

Thermal emission and imaging is easily accomplished using **FRED**'s Graphical User Interface and its built-in scripting language. **FRED** uses efficient calculations using standard optical engineering algorithms to accomplish thermal imaging and emission although brute force raytracing is also possible. Using techniques derived from radiometry, it is possible to perform thermal imaging, narcissus, stray light, thermal illumination uniformity, and thermal self-emission calculations efficiently and accurately in a small fraction of the time it would take to trace the requisite number of rays using **FRED**.

1. What is thermal emission and thermal imaging?

Thermal imaging is defined as the process of producing a visible two-dimensional image of a scene that is dependent on the differences in thermal or infrared radiation from the scene reaching the aperture of the imaging device. Thermal imaging systems normally subtract the background to enhance the contrast of the variations in the infrared scene. When the background is not uniform, as in the presence of narcissus, a stray signal can be produced. This process is especially important for Defense and Security concerns where items that have different thermal temperatures or emissivities can be found when differentiated from the rest of an imaged scene. Primary applications for this are: detection, classification and tracking of concealed weapons, personnel, vehicles, objects and materials hidden on persons or in baggage, vehicles or shipping containers. Figure 1 is an excellent example of a household item, a teapot, imaged through a camera lens using a thermal detector when simulated in **FRED**.

Thermal emission is energy emitted from the environment or structure around an optical instrument that causes stray light problems. Narcissus is one thermal emission problem where thermal radiation in an infrared system appears as a dark circular area on a displayed image due to radiation reflecting onto the detector.

Usually, these systems work by detecting a small signal superimposed on a large background. At room temperature, the peak of the blackbody emission curve is at about $10\mu\text{m}$. Thus the world “glows” at this wavelength, and small variations in this glow indicate differences in temperature or emissivity. Specifically, when an image of the cooled detector is imaged back on itself, a locally strong absence of background occurs. This typically appears as a dark spot in the center of the image. One might call this “stray dark” instead of stray light.

In infrared radiometers that measure absolute radiance instead of a relative signal, any background radiation is unacceptable. In such an instrument it may be necessary to cool the entire instrument to cryogenic temperatures to eliminate the stray light caused by self-emission.

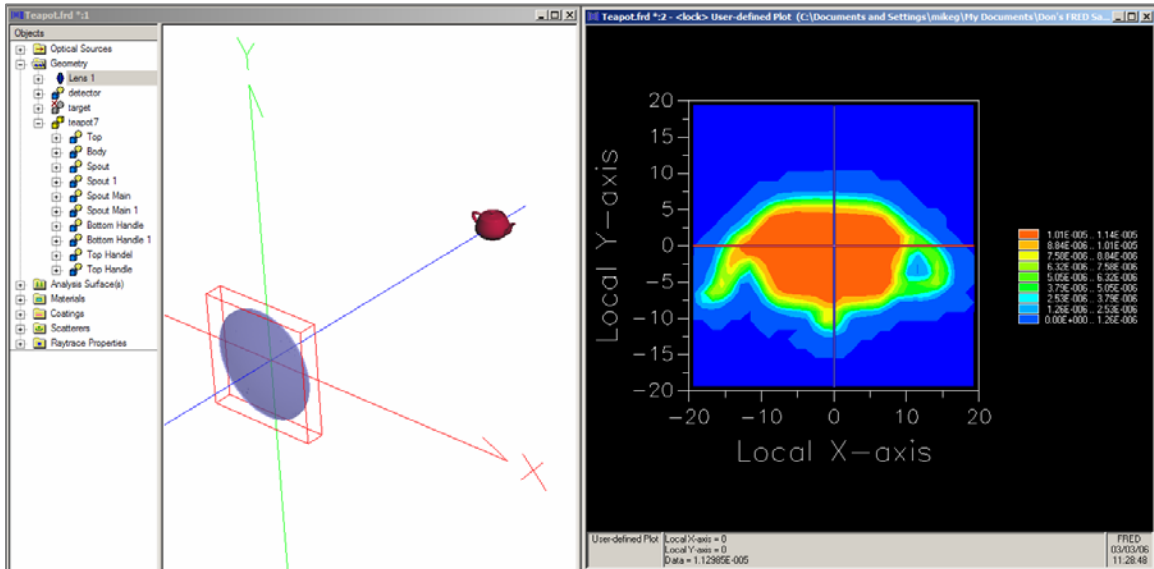


Figure 1 – This figure demonstrates a simple problem with a warm teapot with different emissivities and temperature distribution across the teapot’s surface. The teapot is then imaged through a singlet with the detector positioned behind the singlet. This type of problem, where mechanical structure is emitting to a detector, can be found in many infrared systems leading to thermal problems that have to be either moved or baffled.

More recently, efforts have been aimed at expanding IR sensing to applications including temperature measurement and mapping, forest fire sensing and suppression, surveillance, and multi-spectral earth imaging, etc.

Many of these applications are done over long distances, through the atmosphere, and where absorption of IR energy is a factor in the performance of these systems. Military and space-based applications, generally, can be addressed by detectors whose operating wavelengths fall between 8.0-15 microns where atmospheric absorption is minimized. Other applications fall in the broader waveband of .09-300 microns.

2. How does **FRED** ray trace and visualize thermal emissions and imaging?

To track thermal emission in FRED there are several methods. The first is to create a source and brute force ray trace it through the optical system. The second way is to ray trace backwards from the detector through the system which requires fewer rays. It is extremely important in both methods to be able to visualize the thermal image with both 2D and 3D plots.

There are really two issues here: computation time and accuracy. In a complex system, ray trace times can become excessive particularly if the analyst is trying to access the

effect of incremental changes on the design and wants to do so in “real time”. Reverse raytracing can make the calculation almost interactive. Furthermore since power converges faster than uniformity, the analyst is almost certainly assured of an accurate result, even if only a few rays from each differential area reach the thermal source.

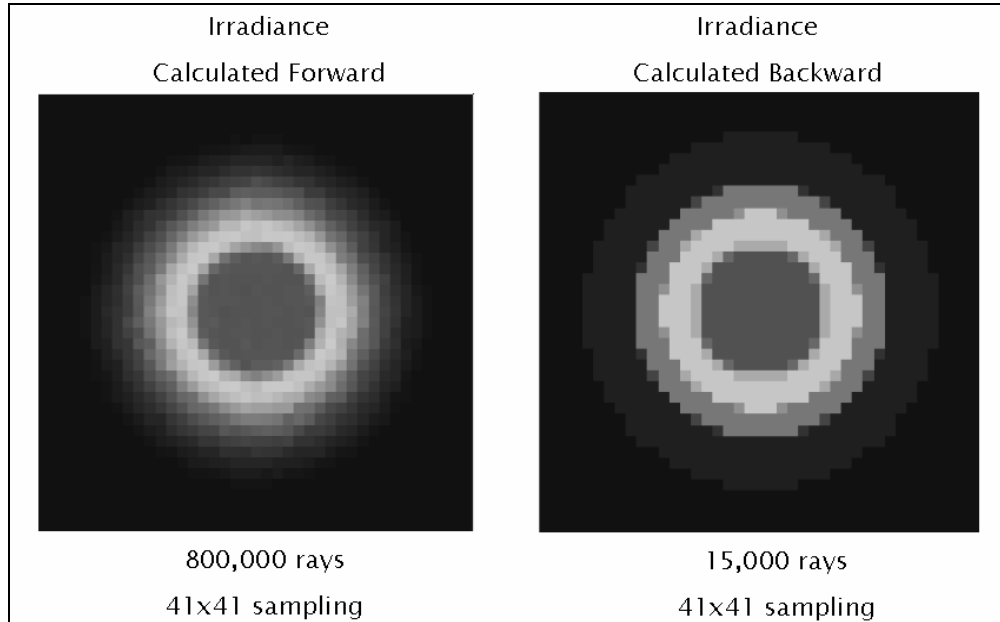


Figure 2 – Comparison of two irradiance calculations: one performed using forward raytracing and the other using backward raytracing. The latter required 53x fewer rays to reach the same level of accuracy.

Thermal self-emission is simple to describe: each optical and mechanical structure radiates energy as a Lambertian emitter as a function of its temperature and emissivity. This emitted energy is simulated by tracing rays; these rays follow the laws of geometrical optics as they transmit and reflect during their propagation through the system. The rays (and hence the thermal energy they represent) are accumulated at the FPA.

Pursuant to this calculation, most software let the user set the temperature and emissivity of objects. From a statistical standpoint, this is entirely the wrong thing to do! In most “real” systems, the FPA subtends a very small solid angle relative to the optical and mechanical components and so when a great many rays are traced, few, if any, reach the FPA (Figure 3). The result is a poor estimate of the thermal self-emission.

There is a more efficient but non-optimal approach. Either directly or indirectly, most software allow the user to specify a preferred radiation direction; these are referred to as “importance sampling” in the literature. Using this technique, the user specifies an importance sampling for each optical and mechanical component. During the ray trace, rays are scattered towards these preferred directions that can be very efficient at conducting rays to the FPA (Figure 4). This tremendously improves the statistics and yields an accurate assessment of the thermal self-emission. However in a complex, “real life” system with many structural components, this is tedious and time consuming to do, especially if the analyst needs to do temperature and emissivity trades.

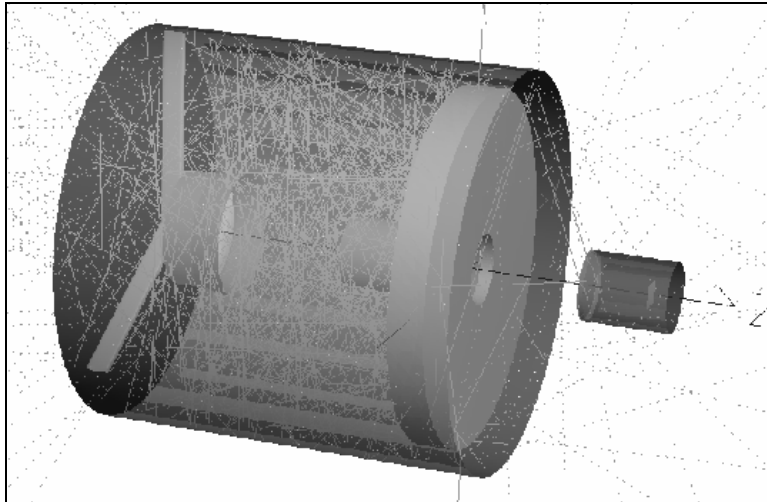


Figure 3 - Brute force raytracing of the thermally emitting main barrel. In this case, none of the emitted rays reached the FPA.

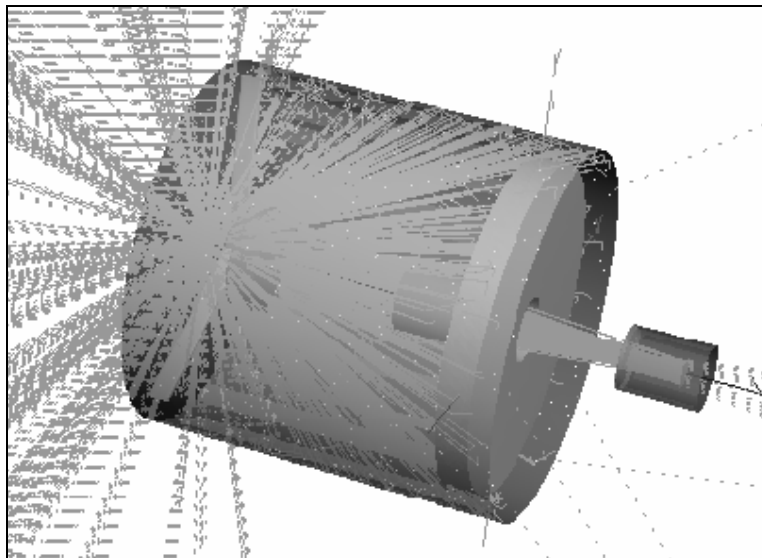


Figure 4 - Emitting towards an importance direction is more efficient but tedious to setup.

The best way to approach this calculation is to employ the mathematics of thermal radiometry. Computing the thermal self-emission (TSE) is basically a summation of the form

$$TSE = \sum_{\text{all objects}} L_{\text{object}} A_{\text{object}} \Omega_{\text{detector}} \quad (1)$$

Since thermal emission is Lambertian, we can replace L with an equivalent expression derived from thermal exitance

$$TSE = \sum_{\text{all objects}} \left[\frac{\varepsilon f \sigma T^4}{\pi} \right] A_{\text{object}} \Omega_{\text{detector}} \quad (2)$$

where ε = emissivity, f = fractional blackbody integral, σ = Stefan-Boltzmann constant, T = temperature (deg K). The problem is how to efficiently calculate A_{object} and Ω_{detector} .

Reversing the roles of the source and collector (using Eq. 3), we can write

$$TSE = \sum_{\text{all objects}} L_{\text{object}} A_{\text{detector}} \Omega_{\text{object}} \quad (3)$$

Note that the area of the detector A_{detector} is a fixed quantity. To efficiently calculate the solid angle of each emitting object, we invoke symmetry in the radiance equation

$$P_{\text{object}} = L_{\text{detector}} A_{\text{detector}} \Omega_{\text{object}} \quad (4)$$

From eq. 4, we note that if we launch rays from the detector with $L = 1 / \pi A_{\text{detector}}$, then the power incident on the object will be numerically equal to its projected solid angle. (Recall that the projected solid angle is equal to Ω/π .) The emitted detector power is therefore given by

$$\Phi = \sin^2 \theta \quad (5)$$

(Actually this is correct only if the detector is radiating into a cone. If we desire to radiate into a rectangular volume, the correct detector power is a factor of $4/\pi$ larger.)

Knowing how to calculate the projected solid angle efficiently is the key to the technique; this is the second “clever trick”. Since the solid angles are constant, they need to be calculated only once.

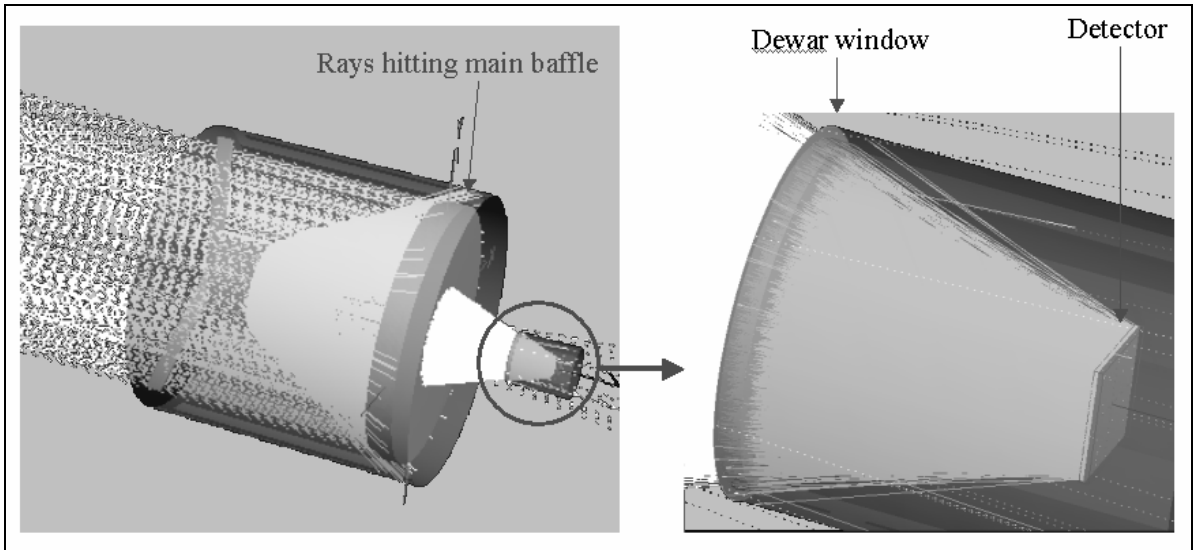


Figure 5 - Tracing rays backwards from the FPA to obtain accurate projected solid angle value. In most cases, the dewar window is the nearest limiting aperture relative to the detector and so we typically launch rays only into its solid angle. The thermal self-emission calculation is contained in eq. 6, which is readily implemented in a spreadsheet (Fig. 6). For completeness, the final equation for thermal self-emission is given by

$$TSE = \sum_{all\ objects} \epsilon f \sigma T^4 A_{detector} \left(\frac{\Omega}{\pi} \right) \quad (6)$$

where the expression (Ω/π) is the projected solid angle calculated by the ray trace.

Surface	Incident Power	Temperature	Emissivity	Contribution
.fastcass.Primary mirror.Reflecting Surface	0.008691	300	0.02	3.99173E-08
.fastcass.Primary mirror.Back Surface	0.070631	300	1	1.62194E-05
.fastcass.Primary mirror.Hole	0.006843	300	1	1.57144E-06
.fastcass.Secondary mirror.Reflecting Surface	0.012657	300	0.02	5.81299E-08
.fastcass.dewar.FPA.detector array	1.17E-05	300	0	0
.fastcass.dewar.dewar window.Surface 1	0.117072	300	0.02	5.37675E-07
.fastcass.dewar.dewar window.Surface 2	0.117142	300	0	0
.fastcass.dewar.dewar window.Edge	7.02E-05	300	1	1.61158E-08
.fastcass.dewar.outer wall.Surf 1	5.85E-05	90	1	1.08775E-10
.main barrel.Surf 3	0.000795	300	1	1.82663E-07
.Primary baffle.Surf 3	0.026659	300	1	6.12191E-06
.Secondary baffle.Surf 3	0.002644	300	1	6.07087E-07
.Secondary baffle.Surf 4	0.000409	300	1	9.40179E-08
.Secondary baffle.Surf 5	0.000175	300	1	4.02934E-08
.Secondary struts.strut a.Surf 6	9.36E-05	300	1	2.14892E-08
.Secondary struts.strut b.Surf 6	9.36E-05	300	1	2.14892E-08
.Secondary struts.strut c.Surf 6	0.000105	300	1	2.4176E-08
			Total	2.55559E-05

Figure 6 - Calculating thermal self-emission using a spreadsheet such as Excel™. The column “Incident Power” is actually projected solid angle. The column “Contribution” implements eq. 6.

The advantages of reverse raytracing for thermal self-emission calculations are numerous and include

1. Solid angle accuracy is determined by the number of rays traced from the detector
2. Temperatures and emissivities can be readily changed in the spreadsheet for “what if” trades
3. There is no tedious setup time; setup time is no longer a function of model complexity
4. A map of the thermal self-emission at the detector can be accomplished by dividing the detector into differential areas and raytracing and summing from each area

Modern optical software is extremely powerful and enabling, giving both the experienced and inexperienced users the means to complete many sophisticated analyses with the click of a button on a toolbar. However there are many occasions where users need to perform calculations not immediately available in a predefined fashion by the software. Employing “clever tricks” is oftentimes the only way to perform these calculations in a reasonable period of time and to the level of accuracy required

3. How does **FRED** create geometry for thermal analysis?

System geometry can be created directly in **FRED** through the simple to use graphical interface, imported from IGES or STEP CAD formats and optical design programs. The program has many options to create surfaces including standard planes, conics, cylinders, ellipsoids, hyperboloids, toroids, polynomial surfaces, Zernike, Nurb, Meshed, revolved curves, extruded curbs, composite curves, splines and user-defined surfaces.

Since **FRED** has a multiple document user interface, components may be cut, copied, and pasted between documents. Entities may be logically arranged into hierarchies of assemblies, subassemblies, elements, etc. that correspond to the physical layout of the system; each can be located relative to any arbitrary coordinate system. Any surface may be trimmed (sliced) by any implicit surface, or by an aperture collection curve, which is defined below.

Any surface can be defined in terms of temperature and emissivity. It is also possible to define each surface as a thermally emitting source to propagate thermal energy and create thermal imaging plots.

4. How does **FRED** keep track of thermal emission paths?

FRED has the capability to do an advanced ray trace that keeps track of all the paths for all rays traced in a system. Figure 7 shows a list of ray paths for the thermal imaging of a teapot by using backward raytracing from detector through the lens and then to the teapot as shown in Figure 1.

This ray history is a complete report of the power for each path, the number of rays that followed that path, how they reached the last entity and how many surfaces they went through (Event Count). It is also possible to take any ray trace path and copy it to a user defined path list (select the path, right mouse click on the path and then select the option to copy this path to the user-defined path list). This path will now show up as an optional path in the advanced ray trace as one of the ray methods to use. It is then possible to do either spot diagrams or irradiance spread functions on only this path.

It is by using this method that it is possible to see how much power each of the ghosting, straight shot, single or multiple scatter paths contribute versus the signal path.

	Total Power	Ray Count	Event Count	Spec Refl Count	Spec Tran Count	Scat Refl Count	Scat Tran Count	Absorb Count	Diffract Count	Spec Ancestry	Scat Ancestry	First Entity	Last Entity	Previous Entity
3	0.036	36	3	0	3	0	0	0	0	0	0	Optical Sources.DiffrActArea	Lens 1.Surface 1	Lens 1.Surface 2
1	0.25448	265	4	0	3	0	0	0	0	0	0	Optical Sources.DiffrActArea	teapot7.Body.B-Spline Surface 4	Lens 1.Surface 1
2	0.000227487	1514	4	0	3	1	0	0	0	0	1	Optical Sources.DiffrActArea	teapot7.Body.B-Spline Surface 4	Lens 1.Surface 1
8	1.079770e-8	6	5	0	3	1	0	1	0	0	1	Optical Sources.DiffrActArea	teapot7.Body.B-Spline Surface 4	teapot7.Body.B-Spline Surface 4
4	0.006	6	4	0	3	0	0	1	0	0	0	Optical Sources.DiffrActArea	teapot7.Bottom Handle 1.B-Spline Surface 16	Lens 1.Surface 1
5	8.584253e-5	654	5	0	3	1	0	1	0	0	1	Optical Sources.DiffrActArea	teapot7.Bottom Handle 1.B-Spline Surface 16	teapot7.Body.B-Spline Surface 4
7	5.486221e-5	304	5	0	3	1	0	1	0	0	1	Optical Sources.DiffrActArea	teapot7.Bottom Handle B-Spline Surface 14	teapot7.Body.B-Spline Surface 4
6	2.062647e-5	111	5	0	3	1	0	1	0	0	1	Optical Sources.DiffrActArea	teapot7.Top Handle B-Spline Surface 18	teapot7.Body.B-Spline Surface 4
0	0.693	693	1	0	1	0	0	0	0	0	0	Optical Sources.DiffrActArea	Optical Sources.DiffrActArea	

Figure 7 – Ray Paths for the Teapot Example shown in Figure 1. Note that there are 3 paths reaching the Teapot Body (.teapot7.Body.B-Spline Surface 4) which are paths 1, 2 and 8 for this system shown as the last entity in the 2nd to last column. Path number 2 is the straight path from the detector through the lens and then propagating to the teapot body. Notice the difference in the power for each of these 2 paths, .25448 for the 1st path and .000227487 for the 2nd path. The 2nd path is the scattered portion of the path; the 1st path is the direct thermal contribution. It is also possible to see the definition of this path; it is shown in Figure 9.

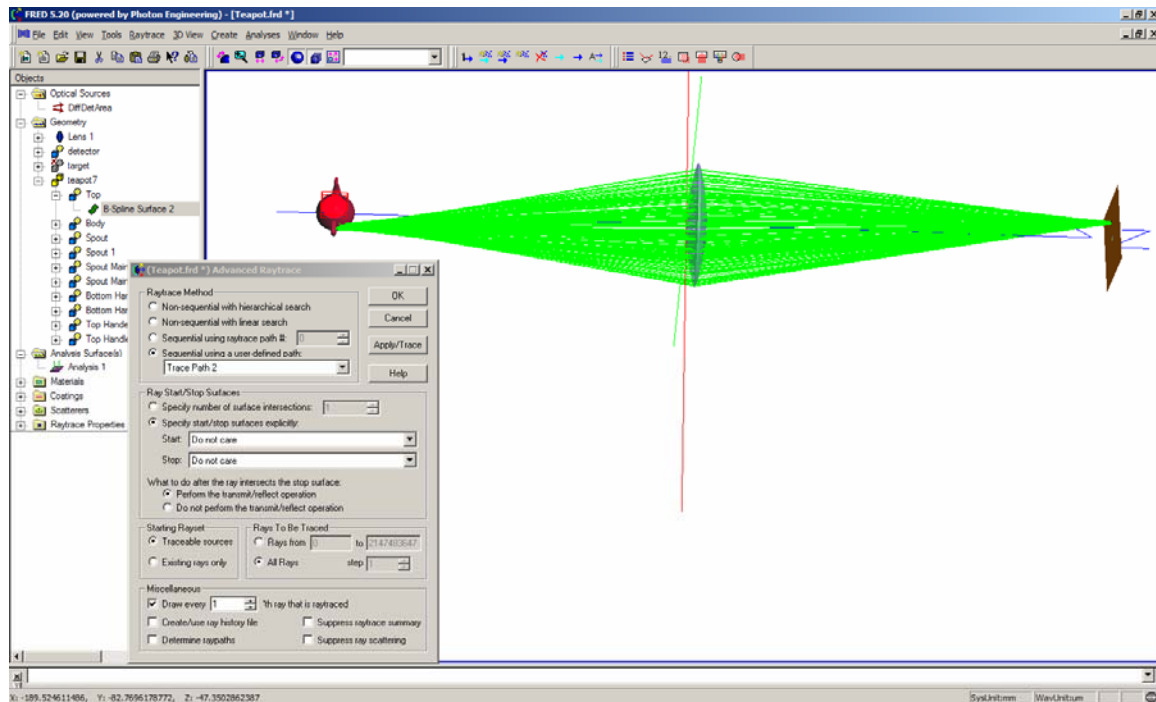


Figure 8– The 2nd path traced through the lens.

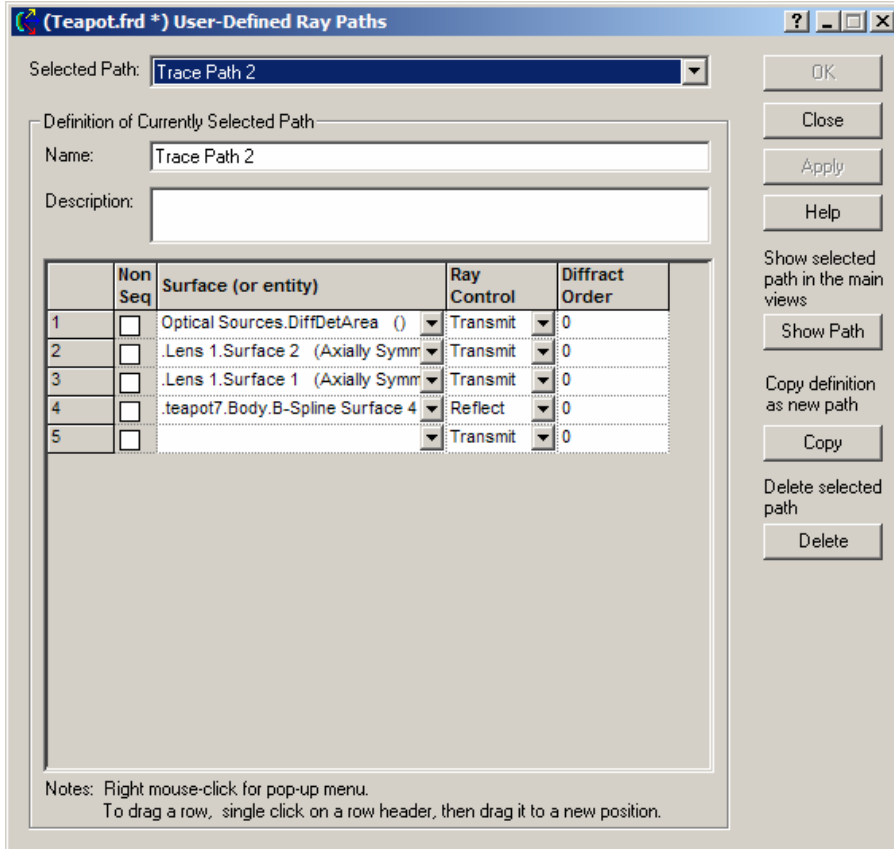


Figure 9 – Ray Path information for the path shown in Figure 8.

5. How does **FRED** show irradiance or thermally imaged plots?

By default, **FRED** shows irradiance or thermally imaged maps using a 3 plot color panel. The top left plot is an isometric pseudo-color plot showing the power per unit area on the analysis surface selected. There is a scale at right which shows power levels for this map. The top right and bottom left panels are cross sections of the top left panel. Click anywhere on the top left map will create a cross section of the 3D plot in both the horizontal and vertical positions; the position and irradiance at this position are reported in the bottom left hand corner of this top left panel. The irradiance map for the teapot system is shown in Figure 10.

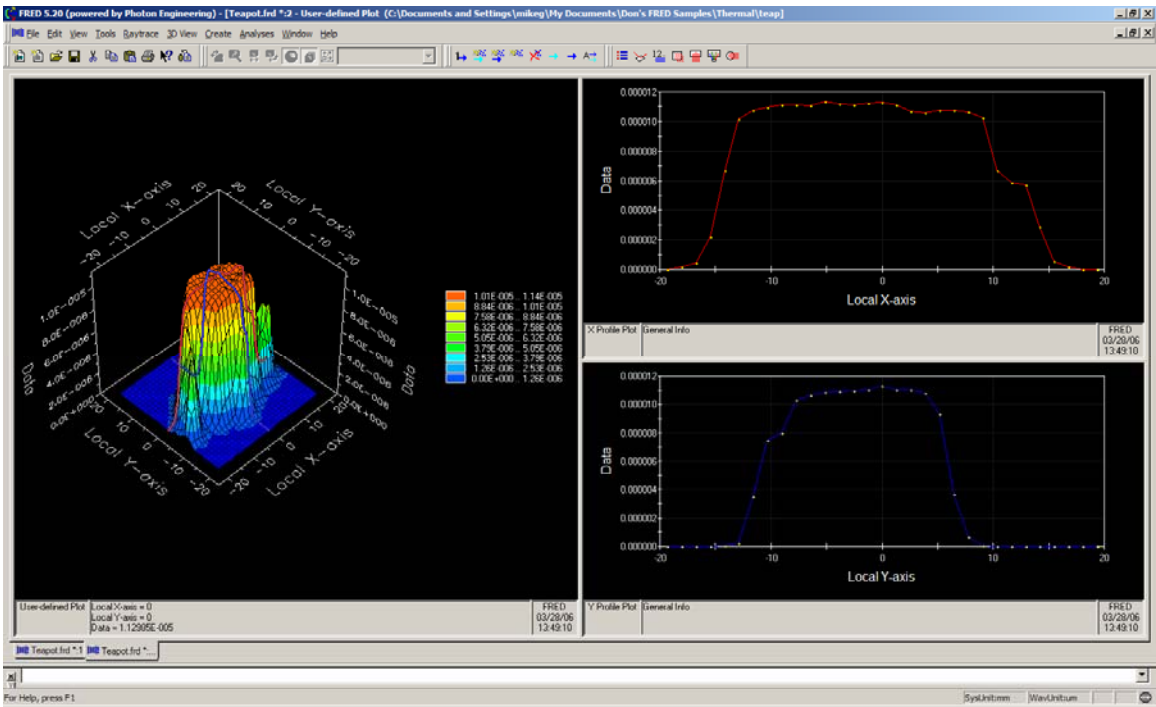


Figure 10 – Irradiance map for the teapot system.

To look at only one panel, double left mouse click on that panel to enlarge the view. The top panel is in perspective mode, to create a 2D image right mouse click to bring up the charting options and select the perspective view to create a 2D view as shown in Figure 11.

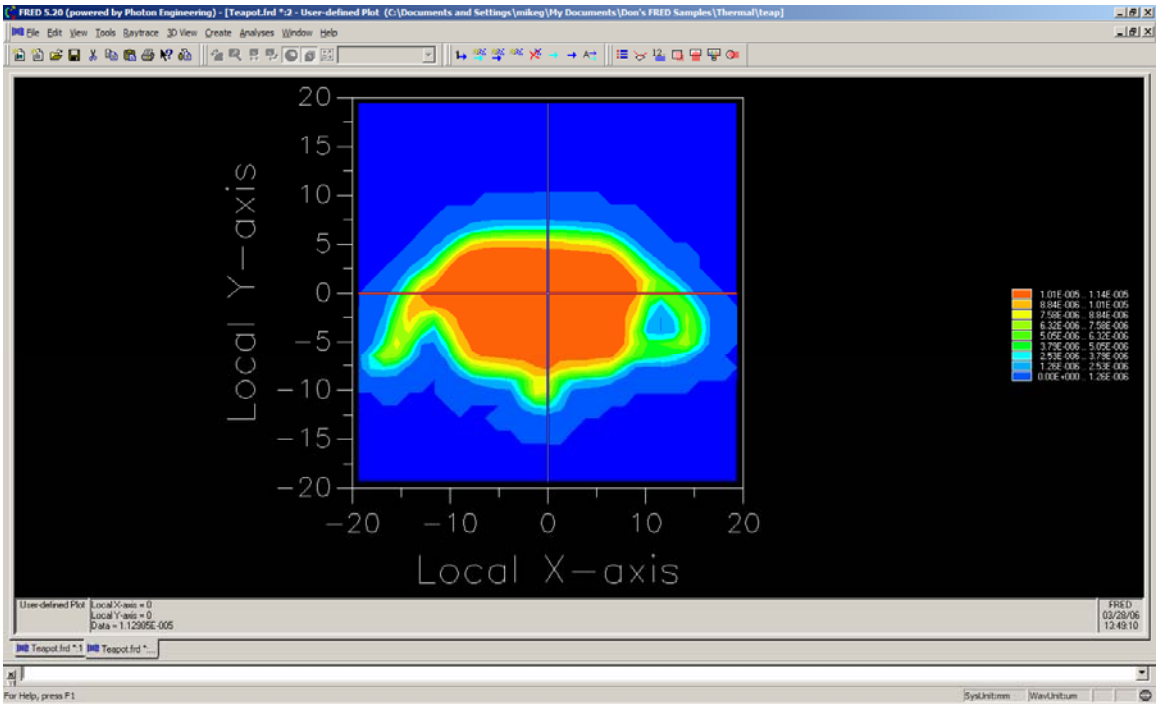


Figure 11 – Thermal image of the teapot in 2D chart mode

6. How does **FRED** define scattering surfaces in thermal applications?

The **Scatterers** folder contains the default and optional user entered scatter models that may be applied to any surface in **FRED**. Each model calculates the appropriate three-dimensional Bidirectional Scatter Distribution Function (BSDF) based on the incident ray angles and orientation of the local surface normal. Alternate definitions of the BSDF are the Bidirectional Reflectance Distribution (BRDF) and the Bidirectional Transmission Distribution Function (BTDF).

FRED comes with three default scatter models: Black Lambertian (4% reflectivity diffuse black), White Lambertian (96% reflectivity diffuse white), and Harvey-Shack (polished surface). Additionally, parametric models for the following types of scatterers are also available in **FRED**: Flat Black Paint (TIS), ABg, Surface Particle (Mie), and Phong. More than one type of scatter model can be applied to a surface. Figure 12 shows the dialogue menu for creating a custom scatter definition, note that **FRED**'s newest scatter definition is a scripted BSDF function where the user defines a scattering equation. Reflected and transmitted scatter components are allowed or halted per the Ray trace Controls currently applied to the surface. Every scatter surface must have at least one scatter direction, which can be set automatically, using the menu bar option, Tools: Determine Scatter Importance Sampling, or manually, from the Scatter tab in the Surface dialog. Every scatter direction is applied to every scatter model assigned to the surface. The dialogue to set the importance sampling for a surface is shown in Figure 13. It is possible to define multiple importance sampling targets by defining targets in a specific or specular direction, towards a specific entity, closed curve, a point in space or an ellipsoidal volume.

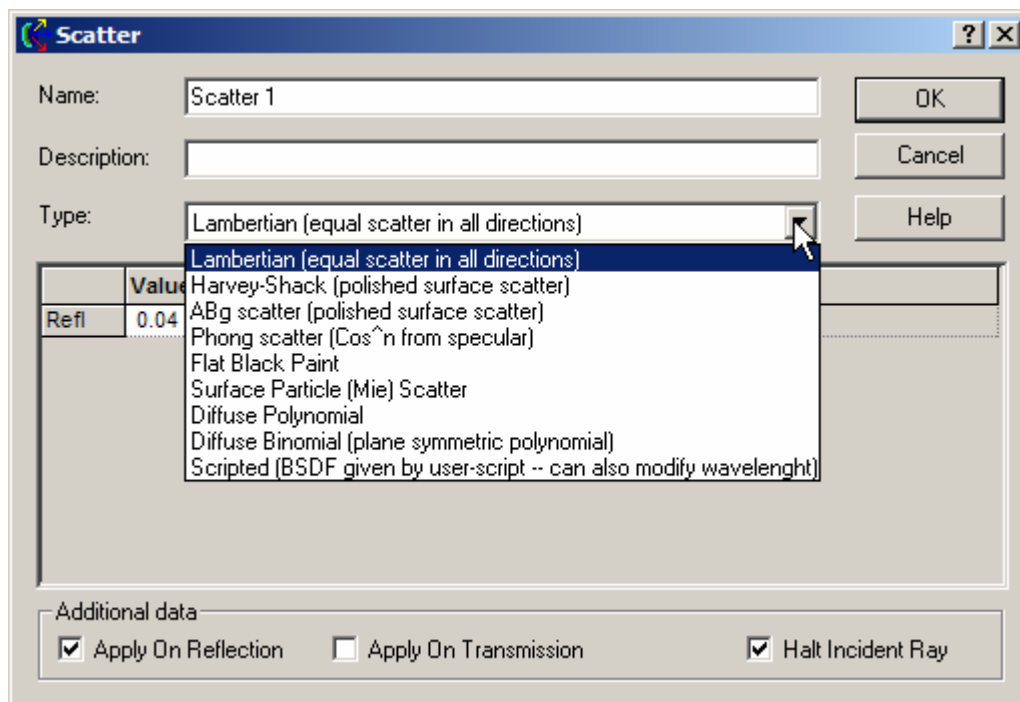


Figure 12 – Scatter dialog showing the number of ways to define scatter

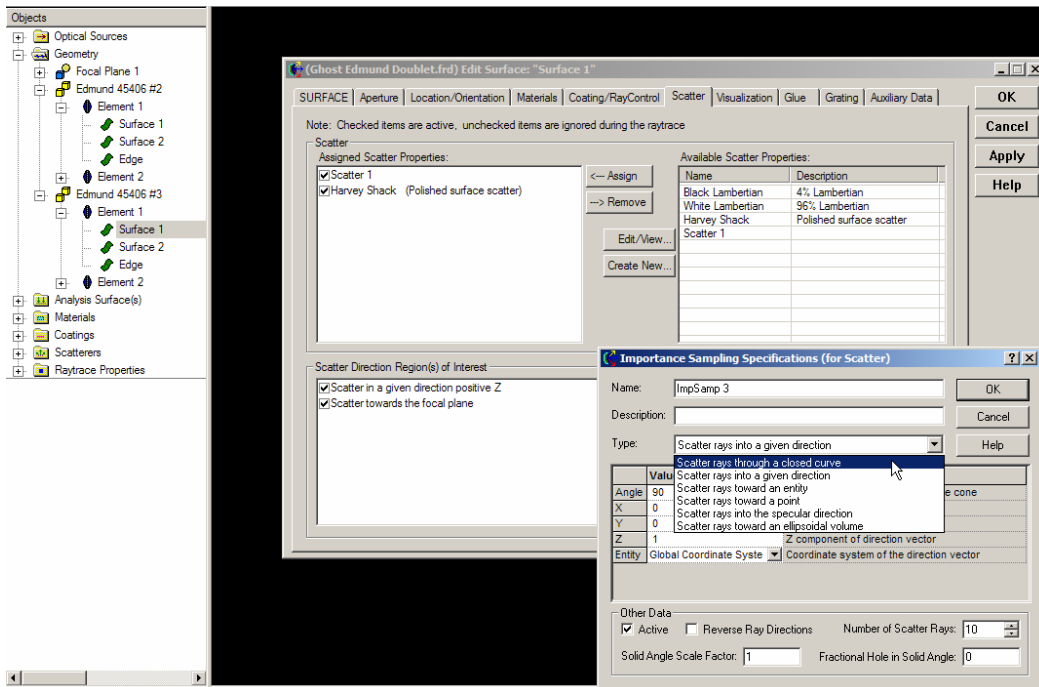


Figure 13 – Importance sampling menu options as applied to a specific surface. Multiple scatter properties can be applied with multiple importance sampling targets as shown. Notice in this figure that both a MIE Scattering scatter property and a Harvey Shack polished surface scatter is assigned and that two importance sampling targets are defined one specifying a direction and one towards a surface, the focal plane.

7. How does **FRED** keep track of thermal scatter paths?

FRED has a stray light report that is available after doing an advanced ray trace when selecting the keep ray history option. It is then possible to request from the Tools menu a stray light report detailing how ghost, and scatter paths reach any surface. Figure 14 shows the advanced ray trace dialogue with both the create/use ray history file option and determine ray path options checked. Figure 15 shows the stray light report for a simple Cassegrain telescope detailing stray light from a source entering the telescope off-axis at 5 degrees. From this report we can see that the largest stray light source at 5 degrees off-axis is off the side of the secondary struts for 73% of the power reaching the detector from this path. The total power is shown in the 4th column showing that only .00197290 Watts reach the detector from this path.

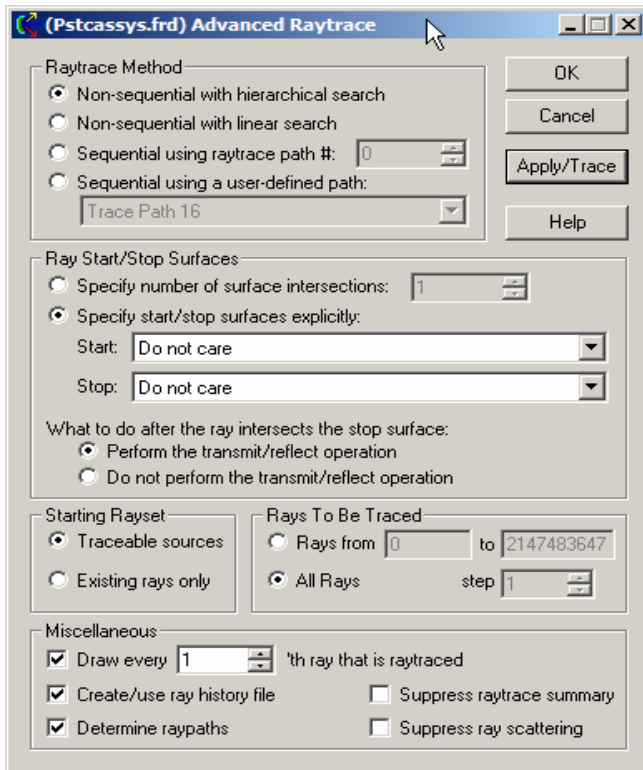


Figure 14 – Advanced Ray trace dialogue

with create/use ray history file option and determine ray path options checked

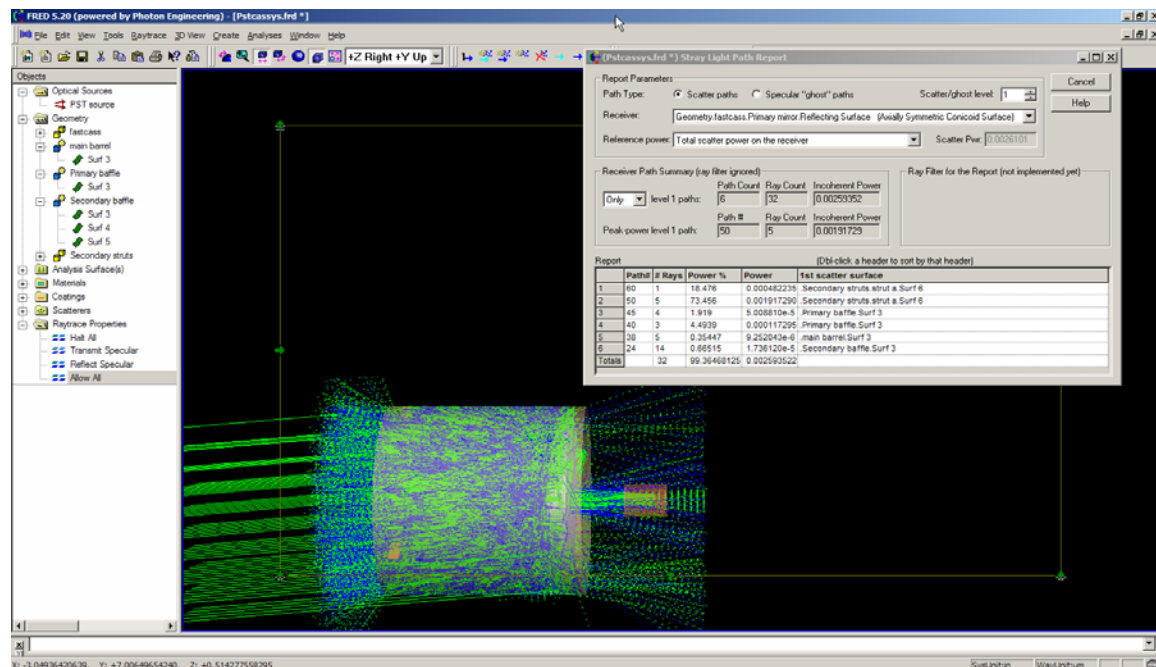


Figure 15 – Stray light report spreadsheet is shown that can be used to track both scatter and ghost paths for any level. The report section shown shows the path number, number of rays, percentage of power and the total power for each reported path along with a descriptive detailing the surfaces.

8. How does **FRED** iterate across multiple pixels on the detector to create a thermal image?

FRED has a built-in compiled Basic scripting language. Almost all Graphical User Interface (GUI) commands are available from the Visual Basic scripting language. **FRED** also has Automation Client and Server capability which can be called or call other Automation enabled programs like Excel. So it is possible to define multiple off-axis sources which in this case are the pixels on the detector and ray trace backward using the For Next loops in the **FRED** Basic scripting language to scan upward and downward across the pixels of the detector. Notice that the BASIC script in Figure 16 writes all the power information to a data file that can be later recalled and viewed using the FRED 3D chart viewer as shown demonstrated in Figure 11.

Figure 16 – **FRED Basic Script to create a thermal image of the teapot**

```
'script for calculating thermal image map
'edited rnp 4 november 2005

'declarations
Dim op As T_OPERATION
Dim trm As T_TRIMVOLUME
Dim irrad(22,22) As Double 'make consistent with sampling
Dim temp As Double
Dim emiss As Double

'setup
nx = 21
ny = 21
numRays = 1000
minWave = 7 'microns
maxWave = 11 'microns
sigma = 5.67e-14 'watts/mm^2/deg k^4

Print ""
Print "THERMAL IMAGE CALCULATION"

detnode = FindFullName( "Geometry.detector.Surf 1" )
Print "found detector array at node " & detnode

srcnode = FindFullName( "Optical Sources.DiffDetArea" )
Print "found differential detector area at node " & srcnode

GetTrimVolume detnode, trm
detcx = trm.xSemiApe
dety = trm.ySemiApe
area = 4 * detcx * dety
Print "detector array semiaperture dimensions are " & detcx & " by " & dety
Print "sampling is " & nx & " by " & ny

'reset differential detector area dimensions to be consistent with sampling
pixelx = 2 * detcx / nx
pixely = 2 * dety / ny
SetSourcePosGridRandom srcnode, pixelx / 2, pixely / 2, numRays, False
Print "resetting source dimensions to " & pixelx / 2 & " by " & pixely / 2

'zero out irradiance array
For i = 0 To ny - 1
    For j = 0 To nx - 1
        irrad(i,j) = 0.0
    Next j
Next i

'main loop
EnableTextPrinting( False )

ypos = dety + pixely / 2
For i = 0 To ny - 1
    xpos = -detcx - pixelx / 2
    ypos = ypos - pixely

    EnableTextPrinting( True )
    Print i
    EnableTextPrinting( False )

    For j = 0 To nx - 1
        xpos = xpos + pixelx
        'shift source
        LockOperationUpdates srcnode, true
        GetOperation srcnode, 1, op
        op.val1 = xpos
        op.val2 = ypos
        SetOperation srcnode, 1, op
        LockOperationUpdates srcnode, false
        'ray trace
```

```

DeleteRays
CreateSource srcnode
TraceExisting 'draw
'radiometry
For k = 0 To GetEntityCount()-1
  If IsSurface( k ) Then
    temp = AuxDataGetData( k, "temperature" )
    emiss = AuxDataGetData( k, "emissivity" )
    If ( temp <> 0 And emiss <> 0 ) Then
      ProjSolidAngleByPi = GetSurfIncidentPower( k )
      frac = BlackBodyFractionalEnergy ( minWave, maxWave, temp )
      irrad(i,j) = irrad(i,j) + frac * emiss * sigma * temp^4 *
ProjSolidAngleByPi
    End If
  End If
Next k

Next j

Next i
EnableTextPrinting( True )

'write out file
Open "C:\Documents and Settings\teapotimage.dat" For Output As #1
Print #1, "GRID " & nx & " " & ny
Print #1, "1e+308"
Print #1, pixelx & " " & pixely
Print #1, -dety+pixelx/2 & " " & -dety+pixely/2

maxRow = nx - 1
maxCol = ny - 1
For rowNum = 0 To maxRow
  row = ""
  For colNum = maxCol To 0 Step -1
    row = row & irrad(colNum,rowNum) & " "
  Next colNum
  Print #1, row
Next rowNum
Close #1
Print "all done!!"

```