

```

; FISH functions to shape tunnels of circular shape with flat floor

; Arrange Pointer Arrays for All Bricks (1-12)

; This function retrieves a pointer from array index IXPOS, IYPOS, and IZPOS
;   when PARRAY points to the beginning of the array.
;   PNT_RETURN is set to the value
def retrieve_pointer
    ioffset = ((pxsize+1)*(pysize+1)*izpos) + ((pxsize+1)*iypos) + ixpos;
    pnt_return = mem(parray+ioffset)
end

; This function sets the pointer stored in array PARRAY at index
;   IXPOS, IYPOS, IZPOS using the value in PNT_SET
def set_pointer
    ioffset = ((pxsize+1)*(pysize+1)*izpos) + ((pxsize+1)*iypos) + ixpos;
    mem(parray+ioffset) = pnt_set
end

; This function allocates memory for the array block and returns
;   a pointer to it in PARRAY. Size is specified by PXSIZE, PYSIZE,
;   and PZSIZE
def get_parray
    isize = (pxsize+1) * (pysize+1) * (pzsize+1)
    parray = get_mem(isize)
end

; This function creates the array using size info (PXSIZE, PYSIZE,
;   and PZSIZE) and fills the pointers to grid-points using the
;   ranges XMIN, XMAX YMIN, YMAX and ZMIN, ZMAX along the the ratios
;   XRAT, YRAT, and ZRAT. This is done to speed up finding gridpoints
;   in a brick even if their positions have changed
def create_and_fill_pointers
    get_parray
    loop ixpos (0,pxsize)
        loop iypos (0,pysize)
            loop izpos (0,pzsize)
                if xrat # 1.0 then
                    fxpos = ((xrat^ixpos)-1.0) / ((xrat^pxsize)-1.0)
                else
                    fxpos = float(ixpos) / float(pxsize)
                endif
                fxpos = (fxpos*(xmax-xmin)) + xmin
                if yrat # 1.0 then

```

```

        fypos = ((yrat^iypos)-1.0) / ((yrat^pysize)-1.0)
    else
        fypos = float(iypos) / float(pysize)
    endif
    fypos = (fypos*(ymax-ymin)) + ymin
    if zrat # 1.0 then
        fzpos = ((zrat^izpos)-1.0) / ((zrat^pzsize)-1.0)
    else
        fzpos = float(izpos) / float(pzsize)
    endif
    fzpos = (fzpos*(zmax-zmin)) + zmin
    pnt_set = gp_near(fxpos, fypos, fzpos)
    set_pointer
end_loop
end_loop
end_loop
end

```

; Creates the index array for all 12 bricks

```

def create_all_bricks
    ii = out('Indexing Brick 1')
    pxsize = zone_x_a      ; Brick 1
    pysize = zone_x_b
    pzsize = zone_y_tun
    xmin = x0brick
    xmax = xlbrick
    ymin = y0brick
    ymax = ylbrick
    zmin = zlbrick
    zmax = z2brick
    xrat = 1.0
    yrat = 1.0
    zrat = 1.0
    create_and_fill_pointers
    parray01 = parray
    ii = out('Indexing Brick 2')
    pxsize = zone_x_a      ; Brick 2
    pysize = zone_z_a
    pzsize = zone_y_tun
    xmin = x0brick
    xmax = xlbrick
    ymin = ylbrick
    ymax = y2brick
    zmin = zlbrick

```

```

zmax = z2brick
xrat = 1.0
yrat = zone_rat_a
zrat = 1.0
create_and_fill_pointers
parray02 = parray
ii = out('Indexing Brick 3')
pxsize = zone_z_b      ; Brick 3
pysize = zone_x_b
pzsize = zone_y_tun
xmin = xlbrick
xmax = x2brick
ymin = y0brick
ymax = ylbrick
zmin = zlbrick
zmax = z2brick
xrat = zone_rat_b
yrat = 1.0
zrat = 1.0
create_and_fill_pointers
parray03 = parray
ii = out('Indexing Brick 4')
pxsize = zone_z_b      ; Brick 4
pysize = zone_z_a
pzsize = zone_y_tun
xmin = xlbrick
xmax = x2brick
ymin = ylbrick
ymax = y2brick
zmin = zlbrick
zmax = z2brick
xrat = zone_rat_b
yrat = zone_rat_a
zrat = 1.0
create_and_fill_pointers
parray04 = parray
ii = out('Indexing Brick 5')
pxsize = zone_x_a      ; Brick 5
pysize = zone_x_b
pzsize = zone_ff_a
xmin = x0brick
xmax = xlbrick
ymin = y0brick
ymax = ylbrick

```

```

zmin = z2brick
zmax = z3brick
xrat = 1.0
yrat = 1.0
zrat = zone_rat_z
create_and_fill_pointers
parray05 = parray
ii = out('Indexing Brick 6')
pxsize = zone_x_a      ; Brick 6
pysize = zone_z_a
pzsize = zone_ff_a
xmin = x0brick
xmax = x1brick
ymin = y1brick
ymax = y2brick
zmin = z2brick
zmax = z3brick
xrat = 1.0
yrat = zone_rat_a
zrat = zone_rat_z
create_and_fill_pointers
parray06 = parray
ii = out('Indexing Brick 7')
pxsize = zone_z_b      ; Brick 7
pysize = zone_x_b
pzsize = zone_ff_a
xmin = x1brick
xmax = x2brick
ymin = y0brick
ymax = y1brick
zmin = z2brick
zmax = z3brick
xrat = zone_rat_b
yrat = 1.0
zrat = zone_rat_z
create_and_fill_pointers
parray07 = parray
ii = out('Indexing Brick 8')
pxsize = zone_z_b      ; Brick 8
pysize = zone_z_a
pzsize = zone_ff_a
xmin = x1brick
xmax = x2brick
ymin = y1brick

```

```

ymax = y2brick
zmin = z2brick
zmax = z3brick
xrat = zone_rat_b
yrat = zone_rat_a
zrat = zone_rat_z
create_and_fill_pointers
parray08 = parray
ii = out('Indexing Brick 9')
pxsize = zone_x_a      ; Brick 9
pysize = zone_x_b
pzsize = zone_ff_b
xmin = x0brick
xmax = x1brick
ymin = y0brick
ymax = y1brick
zmin = z0brick
zmax = z1brick
xrat = 1.0
yrat = 1.0
zrat = zone_rat_z_inv
create_and_fill_pointers
parray09 = parray
ii = out('Indexing Brick 10')
pxsize = zone_x_a      ; Brick 10
pysize = zone_z_a
pzsize = zone_ff_b
xmin = x0brick
xmax = x1brick
ymin = y1brick
ymax = y2brick
zmin = z0brick
zmax = z1brick
xrat = 1.0
yrat = zone_rat_a
zrat = zone_rat_z_inv
create_and_fill_pointers
parray10 = parray
ii = out('Indexing Brick 11')
pxsize = zone_z_b      ; Brick 11
pysize = zone_x_b
pzsize = zone_ff_b
xmin = x1brick
xmax = x2brick

```

```

ymin = y0brick
ymax = y1brick
zmin = z0brick
zmax = z1brick
xrat = zone_rat_b
yrat = 1.0
zrat = zone_rat_z_inv
create_and_fill_pointers
parray11 = parray
ii = out('Indexing Brick 12')
pxsize = zone_z_b      ; Brick 12
pysize = zone_z_a
pzsize = zone_ff_b
xmin = x1brick
xmax = x2brick
ymin = y1brick
ymax = y2brick
zmin = z0brick
zmax = z1brick
xrat = zone_rat_b
yrat = zone_rat_a
zrat = zone_rat_z_inv
create_and_fill_pointers
parray12 = parray
end

```

```

create_all_bricks

```

```

; Finds the extreme X,Y, and Z positions on all six sides of an index
; in a brick

```

```

def find_extremes
  ixsave = xpos
  iysave = ypos
  izsave = zpos
  xpos = 0
  retrieve_pointer
  xmin = xpos(pnt_return)
  xpos = pxsize
  retrieve_pointer
  xmax = xpos(pnt_return)
  xpos = ixsave
  ypos = 0
  retrieve_pointer
  ymin = ypos(pnt_return)

```

```

iypos = pysize
retrieve_pointer
ymax = ypos(pnt_return)
iypos = iysave
izpos = 0
retrieve_pointer
zmin = zpos(pnt_return)
izpos = pysize
retrieve_pointer
zmax = zpos(pnt_return)
izpos = izsave
end

```

```

; Shapes the geometry of brick 3 to match the surface of
; Tunnel B
def shape_tunnel_b ; Fix Surface of Brick 3
    pxsize = zone_z_b
    pysize = zone_x_b
    pzsize = zone_y_tun
    parray = parray03
    xbase = sqrt((zradtunb^2.0)-(zcentunb^2.0))
    ; Move Bottom Row on Outside
    iypos = pxsize
    izpos = 0
    loop iypos (1,pysize)
        retrieve_pointer
        ypos(pnt_return) = (float(iypos)/float(pysize)) * xbase
    end_loop
    ; Move Left Side
    iypos = 0
    loop izpos (1,pzsize)
        retrieve_pointer
        zpos(pnt_return) = (float(izpos)/float(pzsize))*(zradtunb+zcentunb)
    end_loop
    ; Move Top Row
    izpos = pzsize
    loop iypos (1,pysize)
        retrieve_pointer
        theta = (float(iypos)/float(pysize))*(pi/4.0)
        theta = (pi/2.0) - theta
        zpos(pnt_return) = (sin(theta) * zradtunb) + zcentunb
        ypos(pnt_return) = cos(theta) * zradtunb
    end_loop

```

```

; Move Right Side
iypos = pysize
theta_beg = atan2(xbase, zcentunb)
theta_end = pi * 0.75
loop izpos (1, pzsize-1)
    theta = ((float(izpos)/float(pzsize))*(theta_end-theta_beg)) + theta_beg
    retrieve_pointer
    zpos(pnt_return) = (-cos(theta) * zradtunb) + zcentunb
    ypos(pnt_return) = sin(theta) * zradtunb
end_loop
; Move Inside of Face
loop iypos (1, pysize-1)
    loop izpos (1, pzsize-1)
        find_extremes
        retrieve_pointer
        ypos(pnt_return) = ((float(iypos)/float(pysize))*(ymax-ymin)) + ymin
        zpos(pnt_return) = ((float(izpos)/float(pzsize))*(zmax-zmin)) + zmin
    end_loop
end_loop
; Move X Points at Left End
loop iypos (0, pysize)
    loop izpos (0, pzsize)
        ixpos = pxsize
        retrieve_pointer
        fypos = ypos(pnt_return)
        fzpos = zpos(pnt_return)
        fxpos = sqrt((zradtuna^2.0) - ((fzpos-zcentuna)*(fzpos-zcentuna)))
        ixpos = 0
        retrieve_pointer
        xpos(pnt_return) = fxpos
        ypos(pnt_return) = fypos
        zpos(pnt_return) = fzpos
    end_loop
end_loop
; Move All X Locations
xmax = x2brick
loop iypos (0, pysize)
    loop izpos (0, pzsize)
        ixpos = pxsize
        retrieve_pointer
        fypos = ypos(pnt_return)
        fzpos = zpos(pnt_return)
        ixpos = 0
        retrieve_pointer

```



```

    xmin = xpos(pnt_return)
    loop ixpos (1,pxsize-1)
        retrieve_pointer
        if zone_rat_b # 1.0 then
            fxpos = ((zone_rat_b^ixpos)-1.0) / ((zone_rat_b^pxsize)-1.0)
        else
            fxpos = float(ixpos) / float(pxsize)
        endif
        xpos(pnt_return) = (fxpos*(xmax-xmin))+xmin
        ypos(pnt_return) = fypos
        zpos(pnt_return) = fzpos
    end_loop
end_loop
end_loop
end

shape_tunnel_b

; Shapes the geometry of brick 1 to match the tunnel
; intersection. Assumes brick 3 is already shaped
def shape_tunnel_int
    parray = parray01
    pxsize = zone_x_a
    pysize = zone_x_b
    pzsize = zone_y_tun
    ; Top Face
    ixpos = pxsize
    iypos = pysize
    izpos = pzsize
    retrieve_pointer
    ymax = ypos(pnt_return)
    loop iypos (0,pysize)
        ixpos = pxsize
        retrieve_pointer
        theta_end = (pi/2.0) - atan2(zpos(pnt_return)-zcentuna, xpos(pnt_return))
        fypos = (float(iypos)/float(pysize))*ymax
        loop ixpos (0,pxsize-1)
            theta = (float(ixpos)/float(pxsize))*theta_end
            retrieve_pointer
            xpos(pnt_return) = sin(theta) * zradtuna
            ypos(pnt_return) = fypos
            zpos(pnt_return) = (cos(theta) * zradtuna) + zcentuna
        end_loop
    end_loop
end_loop

```

```

; Front Face
iypos = 0
loop izpos (0,pzsize-1)
    ixpos = pxsize
    retrieve_pointer
    xmax = xpos(pnt_return)
    loop ixpos (0,pxsize-1)
        izsave = izpos
        izpos = pzsize
        retrieve_pointer
        zmax = zpos(pnt_return)
        izpos = izsave
        retrieve_pointer
        xpos(pnt_return) = (float(ixpos)/float(pxsize))*xmax
        zpos(pnt_return) = (float(izpos)/float(pzsize))*zmax
    end_loop
end_loop
; Bottom Face
ixpos = pxsize
iypos = pysize
izpos = 0
retrieve_pointer
xmax = xpos(pnt_return)
ymax = ypos(pnt_return)
loop ixpos (0,pxsize-1)
    loop iypos (1,pysize)
        retrieve_pointer
        xpos(pnt_return) = (float(ixpos)/float(pxsize))*xmax
        ypos(pnt_return) = (float(iypos)/float(pysize))*ymax
    end_loop
end_loop
; Left Face
ixpos = 0
iypos = pysize
izpos = pzsize
retrieve_pointer
ymax = ypos(pnt_return)
zmax = zpos(pnt_return)
loop iypos (1,pysize)
    loop izpos (1,pzsize-1)
        retrieve_pointer
        ypos(pnt_return) = (float(iypos)/float(pysize))*ymax
        zpos(pnt_return) = (float(izpos)/float(pzsize))*zmax
    end_loop

```

```

end_loop
; Back Face
iypos = pysize
ixpos = 0
izpos = 0
retrieve_pointer
ymin = ypos(pnt_return)
loop izpos (1,pzsize-1)
    zmin = (float(izpos)/float(pzsize))*z2brick
    ixpos = pxsize
    retrieve_pointer
    zmax = zpos(pnt_return)
    ymax = ypos(pnt_return)
    xmax = xpos(pnt_return)
    loop ixpos (0,pxsize-1)
        retrieve_pointer
        xpos(pnt_return) = (float(ixpos)/float(pxsize))*xmax
        ypos(pnt_return) = ((float(ixpos)/float(pxsize))*(ymax-ymin))+ymin
        zpos(pnt_return) = ((float(ixpos)/float(pxsize))*(zmax-zmin))+zmin
    end_loop
end_loop
; Interior of Brick
loop ixpos (1,pxsize-1)
    loop iypos (1,pysize-1)
        loop izpos (1,pzsize-1)
            find_extremes
            retrieve_pointer
            xpos(pnt_return) = ((float(ixpos)/float(pxsize))*(xmax-xmin))+xmin
            ypos(pnt_return) = ((float(iypos)/float(pysize))*(ymax-ymin))+ymin
            zpos(pnt_return) = ((float(izpos)/float(pzsize))*(zmax-zmin))+zmin
        end_loop
    end_loop
end_loop
end

```

shape_tunnel_int

```

; Shapes the geometry of brick 2 to match tunnel A.
; Assumes bricks 1 and 3 are already shaped
def shape_tunnel_a
    parray = parray02
    pxsize = zone_x_a
    pysize = zone_z_a

```

```

pysize = zone_y_tun
ymax = y2brick
loop xpos (0,pxsize)
  loop izpos (0,pysize)
    iypos = 0
    retrieve_pointer
    fxpos = xpos(pnt_return)
    ymin = ypos(pnt_return)
    fzpos = zpos(pnt_return)
    loop iypos (1,pysize)
      if zone_rat_a # 1.0 then
        fypos = ((zone_rat_a^iypos)-1.0)/((zone_rat_a^pysize)-1.0)
      else
        fypos = float(iypos) / float(pysize)
      endif
      retrieve_pointer
      xpos(pnt_return) = fxpos
      ypos(pnt_return) = (fypos*(ymax-ymin))+ymin
      zpos(pnt_return) = fzpos
    end_loop
  end_loop
end_loop
end

```

shape_tunnel_a

; Shapes the geometry of brick 4 to match bricks 1-3

```

def shape_brick_4
  parray = parray04
  pxsize = zone_z_b
  pysize = zone_z_a
  pszsize = zone_y_tun
  loop izpos (0,pszsize)
    ixpos = 0
    iypos = 0
    retrieve_pointer
    fzpos = zpos(pnt_return)
    loop ixpos (1,pxsize)
      iypos = 0
      retrieve_pointer
      fxpos = xpos(pnt_return)
      loop iypos (1,pysize)
        ixsave = ixpos
        ixpos = 0

```

```

        retrieve_pointer
        fypos = ypos(pnt_return)
        ixpos = ixsave
        retrieve_pointer
        xpos(pnt_return) = fxpos
        ypos(pnt_return) = fypos
        zpos(pnt_return) = fzpos
    end_loop
end_loop
end_loop
end

```

shape_brick_4

; General function to shape the geometry of a brick assuming
; that the zmin and zmax gridpoints are in their final positions

```

def shape_ff_brick
    loop ixpos (0,pxsize)
        loop iypos (0,pysize)
            izpos = 0
            retrieve_pointer
            xmin = xpos(pnt_return)
            ymin = ypos(pnt_return)
            zmin = zpos(pnt_return)
            izpos = pzsize
            retrieve_pointer
            xmax = xpos(pnt_return)
            ymax = ypos(pnt_return)
            zmax = zpos(pnt_return)
            loop izpos (1,pzsize-1)
                if zrat # 1.0 then
                    fzpos = ((zrat^izpos)-1.0) / ((zrat^pzsize)-1.0)
                else
                    fzpos = float(izpos) / float(pzsize)
                endif
                retrieve_pointer
                xpos(pnt_return) = (fzpos*(xmax-xmin)) + xmin
                ypos(pnt_return) = (fzpos*(ymax-ymin)) + ymin
                zpos(pnt_return) = (fzpos*(zmax-zmin)) + zmin
            end_loop
        end_loop
    end_loop
end

```

; Shapes the geometry of all 8 top and bottom bricks

def shape_all_ff_bricks

 ; Brick 5

 parray = parray05

 pxsize = zone_x_a

 pysize = zone_x_b

 pzsize = zone_ff_a

 zrat = zone_rat_z

 shape_ff_brick

 ; Brick 6

 parray = parray06

 pxsize = zone_x_a

 pysize = zone_z_a

 shape_ff_brick

 ; Brick 7

 parray = parray07

 pxsize = zone_z_b

 pysize = zone_x_b

 shape_ff_brick

 ; Brick 8

 parray = parray08

 pxsize = zone_z_b

 pysize = zone_z_a

 shape_ff_brick

 ; Brick 9

 parray = parray09

 pxsize = zone_x_a

 pysize = zone_x_b

 pzsize = zone_ff_b

 zrat = zone_rat_z_inv

 shape_ff_brick

 ;Brick 10

 parray = parray10

 pxsize = zone_x_a

 pysize = zone_z_a

 shape_ff_brick

 ; Brick 11

 parray = parray11

 pxsize = zone_z_b

 pysize = zone_x_b

 shape_ff_brick

 ; Brick 12

 parray = parray12

```
    pxsize = zone_z_b  
    pysize = zone_z_a  
    shape_ff_brick  
end
```

```
shape_all_ff_bricks
```