*/*The following code was written in C programming language. It serves to define the geometry for the radiative model of the current CERES telescope and can be modified to model potential next-generation instruments. It calls C^{++} library functions written by TRG doctoral student, Félix Nevárez. This code generates output which can be used to determine the Optical Point Spread Function of the CERES instrument. Each surface within the telescope is modeled, and is referenced by a combination of letters and numbers. This labeling corresponds to the labeled Figure 5.1 (a) of the CERES telescope found in this thesis. Note that all dimensions are in inches, scaled by a factor of ten (10).*/*

```
#include "raytracer.h"
#include "math.h"
#define PI 3.14159265359

main()
{
/*s is the scaling factor to scale all telescope dimensions*/
double s=10;
/*sh is the shift added by the insertion of shims to achieve best focus*/
double sh=.0110 ;
/*apw is the width of a precision aperture*/
double apw=0.0296;
/*d is the distance between precision apertures*/
double d=0.01;
/*theta is direction 1 of incoming rays*/
double theta;
/*phi is direction 2 of incoming rays*/
double phi;

/* Begin defining the CERES telescope geometry*/

/*CY1(Baffle)*/
makeACylinder(0.561*s);
setProperties(0.9,0.1);
setClipPlane (0.0,0.0,(0.0+o),0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.771+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R1(Baffle)*/
makeARing(0.0,0.0,(0.0+o)*s,0.0,0.0,1.0,0.4665*s,0.5610*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*R2(Baffle)*/
makeARing(0.0,0.0,(0.39+o)*s ,0.0,0.0,1.0,0.4570*s,0.5610*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*R3(Baffle)*/
makeARing(0.0,0.0,(0.770+o)*s,0.0,0.0,1.0,0.4475*s,0.5610*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*R4(Baffle)*/
```

```
makeARing(0.0,0.0,(1.149+o)*s ,0.0,0.0,1.0,0.4385*s,0.5610*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*R5(Baffle)*/
makeARing(0.0,0.0,(1.529+o)*s ,0.0,0.0,1.0,0.4290*s,0.5610*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*C1(Baffle)*/
makeACone(0.3239*s,0.561*s);
setProperties(0.99,0.10);
setOrigin(0.0,0.0,(2.0949+o)*s);
setClipPlane(0.0,0.0,(1.771+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.8220+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*CY2(Baffle)*/
makeACylinder(0.472*s);
setProperties(0.99,0.1);
setClipPlane(0.0,0.0,(1.8220+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.8540+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R6(Baffle)*/
makeARing(0.0,0.0,(1.854+o)*s,0.0,0.0,1.0,0.3995*s,0.472*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*C2(CapReflector)*/
makeACone(0.952*s,0.424*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(0.9567+o)*s);
setClipPlane(0.0,0.0,(1.854+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.909+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C3(CapReflector)*/
makeACone(0.1904*s,0.424*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.0994+o)*s);
setClipPlane(0.0,0.0,(1.909+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.920+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C4(CapReflector)*/
makeACone(0.952*s,0.424*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(1.0227+o)*s);
```

```
setClipPlane(0.0,0.0,(1.92+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.975+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C5(CapReflector)*/
makeACone(0.1904*s,0.424*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.1654+o)*s);
setClipPlane(0.0,0.0,(1.975+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(1.986+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C6(CapReflector)*/
makeACone(0.952*s,0.424*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(1.0887+o)*s);
setClipPlane(0.0,0.0,(1.986+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.041+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C7(CapReflector)*/
makeACone(0.1904*s,0.424*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.2314+o)*s);
setClipPlane(0.0,0.0,(2.041+o)*s,0.0,0.0,-1.0);
setClipPlane(0.0,0.0,(2.052+o)*s,0.0,0.0,1.0);
noOutput();
nextOBJ();

/*C8(CapReflector)*/
makeACone(0.952*s,0.424*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(1.1547+o)*s);
setClipPlane(0.0,0.0,(2.052+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.107+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C9(CapReflector)*/
makeACone(0.1904*s,0.424*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.2974+o)*s);
setClipPlane(0.0,0.0,(2.107+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.118+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C10(CapReflector)*/
makeACone(0.952*s,0.424*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(1.2207+o)*s);
setClipPlane(0.0,0.0,(2.118+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.173+o)*s,0.0,0.0,-1.0);
```

```
noOutput();
nextOBJ();

/*C11(CapReflector)*/
makeACone(0.1904*s,0.424*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.363+o)*s);
setClipPlane(0.0,0.0,(2.173+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.184+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C12(CapReflector)*/
makeACone(0.924*s,0.411*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(1.287+o)*s);
setClipPlane(0.0,0.0,(2.184+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.211+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R7(CapReflector)*/
makeARing(0.0,0.0,(2.211+o)*s,0.0,0.0,1.0,0.411*s,0.4745*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*CY3(CapReflector)*/
makeACylinder(0.4745*s);
setProperties(0.99,0.1);
setClipPlane(0.0,0.0,(2.211+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.221+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R8(CapReflector)*/
makeARing(0.0,0.0,(2.221+o)*s,0.0,0.0,1.0,0.4745*s,0.522*s);
setProperties(0.01,0.9);
noOutput();
nextOBJ();

/*CY4(CapReflector)*/
makeACylinder(0.522*s);
setProperties(0.01,0.9);
setClipPlane (0.0,0.0,(2.221+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R9(CapReflector)*/
makeARing(0.0,0.0,(2.263+o)*s ,0.0,0.0,1.0,0.4725*s,0.5220*s);
setProperties(0.01,0.9);
noOutput();
nextOBJ();

/*CY5(TelescopeHousing)*/
```

```
makeACylinder(0.4725*s);
setProperties(0.99,0.1);
setClipPlane (0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.386+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*D1(Spider)*/
makeADisk(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0,0.185*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*leg1(Spider)*/
makeAPlane(0.0,0.0,(2.263+0.113+o)*s,0.0,0.0,1.0);
setProperties(0.99,0.1);
setClipPlane(0.03*s,-0.4725*s,(2.263+0.113+o)*s,0.0,1.0,0.0);
setClipPlane(0.03*s,-0.4725*s,(2.263+0.113+o)*s,-1.0,0.0,0.0);
setClipPlane(-0.03*s,0.0,(2.263+0.113+o)*s,1.0,0.0,0.0);
setClipPlane(0*s,-0.185*s,(2.263+0.113+o)*s,0.0,-1.0,0.0);
noOutput();
nextOBJ();

/*leg2(Spider)*/
makeAPlane(0.0,0.0,(2.263+0.113+o)*s,0.0,0.0,1.0);
setProperties(0.99,0.1);
setClipPlane(0.160*s,0.0925*s,(2.263+0.113+o)*s,0.866,0.5,0.0);
setClipPlane(0.459*s,0.23045*s,(2.263+0.113+o)*s,-0.5,0.866,0.0);
setClipPlane(0.459*s,0.23045*s,(2.263+0.113+o)*s,-0.866,-0.5,0.0);
setClipPlane(0.429*s,0.28225*s,(2.263+0.113+o)*s,0.5,-0.866,0.0);
noOutput();
nextOBJ();

/*leg3(Spider)*/
makeAPlane(0.0,0.0,(2.263+0.113+o)*s,0.0,0.0,1.0);
setProperties(0.99,0.1);
setClipPlane(-0.160*s,0.0925*s,(2.263+0.113+o)*s,-0.866,0.5,0.0);
setClipPlane(-0.459*s,0.23045*s,(2.263+0.113+o)*s,0.5,0.866,0.0);
setClipPlane(-0.459*s,0.23045*s,(2.263+0.113+o)*s,0.866,-0.5,0.0);
setClipPlane(-0.429*s,0.28225*s,(2.263+o+0.113)*s,-0.5,-0.866,0.0);
noOutput();
nextOBJ();

/*side1toleg1(Spider)*/
makeAPlane(0.03*s,0.0,(2.263+0.113+o)*s,-0.9659,0.0,0.2588);
setProperties(0.001,0.9);
setClipPlane(0.03*s,-0.4725*s,(2.263+0.113+o)*s,0.0,1.0,0.0);
setClipPlane(0.0,0.0,(2.263+0.113+o)*s,0.0,0.0,-1.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0*s,-0.185*s,(2.263+0.113+o)*s,0.0,-1.0,0.0);
noOutput();
nextOBJ();

/*side2toleg1(Spider)*/
makeAPlane(-0.03*s,0.0,(2.263+0.113+o)*s,-0.9659,0.0,-0.2588);
setProperties(0.001,0.9);
```

```
setClipPlane(0.03*s,-0.4725*s,(2.263+0.113+o)*s,0.0,1.0,0.0);
setClipPlane(0.03*s,0.0,(2.263+0.113+o)*s,0.0,0.0,-1.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0*s,-0.185*s,(2.263+0.113+o)*s,0.0,-1.0,0.0);
noOutput();
nextOBJ();


/*side1toleg2(Spider)*/
makeAPlane(0.429*s,0.28225*s,(2.263+0.113+o)*s,0.48295,-0.866,0.2588);
setProperties(0.001,0.9);
setClipPlane(0.160*s,0.0925*s,(2.263+0.113+o)*s,0.866,0.5,0.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.4092*s,0.23625*s,(2.263+o+0.113)*s,-0.866,-0.5,0.0);
setClipPlane(0.0,0.0,(2.263+o+0.113)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*side2toleg2(Spider)*/
makeAPlane(0.459*s,0.23045*s,(2.263+0.113+o)*s,-0.48295,0.866,0.2588);
setProperties(0.001,0.9);
setClipPlane(0.160*s,0.0925*s,(2.263+0.113+o)*s,0.866,0.5,0.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.4092*s,0.23625*s,(2.263+0.113+o)*s,-0.866,-0.5,0.0);
setClipPlane(0.0,0.0,(2.263+0.113+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();


/*side1toleg3(Spider)*/
makeAPlane(-0.459*s,0.23045*s,(2.263+0.113+o)*s,0.48295,0.866,0.2588);
setProperties(0.001,0.9);
setClipPlane(-0.160*s,0.0925*s,(2.263+0.113+o)*s,-0.866,0.5,0.0);
setClipPlane(-0.4092*s,0.23625*s,(2.263+o+0.113)*s,0.866,-0.5,0.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.263+o+0.113)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*side2toleg3(Spider)*/
makeAPlane(-0.429*s,0.28225*s,(2.263+0.113+o)*s,-0.48295,-0.866,0.2588);
setProperties(0.001,0.9);
setClipPlane(-0.160*s,0.0925*s,(2.263+0.113+o)*s,-0.866,0.5,0.0);
setClipPlane(-0.4092*s,0.23625*s,(2.263+0.113+o)*s,0.866,-0.5,0.0);
setClipPlane(0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.263+o+0.113)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*CY6(Spider)*/
makeACylinder(0.185*s);
setProperties(0.99,0.1);
setClipPlane (0.0,0.0,(2.263+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.506+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();
```

```
/*CY7(Spider)*/
makeACylinder(0.16*s);
setProperties(0.99,0.1);
setClipPlane (0.0,0.0,(2.448+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.4762+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C13(Spider)*/
makeACone(0.2205*s,0.185*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.2855+o)*s);
setClipPlane(0.0,0.0,(2.4762+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.506+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*M2(SecondaryMirror)*/
makeASphere(1.318*s);
setProperties(0.001,1.0);
setOrigin(0.0,0.0,(1.14+o)*s);
setClipPlane(0.0,0.0,(2.448+o)*s,0.0,0.0,1.0);
noOutput();
nextOBJ();

/*C14(TelescopeHousing)*/
makeACone(0.2483*s,0.4725*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.6343+o)*s);
setClipPlane(0.0,0.0,(2.386+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.448+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R10(TelescopeHousing)*/
makeARing(0.0,0.0,(2.448+o)*s ,0.0,0.0,1.0,0.3545*s,0.4295*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*CY8(TelescopeHousing)*/
makeACylinder(0.4295*s);
setProperties(0.99,0.1);
setClipPlane (0.0,0.0,(2.448+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.498+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*R11(TelescopeHousing)*/
makeARing(0.0,0.0,(2.498+o)*s ,0.0,0.0,1.0,0.365*s,0.4295*s);
setProperties(0.99,0.1);
noOutput();
nextOBJ();

/*C15(TelescopeHousing)*/
```

```
makeACone(0.2807*s,0.486*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.2873+o)*s);
setClipPlane(0.0,0.0,(2.498+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.568+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C16(TelescopeHousing)*/
makeACone(0.2807*s,0.486*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.8487+o)*s);
setClipPlane(0.0,0.0,(2.568+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.638+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C17(TelescopeHousing)*/
makeACone(0.2807*s,0.486*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.4273+o)*s);
setClipPlane(0.0,0.0,(2.638+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.708+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C18(TelescopeHousing)*/
makeACone(0.2807*s,0.486*s);
setProperties(0.99,0.1);
setOrigin(0.0,0.0,(2.9887+o)*s);
setClipPlane(0.0,0.0,(2.708+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.778+o)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*CY9(TelescopeHousing)*/
makeACylinder(0.365*s);
setProperties(0.99,0.1);
setClipPlane (0.0,0.0,(2.778+o)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.831+o+sh)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*M1(PrimaryMirror)*/
makeASphere(1.446*s);
setProperties(0.001,1.0);
setOrigin(0.0,0.0,(1.432+o+sh)*s);
setClipPlane(0.0,0.0,(2.831+o+sh)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.867+o+sh)*s,0.0,0.0,-1.0);
noOutput();
nextOBJ();

/*C19(PrimaryMirrorInsert)*/
makeACone(0.0466*s,0.174*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.820+o+sh)*s);
```

```
setClipPlane(0.0,0.0,(2.857+o+sh)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.867+o+sh)*s,0.0,0.0,-1.0);
noOutput();

/*C20(PrimaryMirrorInsert)*/
makeACone(0.082*s,0.135*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.939+o+sh)*s);
setClipPlane(0.0,0.0,(2.857+o+sh)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.915+o+sh)*s,0.0,0.0,-1.0);
noOutput();

/*C21(PrimaryMirrorInsert)*/
makeACone(0.109*s,0.0509*s);
setProperties(0.001,0.9);
setOrigin(0.0,0.0,(2.831+o+sh)*s);
setClipPlane(0.0,0.0,(2.915+o+sh)*s,0.0,0.0,1.0);
setClipPlane(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0);
noOutput();

/*Precision Aperture*/
/*If number of precision apertures =1*/
/*D2 1precisionaperture*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.3*s);
setProperties(0.001,0.9);
setClipPlane(0.0148*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane(-0.0148*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();


/*If number of precision apertures =2*/
/*D2 2precisionapertures*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.03950*s);
setProperties(0.001,0.9);
setClipPlane((0.0148+(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((((apw+d)/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*s/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*s/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((-((apw+d)/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*s/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*s/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();

/*If number of precision apertures =3*/
/*D2 3precisionapertures*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.3*s);
```

```
setProperties(0.001,0.9);
setClipPlane(0.0148*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane(-0.0148*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148+apw+d)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((apw+d-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-apw-d)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((-apw-d-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();

/*If number of precision apertures =4*/
/*D2 4precisionapertures*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.1*s);
setProperties(0.001,0.9);
setClipPlane((0.0148+(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((((apw+d)/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*s/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*s/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((-((apw+d)/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*s/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*s/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148+3*(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*3*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*3*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane(((3*(apw+d)/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*s*3/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*3*s/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-3*(apw+d)/2)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*3*s/2,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*3*s/2,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((-((apw+d)*3/2)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*s*3/2,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*s*3/2,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
```

```
/*If number of precision apertures =5*/
/*D2 5precisionapertures*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.03950*s);
setProperties(0.001,0.9);
setClipPlane(0.0148*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(0.0,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane(-0.0148*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(0.0,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148+apw+d)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((apw+d-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-apw-d)*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((-apw-d-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148+2*(apw+d))*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane((apw+d)*2*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane((apw+d)*2*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((2*(apw+d)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane((apw+d)*2*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane((apw+d)*2*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();
setClipPlane((0.0148-2*(apw+d))*s,0.0,0.0,-1.0,0.0,0.0);
setClipPlane(-(apw+d)*2*s,0.0296*s,0.0,-1.0,-1.0,0.0);
setClipPlane(-(apw+d)*2*s,0.0296*s,0.0,1.0,-1.0,0.0);
setClipPlane((2*(-apw-d)-0.0148)*s,0.0,0.0,1.0,0.0,0.0);
setClipPlane(-(apw+d)*2*s,-0.0296*s,0.0,1.0,1.0,0.0);
setClipPlane(-(apw+d)*2*s,-0.0296*s,0.0,-1.0,1.0,0.0);
clipAsAHole();

/*Adiskforvisualization of blur circle at precision aperture*/
makeADisk(0.0,0.0,(2.9404+o+sh)*s,0.0,0.0,-1.0,0.3*s);
setProperties(1.0,1.0);
nextOBJ();

/*Loop to vary input angles for determination of OPSF*/
for (phi=88.6; phi<90.0; phi=phi+0.1)
{
for(theta=0.0; theta < 4.5; theta=theta+0.1)
{
setSourceDir(sin(theta*PI/180.0)*sin(phi*PI/180.0),cos(phi*PI/180.0),
cos(theta*PI/180.0)*sin(phi*PI/180.0));
startRayTrace(100000);
}
}
}
```

The following code was written in FORTRAN. It is used to post process the output files generated from running the provided C code (the first eleven (11) pages of Appendix C). This particular code is used for the case in which five detectors, thus five precision apertures are placed at the focal plane of the telescope. The output files generated from this code are in a form that can be directly imported into excel, and plotted to obtain the Optical Point Spread Function at each detector. Note that all dimensions are in inches and are scaled by ten (10).

```
      PROGRAM OPSF
      IMPLICIT NONE
      REAL*8 APW,D,XPOS,YPOS,ZPOS,AP1,AP2,AP3,MAX1,MAX2,MAX3,MAXNUM1,
     $MAXNUM2,MAXNUM3
       INTEGER I,NUM1,NUM2,NUM3,LINES,J,NUMTHETA,NUMPHI,N,NUMTOTAL,COUNT
     $,NUM,L,P,R,O,C
      CHARACTER*100 FICHIER
      CHARACTER*100 SC_LET
      DIMENSION AP1(15,45), AP2(15,45), AP3(15,45),MAX1(15),
     $MAX2(15),MAX3(15)

*     Open files for output (ap1, ap2, and ap3).  Open file numberb.dat which contains the ends
*     of the file names produced by Felix's code.   Numberb.dat contains lines '001.dat', '002.dat',
*     etc. up to the number of output files produced.

      OPEN(15,FILE='numberb.dat',STATUS='OLD')
      OPEN(16,FILE='ap1.dat',STATUS='OLD')
      OPEN(17,FILE='ap2.dat',STATUS='OLD')
      OPEN(18,FILE='ap3.dat',STATUS='OLD')

*     APW is the width of the precision apertures, D is the distance between them.

      APW=0.296D0
      D=0.1D0
      NUM1=0
      NUM2=0
      NUM3=0
      N=1
      COUNT=1
      NUMTHETA=45
      NUMPHI=15
      NUMTOTAL=NUMTHETA*NUMPHI

*     Initialize the value in the arrays containing maximum response for each valueof phi (max1...),
*     initialize the maximum response of each aperture (maxnum1,...) to zero.

      DO 1 C=1,NUMPHI
      MAX1(C)=0
      MAX2(C)=0
      MAX3(C)=0
1     CONTINUE
      MAXNUM1=0
      MAXNUM2=0
      MAXNUM3=0

      DO WHILE (N.LE.NUMPHI)

      DO 10 J=COUNT,NUMTHETA*N
```

```
      NUM1=0
      NUM2=0
      NUM3=0

      READ(15,*)SC_LET

     FICHIER = 'OUTPUT'
    $    //SC_LET
      OPEN(24,FILE=FICHIER,STATUS='OLD')

*     Start a loop in which each output file is opened and read.  Each output file corresponds to a given
*     theta, phi combination (see previous C code).  When reading a given output file, each
*     line contains a set of coordinates for a ray arriving at the focal plane.  If the arriving ray falls within
*     the central aperture, the counter NUM1 is incremented.  Similarly, if it falls in the second
*     aperture, NUM2 is incremented and if it falls within the third aperture, NUM3 is incremented.


      DO 235 I=1,50000

      READ(24,*, END=30)XPOS,YPOS,ZPOS
      IF(XPOS.GT.(-APW/2.0D0).AND.XPOS.LE.0)THEN
      IF(ABS(YPOS).LT.(XPOS+0.296d0))THEN
      NUM1=NUM1+1
      END IF
      END IF

      IF(XPOS.GT.0.AND.XPOS.LT.(APW/2.d0))THEN
      IF(ABS(YPOS).LT.(0.296d0-XPOS))THEN
      NUM1=NUM1+1
      END IF
      END IF

      IF(ABS(XPOS).GT.(APW/2.d0+D).AND.ABS(XPOS)
     &.LE.(APW+D))THEN
      IF(ABS(YPOS).LT.((abs(XPOS)-(APW+D))+0.296d0))THEN
      NUM2=NUM2+1
      END IF
      END IF

      IF(ABS(XPOS).GT.(APW+D).AND.ABS(XPOS).LT.
     &(1.5D0*APW+D))THEN
      IF(ABS(YPOS).LT.(0.296d0-(ABS(XPOS)-
     &(APW+D))))THEN
      NUM2=NUM2+1
      END IF
      END IF

      IF(ABS(XPOS).GT.(1.5d0*APW+2.d0*D).AND.ABS(XPOS)
     &.LE.(2.0d0*(APW+D)))THEN
      IF(ABS(YPOS).LT.(ABS(XPOS)-(2.0d0*(APW+D))+0.296d0))
     &THEN
      NUM3=NUM3+1
      END IF
      END IF

      IF(ABS(XPOS).GT.(2.0d0*(APW+D)).AND.ABS(XPOS).LT.
```

```
     &(2.5d0*APW+2.d0*D))THEN
      IF(ABS(YPOS).LT.(0.296d0-(ABS(XPOS)-(2.0d0*
     &(APW+D)))))THEN
      NUM3=NUM3+1
      END IF
      END IF
      NUM=J-NUMTHETA*(N-1)

235    CONTINUE

30    AP1(N,NUM)=NUM1
      AP2(N,NUM)=NUM2
      AP3(N,NUM)=NUM3

10     CONTINUE

      COUNT=J
      N=N+1
      END DO

*     Determine the maximum response within each of the detectors

      DO 60 R=1,NUMPHI

      DO 70 P=1,NUMTHETA
      IF (AP1(R,P).GT.MAX1(R)) THEN
      MAX1(R)=AP1(R,P)
      END IF
      IF (AP2(R,P).GT.MAX2(R))THEN
      MAX2(R)=AP2(R,P)
      END IF
      IF (AP3(R,P).GT.MAX3(R))THEN
      MAX3(r)=AP3(R,P)
      END IF
70     CONTINUE
60     CONTINUE

      DO 80 O=1,NUMPHI
      IF (MAX1(0).GT.MAXNUM1) THEN
      MAXNUM1=MAX1(O)
      END IF
      IF (MAX2(O).GT.MAXNUM2) THEN
      MAXNUM2=MAX2(O)
      END IF
      IF (MAX3(O).GT.MAXNUM3) THEN
      MAXNUM3=MAX3(O)
      END IF
80     CONTINUE

*     Write the output for the Optical Point Spread Function for each detector, normalized
*     by the maximum response within all detectors (maximum response will be at central
*     detector.

      DO 50 L=1,NUMTHETA

     WRITE(16,200)AP1(15,L)/maxnum1,AP1(14,L)/maxnum1,
```

```
     $AP1(13,L)/maxnum1,AP1(12,L)/maxnum1,AP1(11,L)/maxnum1,
     $AP1(10,L)/maxnum1,AP1(9,L)/maxnum1,AP1(8,L)/maxnum1,
     $AP1(7,L)/maxnum1,AP1(6,L)/maxnum1,AP1(5,L)/maxnum1,
     $AP1(4,L)/maxnum1,AP1(3,L)/maxnum1,AP1(2,L)/maxnum1,
     $AP1(1,L)/maxnum1

      WRITE(17,200)AP2(15,L)/maxnum1,AP2(14,L)/maxnum1,
     $AP2(13,L)/maxnum1,AP2(12,L)/maxnum1,AP2(11,L)/maxnum1,
     $AP2(10,L)/maxnum1,AP2(9,L)/maxnum1,AP2(8,L)/maxnum1,
     $AP2(7,L)/maxnum1,AP2(6,L)/maxnum1,AP2(5,L)/maxnum1,
     $AP2(4,L)/maxnum1,AP2(3,L)/maxnum1,AP2(2,L)/maxnum1,
     $AP2(1,L)/maxnum1

      WRITE(18,200)AP3(15,L)/maxnum1,AP3(14,L)/maxnum1,
     $AP3(13,L)/maxnum1,AP3(12,L)/maxnum1,AP3(11,L)/maxnum1,
     $AP3(10,L)/maxnum1,AP3(9,L)/maxnum1,AP3(8,L)/maxnum1,
     $AP3(7,L)/maxnum1,AP3(6,L)/maxnum1,AP3(5,L)/maxnum1,
     $AP3(4,L)/maxnum1,AP3(3,L)/maxnum1,AP3(2,L)/maxnum1,
     $AP3(1,L)/maxnum1

50    CONTINUE
200   FORMAT(F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,
     $F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,F6.4,1x,
     $F6.4,1x,F6.4,1x)
      END
```