts cnt))
 ; entity record
 (setq objrec (entget objname))
 ; pt corrd
 (setq objpt
  (cdr (assoc 10 objrec)))
 ; add to plist
 (setq plist
  (append plist (list objpt)))
 ; delete pt
 (setq itemp (entdel objname))
 ; inc count
 (setq cnt (+ cnt 1))
)
; add to point list
(setq flist
 (append (list plist) flist))
```

The DIVIDE command generates equal distanced points. The ssget function is used to select these. The sslength function determines the number of points found. The ssname function extracts the entity name from the selection list and the entget function returns an entity record from which the location coordinate of the point, DXF code 10, is found using the assoc function. The entdel function is then used to delete the point.

Review the entire structure of function PROG27c, generating the frames, member locations, and then the members themselves.

Sample script file for PROG27c:

```
;-----------------------------------
(prog27c)
; lower-left
0',0'
; upper-right
@40',20'
; edge height
10'
; center height
20'
; Y offset
8'
; lines
12
; keep line factor
0.75
; cycles
1.0
; frame section radius
4.5"
; member section radius
3.0"
; straight, fit, spline6, spline5
spline5
; number of members
10
;-----------------------------------
```

Completed function PROG27c:

```
;-----------------------------------
(defun prog27c ()
 (graphscr)
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 ; get boundary points and
 ; parameters
 (prompt "\nPROG27c: tubular members")
 (setq spnt
  (getpoint "\nPick start point:"))
 (setq cpnt
  (getcorner spnt
  "\nPick end point:"))
 (setq zedge
  (getdist spnt
  "\nEnter edge height:"))
 (setq zmid
  (getdist spnt
  "\nEnter midpoint height:"))
 (setq yoff
  (getdist spnt
  "\nEnter midpoint offset:"))
 (setq numtimes
  (getint
  "\nEnter number of lines:"))
 (setq linefact
  (getreal
  "\nEnter keep line factor:"))
 (setq numcycles
  (getreal
  "\nNumber of curve cycles:"))
 (setq sectrad1
  (getdist spnt
  "\nEnter frame section radius:"))
 (setq sectrad2
  (getdist spnt
  "\nEnter member section radius:"))
 ; straight, fit, spline6, spline5
 (setq ftype
  (strcase (getstring
  "\nEnter frame type:")))
 (setq nmembers
  (getint
  "\nEnter number of members:"))
```

```
; compute line length,
; width of boundary
(setq linelen
 (- (nth 1 cpnt) (nth 1 spnt)))
; compute increment across
; boundary
(setq xinc
 (/ (- (nth 0 cpnt)
 (nth 0 spnt)) (- numtimes 1)))
; computer ang inc
(setq anginc
 (/ (* 360.0 numcycles)
 (- numtimes 1)))
; set first point and
; first x coord
(setq xpnt spnt)
(setq xpt (nth 0 spnt))
; start ang
(setq ang 0)
; list of frame points
(setq flist (list ))
; loop to repeat lines
(repeat numtimes
 ; compute line length
 (setq linepart1
  (* linelen linefact))
 (setq linepart2
  (* (* linelen (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newlinelen
  (+ linepart1 linepart2))
 ; compute endpoint
 (setq epnt1
  (polar xpnt (dtr 90) newlinelen))
 (setq epnt2
  (polar xpnt (dtr 270) newlinelen))
 ; compute edge height
 (setq linepart1 (* zedge linefact))
 (setq linepart2
  (* (* zedge (- 1.0 linefact))
  (abs (cos (dtr ang)))))
 (setq newzedge
  (+ linepart1 linepart2))
 ; compute midpoint height
 (setq linepart1 (* zmid linefact))
 (setq linepart2
  (* (* zmid (- 1.0 linefact))
  (abs (sin (dtr ang)))))
 (setq newzmid
  (+ linepart1 linepart2))
 ; compute midpoint offset
 (setq newyoff
  (* yoff (sin (dtr ang))))
 ; compute frame points
 (setq pnt1
  (list (nth 0 epnt2)
   (nth 1 epnt2) 0.0))
 (setq pnt2
  (list (+ (nth 0 epnt2) newzedge)
  (nth 1 epnt2) 0.0))
 (setq pnt3
  (list (+ (nth 0 xpnt) newzmid)
  (+ (nth 1 xpnt) newyoff) 0.0))
 (setq pnt4
  (list (+ (nth 0 epnt1) newzedge)
  (nth 1 epnt1) 0.0))
 (setq pnt5
  (list (nth 0 epnt1)
   (nth 1 epnt1) 0.0))
 ; make section
 (command ".PLINE" pnt1 pnt2
  pnt3 pnt4 pnt5 "")
 (setq obj1 (ssadd (entlast)))
 ; convert polyline to a curve
 ; or spline
 (if (/= ftype "STRAIGHT") (progn
  (if (= ftype "FIT") (progn
   (command ".PEDIT" obj1 "f" "")))
  (if (= ftype "SPLINE6") (progn
   (setvar "SPLINETYPE" 6)
   (command ".PEDIT" obj1 "s" "")))
  (if (= ftype "SPLINE5") (progn
   (setvar "SPLINETYPE" 5)
   (command ".PEDIT" obj1 "s" "")))
  (setq obj1 (ssadd (entlast)))
 ))
 ; rotate to YZ
 (command ".ROTATE3D"
  "last" "" "y" pnt1 "-90")
 ; get pts from frame
 (command ".DIVIDE"
  obj1 (+ nmembers 1))
 ; get points
 (setq spts (ssget "_p"))
 ; make pt list
 (setq plist (list ))
 (setq cnt 0)
 ; get pt locations
 (repeat (sslength spts)
  ; entity name
  (setq objname (ssname spts cnt))
  ; entity record
  (setq objrec (entget objname))
  ; pt corrd
  (setq objpt
   (cdr (assoc 10 objrec)))
  ; add to plist
  (setq plist
   (append plist (list objpt)))
  ; delete pt
  (setq itemp (entdel objname))
  ; inc count
  (setq cnt (+ cnt 1))
 )
 ; add to point list
 (setq flist
  (append (list plist) flist))
 ; add cross sections
 (command ".CIRCLE" pnt1 sectrad1)
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; SWEEP frame section
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 ; and path
 (setvar "DELOBJ" 2)
 ; sweep sections
 (command ".SWEEP" obj2 "" obj1)
 ; turn on history
 (setvar "SOLIDHIST" 1)
 ; inc x coord
 (setq xpt (+ xpt xinc))
 (setq xpnt
  (list xpt (nth 1 xpnt)
  (nth 2 xpnt)))
 ; inc ang
 (setq ang (+ ang anginc))
)
(command ".ZOOM" "e")
; horizontal members
(setq ipt 0)
(repeat nmembers
 (setq iframe 0)
 ; start 3DPOLY
 (command ".3DPOLY")
 (repeat numtimes
   ; add pt to 3DPOLY
   (setq npt
    (nth ipt (nth iframe flist)))
   (command npt)
   ; next frame
   (setq iframe (+ iframe 1))
 )
 ; close
 (command "")
 (setq obj1 (ssadd (entlast)))
 ; add cross section
 (command ".CIRCLE" npt sectrad2)
 (command ".ROTATE3D"
  "last" "" "y" npt "-90")
 (setq obj2 (ssadd (entlast)))
 (command ".ZOOM" "e")
 ; SWEEP members across frames
 ; turn off history
 (setvar "SOLIDHIST" 0)
 ; set DELOBJ to delete the sections
 ; and path
```

```
  (setvar "DELOBJ" 2)
  ; sweep sections
  (command ".SWEEP" obj2 "" obj1)
  ; turn on history
  (setvar "SOLIDHIST" 1)
  (command ".ZOOM" "e")
  ; next pt
  (setq ipt (+ ipt 1))
 )
 ; set extrude height
 (command ".ELEV" 0.0 0.0)
 (princ)
)
;-----------------------------------
```

As discussed in function PROG25, the conversion of a PLINE or a 3DPOLY consisting of a series of straight line segments can be controlled by the insertion of additional control points.

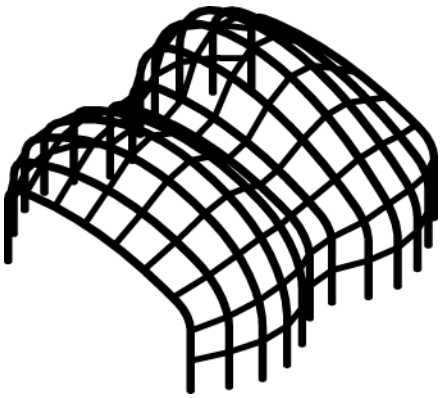

Figure 5.27v: PROG27d, converting frames model to a solid members model, 12 segments, curve fit, two additional points

Add function PROG27d, using a copy of PROG27c, add two additional points which are directly above the support points pnt1 and pnt5. This change also addresses how the bottom of the frame legs meets the ground. The greater the curve, the less the bottom of the frames aligns with the ground.

Add two new points:

```
(setq pnt1a
 (list (+ (nth 0 epnt2) 0.01)
  (nth 1 epnt2) 0.0))
```

And:

```
(setq pnt5a
 (list (+ (nth 0 epnt1) 0.01)
  (nth 1 epnt1) 0.0))
```

The PLINE command is modified to include these new points:

```
; make section
(command ".PLINE" pnt1 pnt1a pnt2
 pnt3 pnt4 pnt5a pnt5 "")
```

The script file for PROG27d is the same as for PROG27c.