```
u0=.v0 cross n0
t0=.u0,.v0,.n0
r=.-vp +/ . * t0
t=.(t0,.0),r,1
(4 {.!.1"1 y) +/ . * t
)
```

10.3 Experiment: Painter's Algorithm and Surface Plotting

We now plot a surface using the painter's algorithm for ordering the polygons. We assume the viewing coordinates are the same as in the previous section and that the scripts *raster.ijs* and *dwin.ijs* have been loaded.

```
   f1=: +&sin"0

   f2=: sin@(+&.*:) f.                  function implemented in Section 5.3

   $x=: y=: _14+28*(i.%<:)61            sample at 61 points in each direction
61

   $xyz=: x ([,],f2)"0/ y               array of x-y-z triples
61 61 3
```

We will use a function `quad` to rearrange 2 by 2 arrays of *x*-*y*-*z* triples into lists of four vertices. We will construct the list of all the quadrilaterals for the surface by applying `quad` to the 2 by 2 tessellations of the 61 by 61 array. In order to use painter's algorithm, we use the "*z*" values of the projection which were previously unused. Thus, `VPZ` is very similar to `VPP`, but it results in the negative of the distance from the viewpoint toward the center of interest.

```
   quad=: 0 1 3 2&{@(,/)

   $polys=: ,/,/2 2 quad ;._3 xyz       get the 3600 polygons
3600 4 3

   $pp=: VPP polys                      get 3600 projected polygons
3600 4 2                                uses default viewing parameters

   avg=: +/ % #                         average

   $paz=: avg"1 VPZ polys               average projected z coordinate
3600                                    for polygons

   (<./,:>./) ,/ pp                     range of projected polygons
_19.4145 _5.60556
 19.4145  5.97225

   _20 _10 20 10 dwin 'view plane'      open graphics window

    0 128 255 dpoly pp /: paz           draw the surface
```

The projected polygons are reordered by `pp /: paz` into the order determined by `paz` and they are colored a light shade of blue.

Next, the color of each polygon on the surface is a hue chosen according to the application of `cile`, from Section 5.3, to the *z*-value of the original data.

```
   $z=: ,/}:}:"1{:"1 xyz
3600

   $colorz=: (256 cile z){Hue 5r6*(i.%<:) 256
3600 3

   dclear ''                            clear the graphics window
```

```
   (colorz /: paz) dpoly pp /: paz
```

Figure 10.3.1 shows the surface.

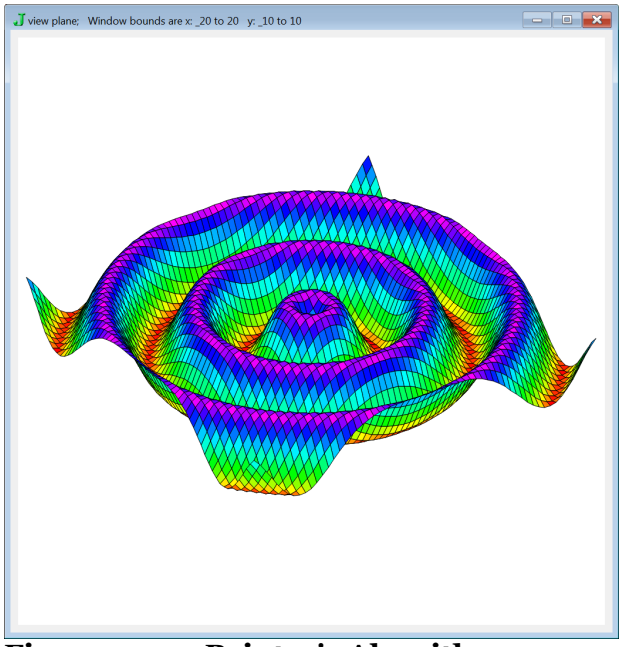

Figure 10.3.1 Painter's Algorithm

You should now create a plot of the function $\sin(x)+\sin(y)+\sin\sqrt{x^2+y^2}$ for $-14 \le x, y \le 14$.

10.4 Perspective Projections

Perspective projections use distance from the view plane to modify the apparent size of polygons. In particular, objects will appear smaller when they are farther away. Figure 10.4.1 shows the general idea of a perspective projection. The scaling ratios are determined by the ratio of the distance *d* from the viewpoint to the view plane to the distance from the viewpoint to the object. That is, the vertical length *y* of the object is a distance *d*-*z* from the viewpoint and hence it needs to be scaled to *Y*, where $Y/d = y/(d-z)$. Thus we need $Y = yd/(d-z)$ and, likewise, $X = xd/(d-z)$; see Figure 10.4.2. The implementation requires only a few changes from the orthogonal projection developed in Section 10.2. We need all three projected coordinates, and we need to apply the above adjustments to get the perspective *X* and *Y* coordinates. We choose a viewpoint further away from the scene, a view distance, and scale our window to fit a different scale.

```
   vp=: 20 30 10                          viewpoint
   ci=: 0 0 0                             center of interest
   up=: 0 0 1                             the up direction
   d=: 10                                 distance to view plane
```

We assume `f2`, `quad`, and `avg` from Section 10.3 are defined. Then we can do the following.

```
   $x=: y=: _14+28*(i.%<:)61              sample at 61 points
61

   $xyz=: x ([,],f2)"0/ y                 array of x-y-z triples
61 61 3

   $polys=: ,/,/2 2 quad ;._3 xyz         polygons
3600 4 3

                                          function to give perspective pairs
   vpper=: (d&*@}: % d&-@{:)@(3&{.)"1@((ci;vp;up)&VPPXYZ)

   $pp=: vpper polys                      perspective projected polygons
3600 4 2

   (<./,:>./) ,/ pp                       range of the polygons
_3.81118 _1.82156
 4.46746 0.906156

   $paz=: avg"1 (ci;vp;up)&VPZ polys      projected z
3600

   $z=: ,/}:}:"1{:"1 xyz
3600
```