USERS' MANUAL

for

MONT2D - Version 2.6

and

MONT3D - Version 2.3

by

James D. Maltby
L-470
Lawrence Livermore National Laboratory
Livermore, CA 94550
(510) 422-2688
Internet Address: maltby1.llnl.gov

and
Charles N. Zeeb, John Dolaghan and Patrick J. Burns
Department of Mechanical Engineering
Colorado State University
Fort Collins, CO 80523
(303) 491-6120 and 7479
Internet Addresses:
czeeb@carbon.LANCE.ColoState.EDU
dolaghan@carbon.LANCE.ColoState.EDU
and
pburns@westnet.net

February, 1994

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Background

This manual encompasses both the 2-D and the 3-D Monte Carlo radiative exchange factor computer codes known as "MONT2D" and "MONT3D," respectively. The codes were developed in the Department of Mechanical Engineering at Colorado State University (CSU) beginning with the work of Scott Statton in 1983 [Statton, 1983] and continuing with the work of James D. Maltby [Maltby, 1987; Maltby, 1990]. The codes are used to calculate radiative exchange factors for enclosures with a nonparticipating medium. The focus is complex geometries rather than sophisticated physics. The types of geometries which can be simulated are shown in Figure 1.1. Note that the 2-D code is capable of simulating both axisymmetric and prismatic geometries, as shown in views (a) and (b), respectively. The 3-D code is capable of simulating geometries modelled as assemblages of generalized quadrilaterals, which are constrained to be flat. Curved surfaces must be approximated by a sufficient number of flat surfaces to "capture" the curvature. Surfaces may absorb photons, or they may reflect or transmit them specularly and/or diffusely. All exterior surfaces must be non-transmissive (it is left to the user to ensure this). All material radiative properties may be explicit functions of the incident photon angle, and be dependent upon energy through the band wavelength formulation. Mark Havstad and Charlie Landram at the Lawrence Livermore National Laboratory (LLNL) have exhaustively exercised the 2-D code, checking it for validity. Similar exercises have been conducted by Donald L. Brown of LLNL and Katherine Bryan of Oak Ridge National Laboratory with the 3-D code.

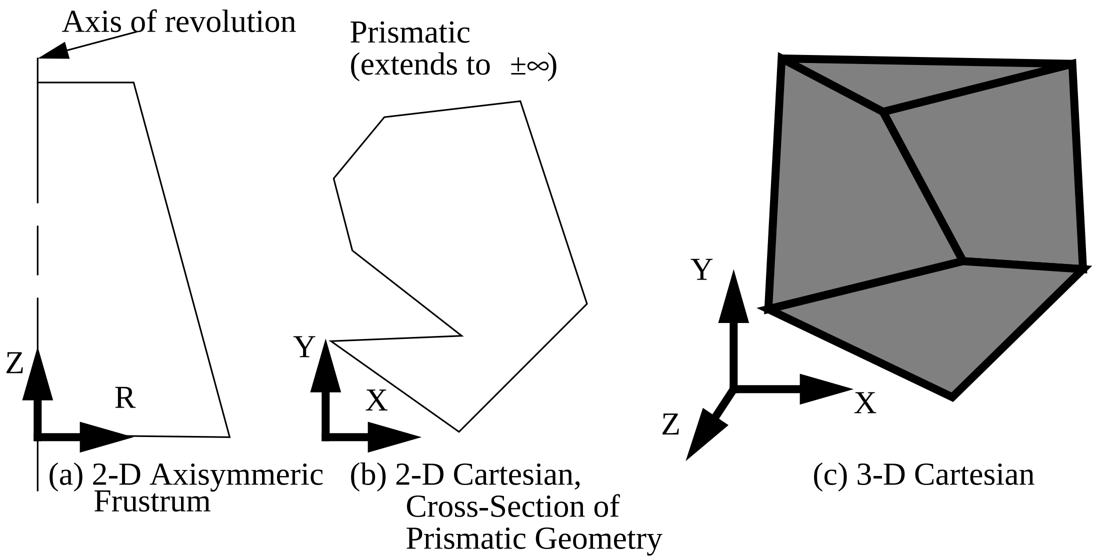


Figure 1.1 Types of Geometries Which Can be Simulated

The codes run on Unix systems, and have been extensively tested on Sun workstations. As several functions at the level of the operating system exist in the code, it may be necessary to modify the code for other Unix systems. Should such problems arise, one of the authors should be contacted. The details for obtaining the codes may be obtained from James Maltby (510) 422-1512, maltby@icdc.llnl.gov.

A number of related publications exist wherein the code has been applied to various problems, including a detailed test of the Separator Development Facility (SDF) [Maltby, 1987], the calculation of radiative exchange in passive solar enclosures [Maltby and Burns, 1986], and application to the Laser Isotope Separation (LIS) process [Burns and Pryor, 1989]. Two recent publications of interest are Burns et al., 1990; and Maltby and Burns, 1991. Finally, an alternative approach for diffuse view factors is the FACET code of Shapiro [Shapiro, 1983].

A number of related codes are available in the public domain. These include the post-processor SMOOTH [Dolaghan et al., 1992] - which smooths the results and establishes reciprocity; the graphical post-processor MPLOT [Shivaswamy, Nagesh and Burns, 1994], which runs on Sun workstations; MONT3E [Crockett et al., 1990] - a code for simulating electron transport in three dimensions in the presence of a spatially nonuniform magnetic field; and SPUT3D [Dolaghan, 1991] - a code for simulating molecular redistribution during sputtering in three-dimensional enclosures. Information on these codes may be obtained from the authors of this report.

1.2 Theoretical Formulation

The present computer codes are formulated in the Monte Carlo style where a large number of photons are emitted from each surface and "traced" until each is absorbed by another surface. Using the subscripts *i* and *j* to denote the emitting and absorbing surfaces, respectively and the superscript *k* to denote the wavelength band, it can be shown that

$$\dot{Q}^k_{i\to j} = \frac{\varepsilon^k_i N^k_{ij}}{N^k_i}\sigma T^4_i F^k_i A_i \qquad (1.1)$$

where:

$\dot{Q}^k_{i\to j}$	= one way rate of radiative heat transfer emitted from surface *i* and absorbed by surface *j* in wavelength band *k* (W).
ε^k_i	= emittance of surface *i* in wavelength band *k*.
N^k_{ij}	= number of photons emitted in wavelength band *k* from surface *i* and absorbed by surface *j*.
N^k_i	= total number of photons emitted in wavelength band *k* from surface *i*.
σ	= Stefan-Boltzman constant, $5.669\text{x}10^{-8}\,(\text{W}\bullet\text{m}^{-2}\bullet{}^{\circ}\text{K}^{-4})$.
T_i	= absolute temperature of surface *i* $({}^{\circ}\text{K})$.
F^k_i	= fraction of the blackbody energy of surface *i* in wavelength band *k*.
A_i	= area of surface *i* (m^2).

The ordinary definitions of the exchange factor E^k_{ij} and the exchange fraction F^k_{ij} are as follows:

$$E^k_{ij} = \varepsilon^k_i F^k_{ij} = \varepsilon^k_i \frac{N^k_{ij}}{N^k_i} \qquad (1.2)$$

The following rules apply to these quantities:

$$\sum_i E_{ij}^k = \varepsilon_i^k \qquad (1.3)$$

or:

$$\sum_i F_{ij}^k = 1 \qquad (1.4)$$

Equations (1.3) and (1.4) express conservation of energy (photons) since all photons must be absorbed by a surface. Given that there must be zero net heat flow between isothermal surfaces, the second law of thermodynamics (entropy principle) follows from equation (1.1) as:

$$E_{ij}^k A_i = E_{ji}^k A_j \qquad (1.5)$$

or:

$$\varepsilon_i^k F_{ij}^k A_i = \varepsilon_j^k F_{ji}^k A_j \qquad (1.6)$$

The net radiative exchange in wavelength band *k* from surface *i* to surface *j* is then:

$$\dot{Q}_{ij}^k = E_{ij}^k A_i \sigma \left(F_i^k T_i^4 - F_j^k T_j^4 \right) \qquad (1.7)$$

and

$$\dot{Q}_{ij}^k = \varepsilon_i^k F_{ij}^k A_i \sigma \left(F_i^k T_i^4 - F_j^k T_j^4 \right) \qquad (1.8)$$

Either of the relations expressed as equation (1.5) or (1.6) may be used to test the exchange factors or the exchange fractions for consistency (convergence). Indeed, either of these relations may be used to manipulate the values in the matrix if either of equations (1.3) or (1.4) is used as a constraint.

1.3 View Factors

The codes may be used to compute view factors, valid for diffuse reflectances independent of incident angle. To do so, all entries for curves of specular and diffuse reflectances and transmittances must be set to 0.0, resulting in an emittance of 1.0 (“black”) for all surfaces. In this limiting case, exchange factors for blackbody surfaces are equal to view factors.

1.4 Code Implementation

The codes are typically used as “preprocessors” for thermal balance studies. As such, the matrix of exchange factors output from the programs are used as input to a thermal balance code. MONT2D and MONT3D are designed to be compatible with the thermal analysis codes TOPAZ2D [Shapiro, 1985] and TOPAZ3D [Shapiro, 1986], respectively, developed at Lawrence Livermore National Laboratory by Art Shapiro. SMOOTH is a postprocessor designed to take advantage of reciprocity to improve the accuracy of the estimates. It operates on the output of MONT2D and MONT3D and produces output compatible with TOPAZ2D and TOPAZ3D.

Fundamentally, the geometry and the material properties are the only quantities necessary to establish the exchange factors. The exchange factors result from the interaction between the geometry and the material properties in a complex fashion, and are unique to a particular geometry/material property combination. Therefore, it is not possible to extend a set of exchange factors calculated for a particular geometry/material property combination to another geometry/material property combination, even if only the geometry or only the material properties vary. After any changes the problem must be rerun. However, if view factors are calculated (all properties black), arbitrary diffuse reflectances may be included in the thermal analysis code since view factors are dependent on geometry alone. Note that diffuse exchange with finite reflectances calculated from the present codes will yield different answers than those obtained using the radiosity/irradiation approach, as the assumption in the radiosity approach is uniform radiosity/irradiation, which is not so for the present code. Specularity or material property dependence on incident angle may not be modelled using view factors.

In addition to radiation exchange, this code can also be used to simulated some special cases of other transport problems in enclosures. If the Knudsen number is much less than 1, rarified gas dynamics problems can be simulated (a vacuum vessel). Also, some simple cases of molecular sputtering in an enclosure can be simulated.

2. COMPUTER CODE BACKGROUND

This chapter presents the background information necessary to run the Monte Carlo computer code. The first three subsections define the geometry of a radiating enclosure in terms of nodes and radiating surfaces. The fourth is a note on the surface concatenation option, while the fifth subsection describes the radiative properties to be specified versus incident cone angle and material types. The sixth describes cosine power dependance of diffuse reflection and transmittance and the seventh explains the available shading algorithms. The eighth subsection discusses accuracy versus the number of photons emitted. The ninth subsection is next on debugging information generated by the program. This is followed by the tenth subsection on the restart capability, whereby a run may proceed from a previously calculated state. The chapter concludes with a discussion of the pseudo-random number generator.

2.1 Nodes

The description of the geometry of the radiation enclosure begins with the specification of the nodes (also called nodal points) of the enclosure. A node is a point in three-dimensional space defined by three coordinates in a Cartesian coordinate system *{X, Y, Z},* as depicted in Figure 2.1. For the 2-D cases, the third coordinate need not be specified. In the case of axisymmetric geometries, the cylindrical coordinates *{R, Z}* are input. Each node, *N,* in the enclosure is assigned a node number which must be positive and a member of the closed, full set $N \varepsilon \{1,NUMNP\}$, where NUMNP is the total number of nodes for the problem. Once the nodes of the enclosure have been specified, radiating surfaces can be defined by specifying sets of nodal points.

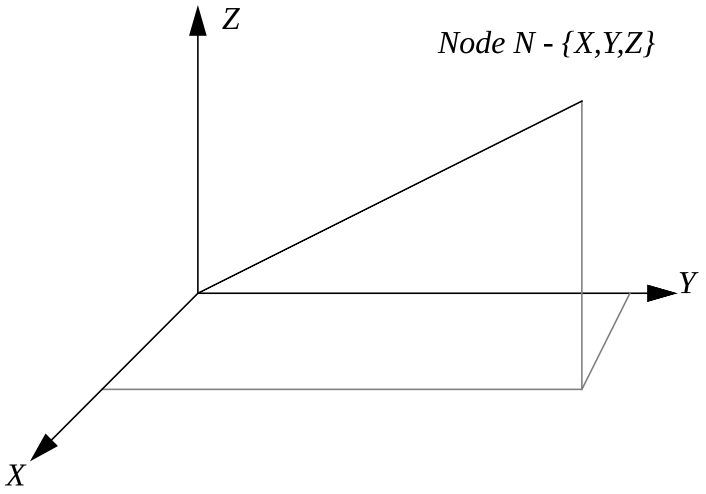


Figure 2.1 Global Coordinate System

2.2 2-D Surfaces

A 2-D radiating surface is formed by specifying 2 nodal points and a surface number. The surface numbers vary between 1 and the total number of surfaces, inclusive. A surface interacts on one side only, as seen in Figure 2.2(a). It is very important to note that the outward normal ***n*** of the radiating surface is always defined as pointing outward to the right as one proceeds from node N_1 to N_2. The surface normal must always point into the enclosure. The “back” of the surface opposite the surface normal is transparent as far as the code is concerned, and a leak will result from incorrect node numbering. Photon emissions are uniformly distributed in area - uniform spacing for Cartesian geometries, and quadratic spacing in R for axisymmetric geometries. Additional detail on this exists in Section 2.7.

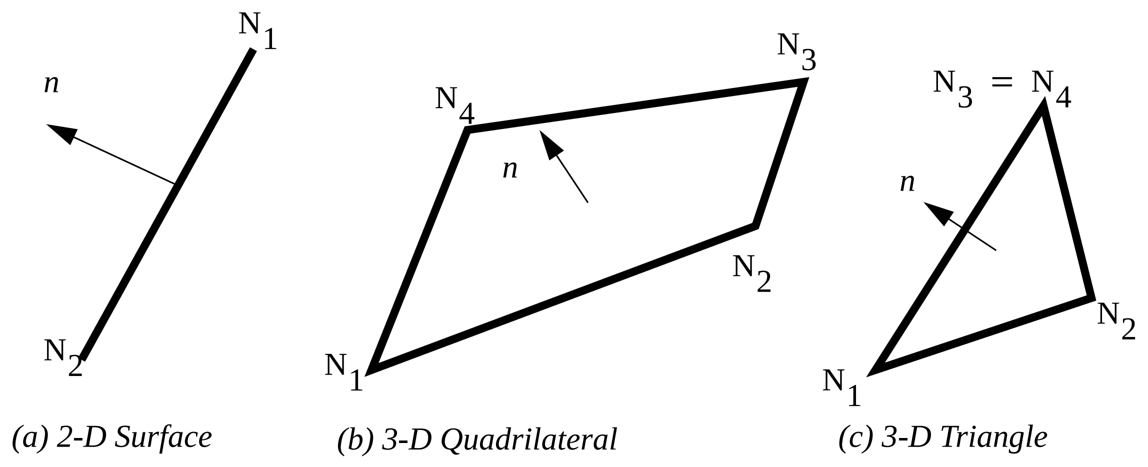
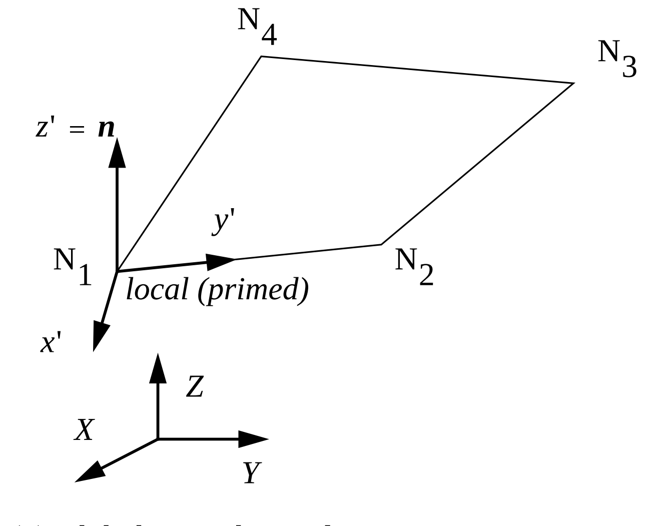


Figure 2.2 Radiating Surface Geometries

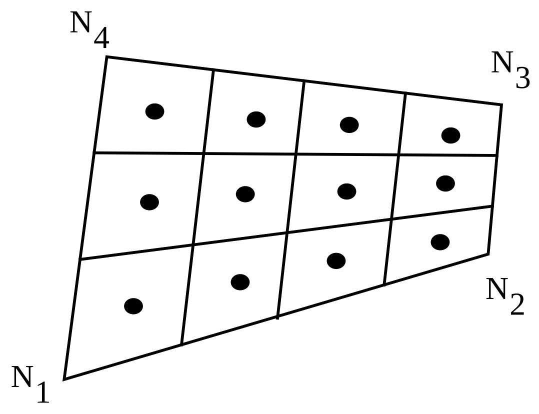
2.3 3-D Surfaces

Three-dimensional radiating surfaces consist of planar quadrilaterals or triangles and are defined by specifying 4 nodal points. If four distinct points are specified, then the surface is a generalized quadrilateral, as shown in Figure 2.2(b). If it is desired to use triangular surfaces, then the last two node numbers must be identical as shown in Figure 2.2(c). The outward normal for both types of surfaces is defined consistent with the right-hand rule, i.e. if the fingers of the right hand are curled in the direction of increasing nodal point number $(N_1 \rightarrow N_4)$, then the thumb of the right hand indicates the direction of the outward normal. This results in a counterclockwise convention for nodal point numbering, when “viewing” the radiating side of the surface from above. Two further caveats apply here: none of the interior angles may exceed $180°$ (i.e., no concave corners are permitted), and the four nodes must be coplanar (to within a small tolerance). This tolerance can be set, see section 3.3 for more details.

Each radiating surface (both 2-D and 3-D) is assigned a material number, and all material properties are independent of spatial position on a single surface. Referring to Figure 2.3(a), one observes the local (primed) coordinate system with vertex at N_1, and axes as shown. Photons are emitted from a matrix of points across a surface, defined by linear functions in the local coordinate system. The surface is divided into equally spaced regions, and emission occurs from the centroids of these subsurfaces. Figure 2.3(b) depicts this subdivision for 4 divisions in the y' direction and 3 divisions in the x' (here, editorial license has been taken as, in general, x' is not aligned with the lines $N_1 - N_4$ or $N_2 - N_3$, as it is normal to the y' and z' axes) direction, where the dots indicate the photon emission points. The number of photons emitted from each point is scaled by the area of its subsurface to eliminate bias that would otherwise result in “hot spots.” At least 3 divisions in x' and y' are recommended for uniform surface emission (default is 5). Additionally, at least 100 photons should be emitted from each emission point to prevent round-off problems in scaling photon emissions to subsurface areas.



(a) Global - Local Coordinate Systems



(b) Photon Emission Points

Figure 2.3 Three-Dimensional Surface Details

2.4 Surface Concatenation

Previous versions of MONT2D and MONT3D permitted surface concatenation. Current versions do not due to the limitations associated with error checking and restart (it is impossible to recover restart information from concatenated information, as some information is lost during the concatenation process). If it is desired, concatenation can be done by *a posteriori* operating on the exchange factor matrices.

2.5 Material Properties

The material properties are defined in terms of a local spherical coordinate system. Figure 2.4 defines the cone angle, θ, and the azimuthal angle, ϕ.

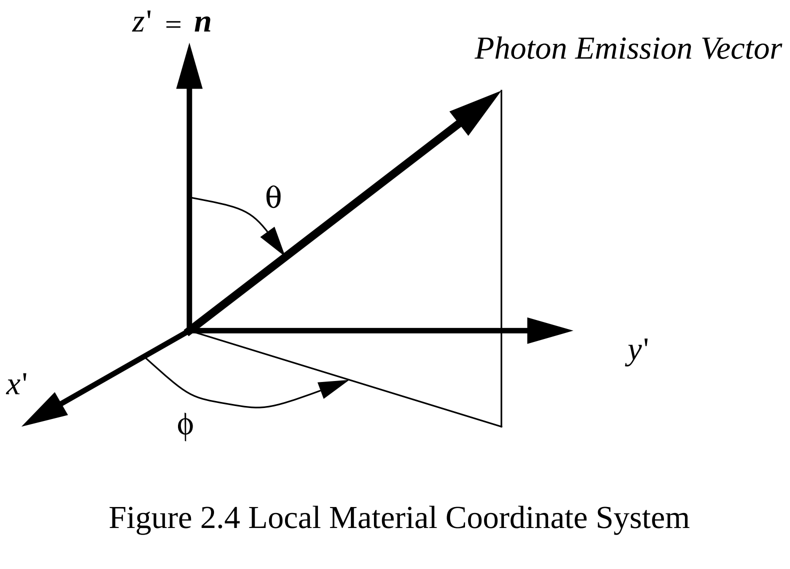


Figure 2.4 Local Material Coordinate System

All material properties are independent of azimuthal angle, ϕ, and dependent upon cone angle, θ. The properties are defined as constant (gray) within a particular radiative band *k*, so that Kirchoff's law applies within each band. Explicitly:

$$\varepsilon^k(\theta) = \alpha^k(\theta) = 1 - \tau_s^k(\theta) - \tau_d^k(\theta) - \rho_s^k(\theta) - \rho_d^k(\theta) \qquad (2.1)$$

where:

$\varepsilon^k(\theta)$	= emittance in wavelength band *k* at outgoing cone angle, θ
$\alpha^k(\theta)$	= absorbance in wavelength band *k* at incident cone angle, θ
$\tau_s^k(\theta)$	= "specular" transmittance in wavelength band *k* at incident cone angle, θ
$\tau_d^k(\theta)$	= "diffuse" transmittance in wavelength band *k* at incident cone angle, θ
$\rho_s^k(\theta)$	= specular reflectance in wavelength band *k* at incident cone angle, θ
$\rho_d^k(\theta)$	= diffuse reflectance in wavelength band *k* at incident cone angle, θ

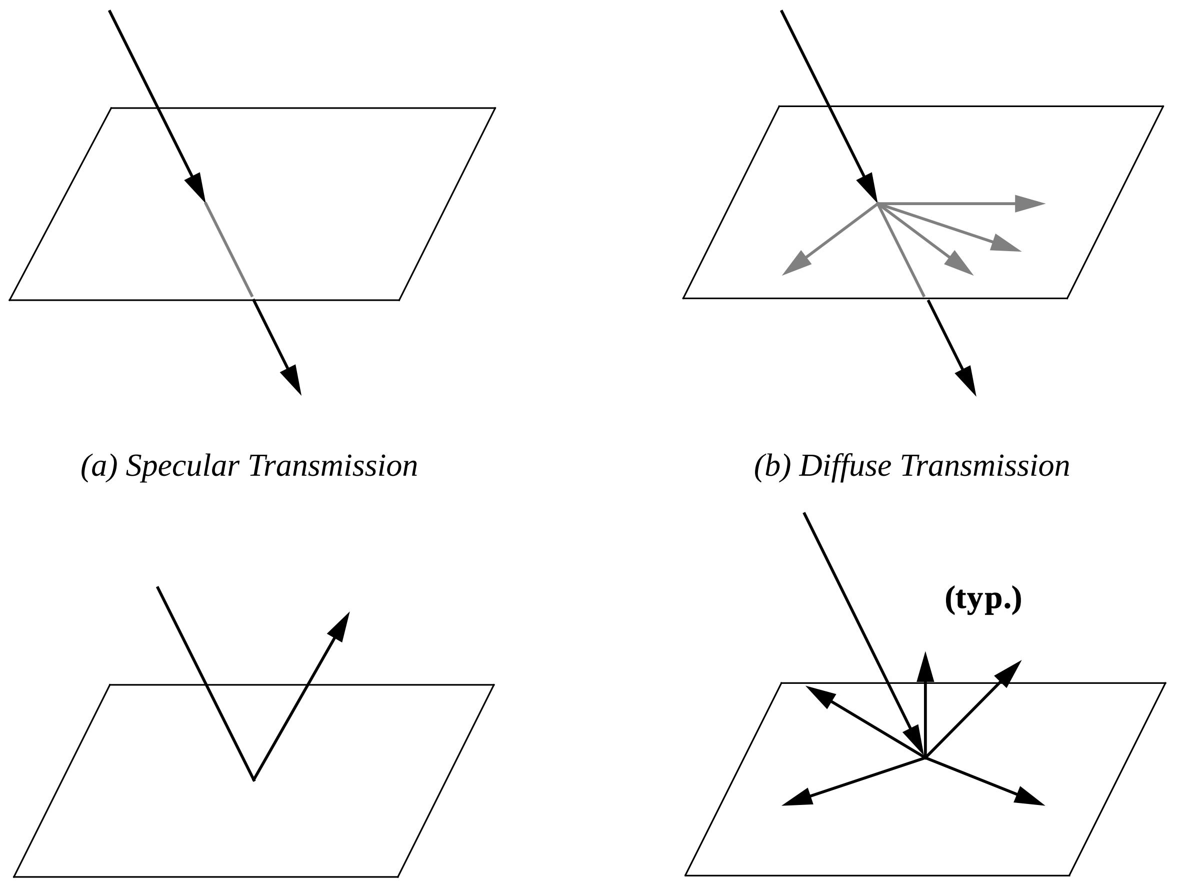
Hereinafter, we drop the explicit dependence upon wavelength band *k*, and carry an implicit dependence. Note that all properties may be considered in terms of probability, i.e. $\rho_s(\theta)$ is the probability that a photon of incident angle θ will be "specularly" transmitted, $\rho_d(\theta)$is the probability that a photon of incident angle θ will be diffusely reflected, etc. As the emittance and absorptance within a given wavelength band are equal by virtue of Kirchoff's law, these properties are determined as the complement of the others. Thus, specification of the specular and diffuse reflectances and transmittances as functions of the incident angle uniquely defines the properties within one wavelength band for a material.

Photon / material interactions of five types may occur. Figure 2.5(a) depicts a "specular" transmission, whereby a photon passes straight through a specularly transmitting surface, with no change in direction. Figure 2.5(b) depicts a "diffuse" transmission, where the photon is transmitted, but its direction is uniformly distributed in solid angle, weighted by projected surface area. This is akin to a diffuse reflection, but from the "back" of the surface.

Figure 2.5(c) depicts a specular (mirrorlike) reflection where "the angle of incidence is the angle of reflection," and Figure 2.5(d) depicts a diffuse (random) reflection, with an equal probability of reflection in any "direction." (Here again, we have exercised editorial license with this statement. In truth, a diffuse reflection has an equal probability of reflection in any solid angle, weighted by projected surface area). Finally, the photon may be absorbed.

2.5.1 Material Property Curves

Figure 2.6 depicts the material properties as functions of the incident cone angle, θ_i within a wavelength band *k*. At any particular value of θ, e.g. $\theta = \theta^*$, the incident photon has the following probabilities: $\tau_s(\theta)$, $\tau_d(\theta)$, $\rho_s(\theta)$, $\rho_d(\theta)$, and $1-\tau_s(\theta)-\tau_d(\theta)-\rho_s(\theta)-\rho_d(\theta)$ for specular and diffuse transmission, specular and diffuse reflection, and absorption (and emission), respectively. Each of these curves must be input by point value as a function of cone angle for every wavelength band for each material. The computer code parabolically interpolates between each 3 successive points entered. Care must be taken to: (1) include bounding points of $\theta = 0^o$ and $\theta = 90^o$ (since no extrapolation is done), and (2) to include enough points, varying smoothly, to result in good interpolation (i.e., discontinuous jumps must be input as "steep" parabolas with 3 non-coincident points used to define the jump).



(a) Specular Transmission

(b) Diffuse Transmission

(c) Specular Reflection

(d) Diffuse Reflection

Figure 2.5 Photon-Material Interactions

2.5.2 Material Types and Photon Emission

Seven different material types are available to provide flexibility to the material model. These material types allow different combinations of photon/material interactions to be defined. Table 2.1 summarizes the material types and indicates which material property curves must be entered.

In addition to photon/material interactions the material type also determines the type, if any, of emission that will take place. Three possible emission modes are modeled; emission according to user input function, beam emission, and normal emission. Note that if the emittance of a material is 0 for all values of theta, then no photons will be emitted from that material unless it is of type 1.

Figures 2.7(a), (b), and (c) show the conventions for θ and ϕ. The cone angle, θ, is always defined from the surface normal, and the azimuthal angle, ϕ, is positive counterclockwise (when viewed from above) from the surface *x*'-axis.

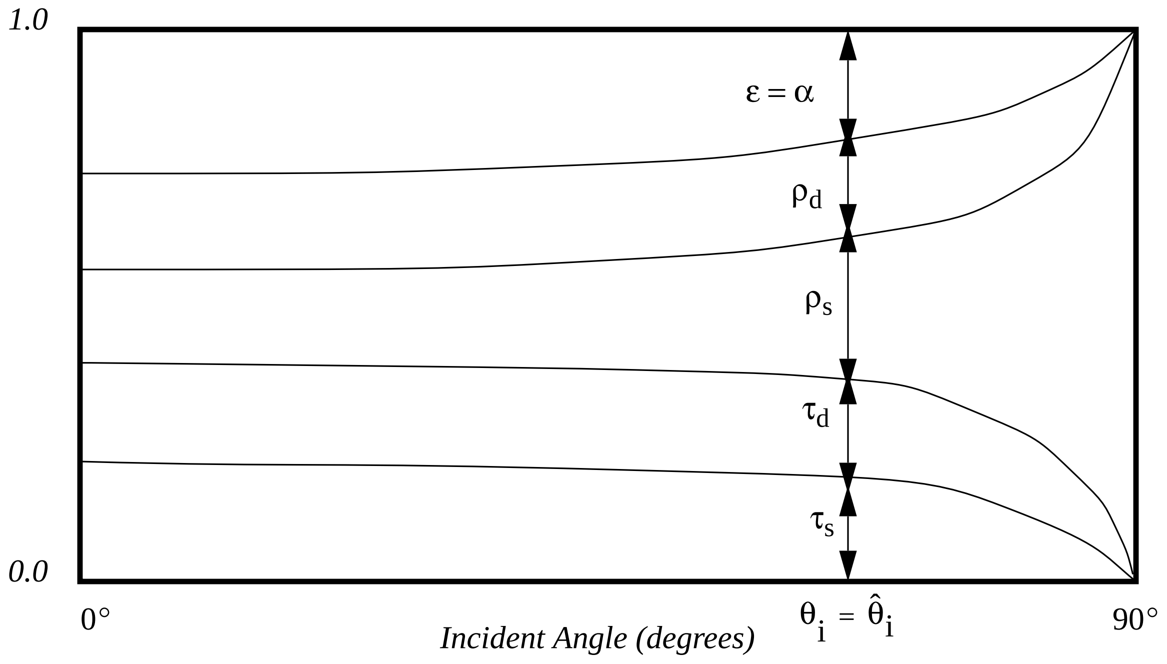


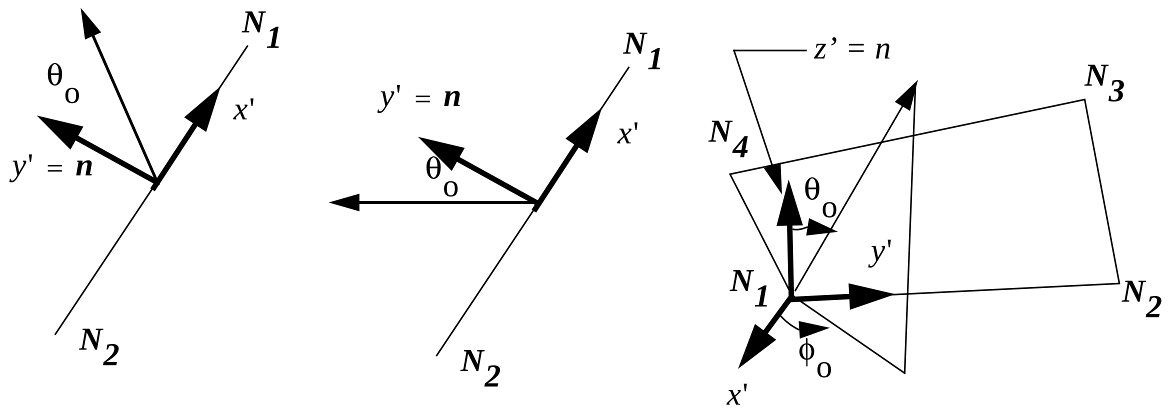
Figure 2.6 Material Properties vs. Incident Cone Angle

Table 2.1 Material Property Summary

Type	Emission	Interactions	Input Curves
2	Function	$\tau_s(\theta)+\rho_s(\theta)+\rho_d(\theta) + \alpha(\theta) = 1$	3: $\tau_s(\theta), \rho_s(\theta), \rho_d(\theta)$
1	Beam	$\tau_s(\theta)+\rho_s(\theta)+\rho_d(\theta)+\alpha(\theta) = 1$	3: $\tau_s(\theta), \rho_s(\theta), \rho_d(\theta)$
0	$\varepsilon(\theta)$	$\tau_s(\theta)+\rho_s(\theta)+\rho_d(\theta)+\alpha(\theta) = 1$	3: $\tau_s(\theta), \rho_s(\theta), \rho_d(\theta)$
-1	“	$\tau_d(\theta)+\rho_s(\theta)+\rho_d(\theta)+\alpha(\theta) = 1$	3: $\tau_d(\theta), \rho_s(\theta), \rho_d(\theta)$
-2	“	$\tau_s(\theta)+\tau_d(\theta)+\rho_s(\theta)+\rho_d(\theta)+\alpha(\theta) = 1$	4: $\tau_s(\theta), \tau_d(\theta), \rho_s(\theta), \rho_d(\theta)$
-3	None	Perfect specular reflection	None
-4	None	Perfect diffuse reflection	None

In the 2-D code, the x'-axis is defined along the ray pointing from node 2 to node 1. Figures 2.7(a) and (b) show the orientation of the emission vector for fixed θ_o, and $\phi_o = 0^o$ and 180^o degrees, respectively. Values of ϕ_o other than 0^o or 180^o are of limited utility in the 2-D code, since they point into or out of the plane of analysis.

For the 3-D code, the x'-axis is defined perpendicular to the ray joining nodes 1 and 2 (***y***'-axis) and the surface normal ***n*** (z'-axis) according to a right-hand rule, as shown in Figure 2.7(c).



(a) 2-D, $\phi_o = 0°$ *(b) 2-D,* $\phi_o = 180°$ *(c) 3-D*

Figure 2.7 Conventions for Outgoing Angles

Care should be taken to number the nodes of a surface correctly to achieve the desired emission direction.

2.5.3 Material Type 2, Emission According to a User-Supplied Function

For material type 2, emission occurs according to a user-supplied function (function *fcn* in subroutine *getang* - in the solution phase). Here, both θ and ϕ must be specified (the example in the code supplies a function only for θ, emission is uniformly distributed in ϕ). The user is cautioned that emission via this option occurs via the acceptance-rejection method, and many trials may occur for each photon emission if the magnitude of the user-supplied function is small. Photon/material interactions are determined by three input curves as shown in Table 2.1.

2.5.4 Material Type 1, Beam Emission

An option (material type 1) is available in the code to simulate beam radiation, with all primary photon emissions from a specified material occurring in a fixed direction. Typical uses would be simulating direct beam sunlight or a laser beam. To use this option, fixed values of ϕ_o and θ_o for a material must be entered by the user, in a format described in Section 3.9. All emissions from a surface using this material will be in the defined direction. However, note that all interactions (absorptions, transmissions, reflections) will depend only on the material property curves defined for that material. The user should be cautious when specifying the beam radiation option, since the reciprocity relations, equations (1.5) and (1.6) will no longer be valid. The other conservation relations still hold.

2.5.5 Material Types 0 Thorough -2, Normal Emission

For material type 0 through -2, emission is a function of θ_o, so all of the photon/material interactions are determined by the input curves. For material 0, three curves are read and $\tau_d(\theta_i)$ is set to zero. Material type -1 also requires three input curves with $\tau_s(\theta_i)$ set to zero. For material type -2, all four curves are input.

2.5.6 Material Types -3 and -4, Perfect Mirrors

Material types -3 and -4 are perfect mirrors (perfect reflectors). Material type -3 is completely specular, and material type -4 is completely diffuse. For surfaces of this material type, no emission occurs, and “zero” exchange factors are written into the exchange matrix file.

2.6 Diffuse Reflectance and Transmittance and Cosine Power Dependance

In general, the probability for emission or re-emission of a photon at the angle θ is proportional to $\cos^r(\theta)$. The value usually picked for r is 1. For metals, r is usually less than 1 and for nonmetals, r is usually greater than 1. An isotropic distribution for θ is obtained by setting $r = 0$. The effect of r on the θ distribution is shown in Figure 2.8. This figure displays $\cos^r\theta$ as a function of θ. The curves have been normalized to their areas. The normalization is such that for $r = 0$ (isotropic distribution), the curve has the value of 1 at all angles.

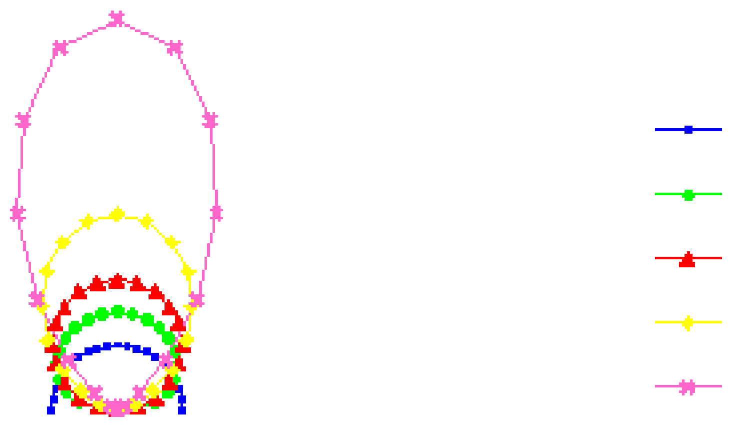


Figure 2.8 Normalized Probability of Re-emission as a Function of r and θ

The code allows the user to specify separate r’s for diffuse reflectance and transmittance for each material; see section 3.9. To prevent violation of Kirchoff’s law for emission, r is always set to 1.

2.7 Shading

If any shading exists in the geometry, a photon’s path may intersect a number of surfaces. The distance from the emission point to the different intersection points must then be computed, and the closest point chosen as the true intersection point (the surface first encountered). For large problems, this results in significant expense. To reduce execution time for large problems with shading, the Margolies shading algorithm [Margolies, 1986] has been implemented. The geometry is divided into a series of grid cells, resulting in rectangles and rectangular parallelepipeds for the 2-D and 3-D geometries, respectively. The photon is traced from grid cell to grid cell, and the search is done within each grid cell only over those surfaces which exist either wholly or partly within that cell. The situation is depicted in Figure 2.9 for a 2-D geometry. This algorithm has resulted in reductions in execution time for 2-D Cartesian enclosures of factors of 2 to factors of

20, with factors of 5 to 7 being typical. For 3-D enclosures, due to the higher overhead incurred in tracing through a 3-D grid, reductions in execution time of factors of 2 to factors of 4 are typical. The optimum grid differs from problem to problem, and must be empirically determined by the user. However, good starting guesses are about 100 total grid cells (about 10 X 10 in two dimensions, and 5 X 5 X 5 in three dimensions).

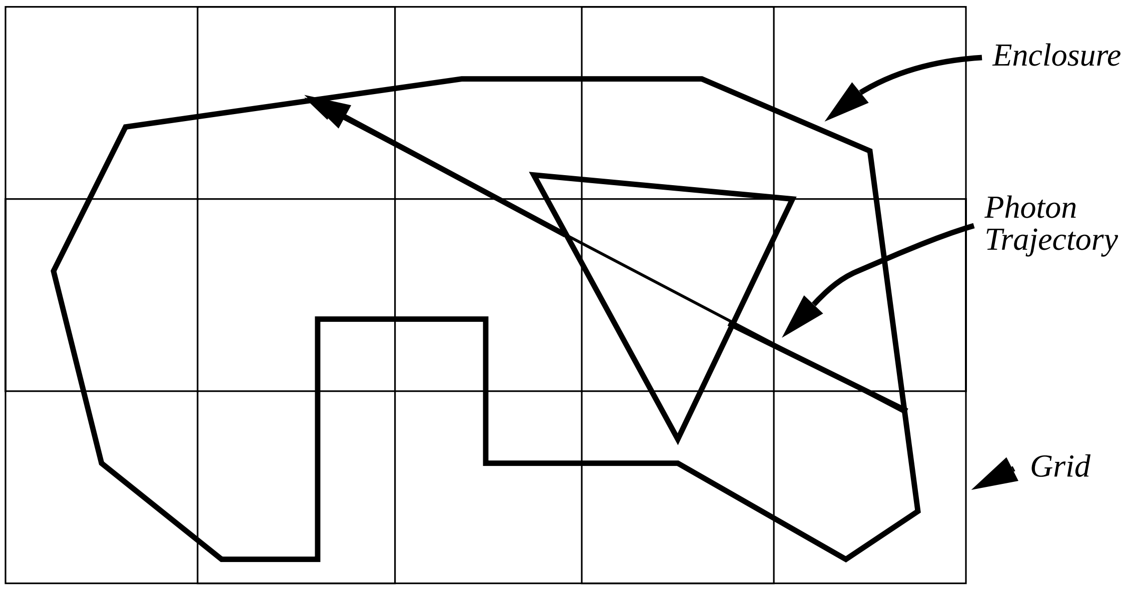


Figure 2.9 2-D Illustration of the Grid Shading Algorithm

Two options exist for defining the grid: (1) a grid generated by the program, with equally spaced grids in the X, the Y, and the Z directions, or (2) a user-defined grid. For option (1), the user need specify only the number of grids in the X, the Y, and the Z directions, while for option (2) the user must specify the grid locations also, as defined in sections 3.4 and 3.11.

2.8 Number of Photons, Convergence and Accuracy

The number of photons emitted is specified for each surface by the relation: # of photons = NBANDS * NDIVX * NDIVY * NPHTON (for the 2-D code, NDIVY equals 1). As Monte Carlo techniques are statistical in nature, “enough” photons must be emitted from each surface to yield a statistically accurate result. This number depends upon the geometry and, to some extent, upon the material properties. As a general rule, greater numbers of surfaces require greater numbers of photons. Execution time increases linearly with number of photons. For moderate problems (about 20 surfaces), it has been found that on the order of 20,000 photons per surface (not subsurface) are required to achieve exchange factors accurate to within about 10%. It is typical to observe convergence in a particular exchange factor as shown in Figure 2.10. The user is cautioned that false convergence may be indicated when comparing two values on the curve as shown. It is therefore wise to check the entire matrix of exchange factors for consistency at several numbers of photons.

To estimate the number of photons required to achieve a given level of accuracy, Table 2.2 is provided. The table gives, for each exchange fraction F_{ij}^{k}, the number of photons, N_{i}^{k} , which

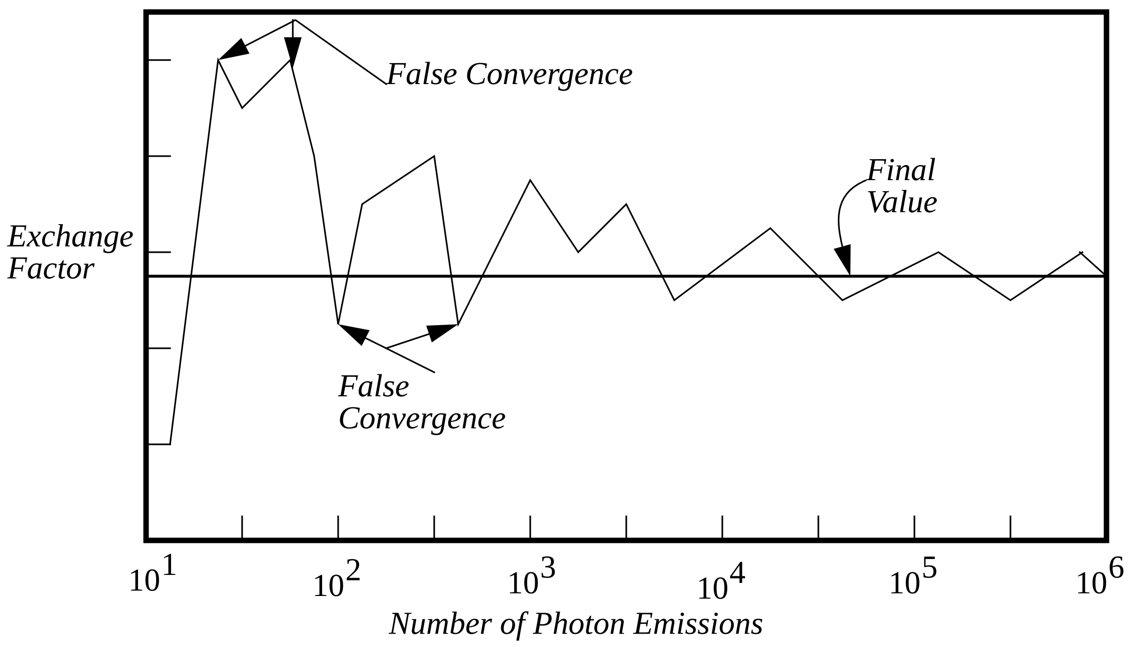


Figure 2.10 Convergence vs. Number of Photon Emissions

must be emitted from surface *i* to achieve 95% confidence that the exchange fraction is within: 1%, 2%, 5%, 10% and 50% of the exact answer. The numbers of photon emissions per surface are calculated from the formula for confidence intervals, C_{ij}^{k}, for the exchange fraction from surface *i* to surface *j* in wavelength band *k* (F_{ij}^{k}), derived by Maltby [Maltby, 1990]:

$$C_{ij}^{k} = Z\sqrt{\frac{1-F_{ij}^{k}}{N_i^k F_{ij}^{k}}} \qquad (2.2)$$

where Z is taken from the standard normal tables, and is 1.96 for 95% confidence. Equation (2.2) yields the fractional accuracy in $F^{k}{}_{ij}$ (n.b., 100 time this value yields the percent accuracy). The codes MONT2D and MONT3D are formulated to attain a specified accuracy for each row *i* of the exchange factor matrix. The program is constructed to loop over successive emissions from each surface *i* if a preset accuracy tolerance is not met after a full surface emission. To explain this, we first note that equation (2.2) provides the confidence interval for only element *ij* of the exchange fraction matrix, when emitting additional photons actually increases the accuracy of all elements in row *i*. Equation (2.2) is modified to account for this with the rationale that exchange fractions affect the accuracy proportional to their size. Thus, we weight each confidence interval by its exchange fraction, sum and then average by dividing this amount by the total number of sides, N_s to yield the *ad hoc* row confidence factor for row *i*, C_i^k :

$$C_i^k = \frac{1}{N_s}\sum_j C_{ij}^k F_{ij}^k = \frac{Z}{N_s}\sum_j\sqrt{\frac{F_{ij}^k(1-F_{ij}^k)}{N_i^k}} \qquad (2.3)$$

If the confidence (as a fraction) for emissions from surface *i* is not met after a full surface emission, then the program continues to perform full surface emissions until either the specified confidence is met, or a maximum number of full surface emissions have occurred. This feature can be used with the restart option to effect a specified accuracy for the surfaces, balanced against CPU usage.

Table 2.2 Accuracy in Exchange Fractions

Exchange	*Level of Accuracy*				
Fraction	*1%*	*2%*	*5%*	*10%*	*50%*
10^{-3}	*38,377,584*	*9,594,396*	*1,535,103*	*383,776*	*15,351*
10^{-2}	*3,803,184*	*950,796*	*152,127*	*38,032*	*1,521*
10^{-1}	*345,744*	*86,436*	*13,830*	*3,457*	*138*

A rough guess of the size of the exchange fractions is the reciprocal of the number of surfaces in the input file. This yields the "average" exchange fraction size, since the sum of any row of the exchange fraction matrix is 1. The number of photons required to be emitted to achieve an "average" level of accuracy may then be estimated from equation (2.2), or obtained from Table 2.2 through interpolation or extrapolation.

Errors in the temperatures calculated from radiative flux balances are smaller than errors in the exchange fractions due to the fourth-root dependence of temperature upon radiative flux. For small errors, one may expect the errors in temperatures to be about one-fourth of the errors in fluxes. Emitting an equal number of photons from each surface may result in a waste of computer time, since some surfaces contribute little to the radiative exchange. A better approach would be to apportion the numbers of emissions to each surface based upon its estimated power output. This is an approach which requires judgement gained through experience with specific geometries, since the power outputs are generally not known *a priori*. In any case, the above approach provides a "potentiometer" which can be used judiciously to adjust solution accuracy.

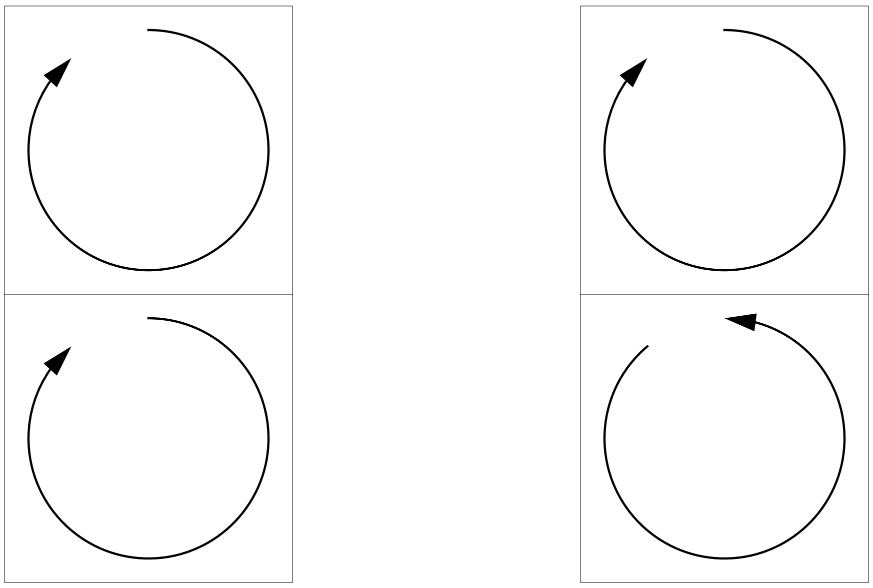
2.9 Debugging, Leak Checking, Lost Photons, and Trajectory Output Capabilities

The codes are equipped to generate information useful in debugging input files. A file containing all the information in the input file (geometry and material properties) is written to disk during the input phase. This file may be used with the stand-alone graphics program MPLOT [Shivaswamy, Nagesh and Burns, 1994] to display the geometry and the material property curves.

An error in the specification of the geometry often results in a "leak" or hole in the enclosure, through which photons may be transmitted and "lost." Leaks may be caused due to disjoint surfaces, missing surfaces, incorrect node numbering on a surface, misplaced nodal points, or insufficient precision in specifying coordinates. During the input phase, the geometry is checked for leaks, and results are written to the output file. Additionally, an ASCII file of potential leaks, identified by type (severity of leak), is written for 3-D geometries. This file may be used with MPLOT to highlight surfaces and sides of surfaces which have been identified as potential problems or leaks. The three types of errors identified are:

Error 1- Reversed Edge:

This error occurs when two surfaces share the same edge. If the surfaces are not defined so that both their surface normals are pointing inward or outward, a reversed edge occurs. An example is given in Figure 2.11. In Figure 2.11(a), the edge between the two surfaces is correct and both surfaces have normals pointing into the page. In Figure 2.11(b), the edge is reversed and the normals of the two surfaces are pointed in opposite directions. To see the type of problem this creates, remember that the surfaces are transparent to photons that hit the back side of the surface. These surfaces are oriented for photons coming in opposite directions. This type of error almost certainly means the enclosure will loose excessive photons resulting in an error termination.


(a) correct edge (b) reversed edge

(Arrows represent direction of increasing node number)

Figure 2.11 Example of a Reversed Edge

Error 2 - No Match Found:

This error occurs when no match was found connecting at least one side of that surface to the side of another surface. This may or may not be okay.

Error 3 - Slip Surface

This error occurs when an edge goes through a node point instead of terminating at it. Although this is sometimes a fatal error, it often is not. If the slip surface creates no “holes” in the geometry, then it should not cause the enclosure to loose photons.

Where photons are lost, the endpoints of each photon ray are written to a separate file, which may then be read by the MPLOT program. Then trajectories of the lost photons may be displayed on the geometry. Because there is no terminus of the ray, a fictitious endpoint is used.

An option is also available to write trajectory information to an output file, which may be subsequently read and plotted by the program MPLOT. This feature is useful in obtaining a “feel” for the underlying physical processes, and to ascertain that the simulation is proceeding as planned. There is a copious amount of information written to the output file during the exercise of this option, so it is suggested that the number of photons emitted per surface be less than about 100.

For more information, the reader is urged to consult the MPLOT documentation [Shivaswamy, Nagesh and Burns, 1994].

2.10 Restart Capability

The codes have been designed to be restarted from a previously computed state. For example, the code may experience a “crash” during execution (for any of a host of reasons). Alternatively, the code may run to completion, and subsequent examination of the answers indicates that they are not sufficiently accurate. In either case, it is desirable to begin a new simulation from the last state available to take advantage of previous work. This prevents waste of computer resources in recomputing information already available. A simulation may be restarted multiple times, until the desired level of accuracy is attained.

To effect this, the current state of the solution must periodically be written to disk, so that it will be available for a restart run. The information is written in unformatted format to disk, and contains the current random seed and a vector of the total number of photons emitted from each surface. This information, together with the current exchange matrix file, is used to effect a restart. The following chapter provides additional detail of the control which can be exercised over writing states to the output file. Chapter 6 gives an example of a restart run.

2.11 Pseudo-Random Numbers

The pseudo-random number generator used is a lagged-Fibonacci [Anderson, 1990] generator. The period of this generator is $(2^{17} - 1)2^{31}$ (= 2.815×10^{14}) if at least one of the initial seeds is odd. (The code ensures that at least one seed is odd.) The generator uses an array of 17 seeds which are generated by a multiple congruential generator from one initial seed. The initial seed can be specified by the user, generated from the current time or set to a default debugging value. See section 3.2.1 for more details.

Each initial seed creates a different sequence of random numbers so different answers are obtained. However, if “enough” photons are emitted to achieve convergence, then the answers, whatever the initial seed, will be identical to within statistical convergence error.

It is not possible to traverse the same sequence of random numbers after a restart run involving emission from more than one surface. Thus, if a run is done with 20,000 photons emitted from each surface, the answers will be different than if an initial run is done with 10,000 photons per surface followed by a restart from this final state where 10,000 additional photons are to be emitted. However, the comments above pertaining to convergence do apply. If “enough” photons are emitted, then the answers will converge (within a statistical tolerance) to a final state independent of the order of emissions.

3. INPUT DECK

The following pages contain the instructions necessary to enable the user to construct the input file (data deck) required by the codes. Default values are used if the input file contains blanks (read as zero by FORTRAN). Input lines are limited to 80 columns in width.

3.1 Title Card

Cols.	Format	Entry	Note
1-48	6A8	Heading to appear on output.	1

Note:

1. Cards with the character "&" in column 1 can be placed anywhere in the data deck for use as comment cards or as spaces. All such cards are ignored.

3.2 Control Cards

3.2.1 Card 1

Cols.	Format	Entry	Note
1-5	I5	Number of dimensions (NDIM).	1
6-10	I5	Number of nodal points (NUMNP).	
11-15	I5	Number of surfaces (NUMEL).	
16-20	I5	Number of materials (NUMMAT).	
21-25	I5	Number of wavelength bands (NBANDS).	2
26-30	I5	Number of photons emitted per band per subsurface (NPHTON).	3
31-35	I5	Maximum number of reflections allowed per photon before a warning is issued (NREFS). (DEFAULT: NREFS = 100)	4
36-40	I5	Maximum number of warnings before the run is aborted (NWARNS). (DEFAULT: NWARNS = 50)	4
41-45	I5	Maximum number of lost photons (NLOST). (DEFAULT: NLOST = 0)	5
46-50	I5	Restart write increment (NINCR). NINCR < 0: Restart run from previously stored state NINCR = 0: No restart information written until the end of the run NINCR > 0: New run; restart file written to disk after every NINCR surfaces	6
51-60	I10	Initial seed for the random number generator (INSEED) INSEED <0: Use the default value for the initial seed INSEED = 0: Obtain the initial seed from the time INSEED > 0: Use the inputted value for the initial seed	

Notes:

1. Must be equal to 2 or 3 for MONT2D or MONT3D, respectively.
2. Radiative properties are defined in wavelength bands, and are constant within a given wavelength band.
3. NPHTON is the number of photons emitted per subsurface per wavelength band so that the number of photons emitted per surface is equal to NPHTON * NBANDS * NDIVX * NDIVY. This is the default value for all surfaces, which may be overridden for any particular surface(s) or all surfaces during the surface input. For the 2-D case, NDIVY equals to 1.
4. The total number of reflections before a run is aborted will be 1 greater than NREFS * NWARNS.
5. Occasionally, due to precision problems, a photon is “lost” (i.e., no receiving surface is found for a given photon path). This input variable specifies the number of such occurrences before the run is aborted.
6. After emission and tracing from every |NINCR| surfaces have completed, the exchange matrix and the restart file is written to disk. Thus, it is possible to restart only from every NINCR states. Selecting a small value of NINCR will ensure that restart information is written frequently to the output file. For example, ten is about an appropriate number of times to write restart information. If NINCR is negative, the previous state is read from the restart and exchange matrix files. Restart may be of two types: (1) where the previous state is incomplete (i.e., the previous run was interrupted), or (2) the previous state is not converged. For negative NINCR, the restart file is rewritten after every |NINCR| surfaces have completed.

3.2.2 Card 2

Cols.	Format	Entry	Note
1-5	I5	Type of geometry (IGEOM).	1
		IGEOM = 0: axisymmetric	
		IGEOM = 1: Cartesian	
6-10	5I1	Output print control code (IPRINT(I)) as follows:	2
6	I1	IPRINT(1) - exchange fractions written to output file	
7	I1	IPRINT(2) - lost photons written to output file	
8	I1	IPRINT(3) - grid information (3D) or surface information (2D) information written to output file	
9	I1	IPRINT(4) - complete material property information written to output file	
10	I1	IPRINT(5) - addresses for IA array written to output file (3-D) or grid segment positions (2D)written to output file	
11-15 I5		Number of concats (NUMCAT).	3
16-20	I5	Data check code (IDATA).	
		IDATA = 0: Normal execution	
		IDATA = 1: Data check only	

21-25	I5	Number of *x*' emission points per surface (NDIVX).	
		(DEFAULT: NDIVX = 5)	
26-30	I5	Number of *y*' emission points per subsurface (NDIVY).	4
		(DEFAULT: NDIVY = 5)	
31-35	I5	Shading in geometry (NSHADE).	5
		NSHADE = (+1): Shading, distance algorithm	
		NSHADE = 0: No shading	
		NSHADE = (-1): Shading, Margolies grid shading algorithm	
		(DEFAULT: NSHADE = 0)	
36-40	I5	Trajectory control code (ITRACES).	6
		ITRACES > 0: Trajectory information written to disk file (*.trc* extension).	
		ITRACES = 0: Trajectory information not written to disk file.	
		(DEFAULT: ITRACES = 0)	
41-45	I5	Number of CPU’s (NCPU).	7
46-50	I5	Maximum number of photon convergence loops (NPLOOPS).	8
		(DEFAULT: NPLOOPS = 1)	

Notes:

1. IGEOM is used by the thermal balance codes as a consistency check.
2. If IPRINT(1) = 1, then exchange fractions are printed in the output file. If I PRINT(1) = 0, they are written only to the binary exchange matrix files. In general, the information is printed if IPRINT(i) = 1 and is not if IPRINT(i) = 0.
3. This option is not available in this version. This space is being retained in the input file for backward compatibility in format. If present, NUMCAT must be 0.
4. NDIVY = 1 for the 2-D code, and NDIVX divisions are used along the 2-D surface.
5. Any geometry (enclosure) that has a non-convex outer surface or any internal surface(s) has shading. Only an enclosure wherein each surface can fully “see” every other surface has no shading. Most geometries are convex, so it is suggested that, in general, NSHADE be set as non-zero.
6. The program dumps trajectory information to the *.trc* file if this option is set. This file is used to “view” the trajectories using the MPLOT program, useful in establishing physical intuition. If chosen, a copious amount of information is printed; the user is therefore advised to select this option only for very few photons per surface.
7. This option is no longer supported. This space is being retained in the input file for backward compatibility in format. If present, NCPU is ignored.
8. NPLOOPS is the maximum number of full surface photon convergence loops. In each full surface emission, NPHT photons are emitted at each subsurface. If convergence to the specified tolerance for the surface (see sections 3.4 and 3.7) is not attained within NPLOOPS full surface emissions, then a warning is printed to the screen, and execution continues with the next surface. If 0 is entered for NPLOOPS, convergence checking is assured by establishing a ridiculously large convergence tolerance. Convergence is then assessed via equation (2.2).

3.3 Scaling Factors

Cols.	Format	Entry	Note
1-10	E10.0	Scale for X (XSCALE). (DEFAULT: XSCALE = 1.0)	1
11-20	E10.0	Shift for X (XSHIFT). (DEFAULT: XSHIFT = 0.0)	1
21-30	E10.0	Scale for Y (YSCALE). (DEFAULT: YSCALE = 1.0)	1
31-40	E10.0	Shift for Y (YSHIFT). (DEFAULT: YSHIFT = 0.0)	1
41-50	E10.0	Scale for Z (ZSCALE). (DEFAULT: ZSCALE = 1.0)	1
51-60	E10.0	Shift for Z (ZSHIFT). (DEFAULT: ZSHIFT = 0.0)	1
61-70	E10.0	Increment of cone angle (DELT). (DEFAULT: DELT = 0.01 degrees)	2
71-80	E10.0	Tolerance for noncoplanar surfaces (SPLITOL) (DEFAULT: SPLITOL = 0.0001 degree)	3, 4

Notes:

1. These factors can be used when one needs to scale from one unit to another (i.e. meters to centimeters) or when one wishes to shift the coordinate system. Since there is no inherent length scale, scaling all axes equally has no effect; nor does shifting of any axis(es).

   Where:

   X = X * XSCALE + XSHIFT,

   Y = Y * YSCALE + YSHIFT, and

   Z = Z * ZSCALE + ZSHIFT

2. DELT = $\Delta\theta$: the increment used in numerically integrating the cumulative distribution function for emission versus cone angle. The range is: 1E-7 $\leq$ DELT $\leq$ 0.1.If a value outside this range is entered for DELT, DELT is set to the default value of 0.01. Note that very small values of DELT result in the consumption of excessive computer time.
3. SPLITOL is only used in the 3D code. In order to reduce the loss of photons, it is important that the four nodes of a quadrilateral surface be coplanar. The code checks this by dividing each quadrilateral element into two triangles. (By definition a triangle is coplanar.) It then calculates the dot product of both triangles. If the dot product differs from one by more than a specified tolerance (SPLITOL), it divides the surface into the two coplanar triangles created above for intersection and emission calculations. This division is transparent to the user. Split surfaces will be listed in the output file, but the final results will give exchange factors only for the surfaces originally defined. The range of acceptable values is: 1E-20 $\leq$ SPLITOL $\leq$ 0.01. If a positive value outside this range is entered for SPLITOL, SPLITOL is set to the default value of 0.0001.

4. Noncoplanar surfaces may cause excessive loss of photons, but if SPLITOL is not set low enough, not all noncoplanar surfaces will be split. In order to ensure that all noncoplanar surfaces are split, it is possible to force the code to split all quadrilateral surfaces into triangles. Again, the effect of this will be transparent to the user and will not affect the results (except for reducing the number of lost photons.) To turn this feature on, enter a negative value of SPLITOL such that $-0.01 \leq$ SPLITOL < 0.

3.4 Grid Dimensions (Shading)

Conditions: NSHADE < 0 (omit otherwise)

Cols.	Format	Entry	Note
1-5	I5	Number of grid cells in the X-direction (NGX).	1
6-10	I5	Number of grid cells in the Y-direction (NGY).	
11-15	I5	Number of grid cells in the Z-direction (NGZ).	2

Notes:

1. If NGX is input as a negative number, then the user must enter the grid coordinates as specified in section 3.11.
2. NGZ need not be entered for MONT2D.

3.5 Default Convergence Tolerance

Conditions: NPLOOPS > 0 (omit otherwise)

Cols.	Format	Entry	Note
1-10	E10.0	Default convergence tolerance for photon emissions (ERRDEF).	1
		(DEFAULT: ERRDEF = 1 x 10^{10})	2

Notes:

1. This is the default tolerance for convergence of the surface exchange fractions. This may be overridden for (a) particular surface(s) during surface input.
2. A large value, e.g.,1 x 10^{10} ensures that only one full surface emission loop will be performed.

3.6 Nodal Point Data

3.6.1 3-D Code

Cols.	Format	Entry	Note
1-5	I5	Node point number (N).	
6-10	I5	Increment in number of points to be generated (INC).	1
		(DEFAULT: INC = 0, no nodes generated)	
11-30	E20.0	X-coordinate: X(N).	
31-50	E20.0	Y-coordinate: Y(N).	
51-70	E20.0	Z-coordinate: Z(N).	

Note:

1. Nodal points are generated in increments of INC from *the previous node input to the current node*. The coordinates are obtained by linearly interpolating all coordinates between the ones input on the previous card and the present ones. Care must be taken such that there are an integer number of generated nodes between the present node number and the one input on the previous card. Note that, as the program calculates and fills these nodes, they should not be input elsewhere.

3.6.2 2-D Code

Cols.	Format	Entry	Note
1-5	I5	Node point number (N).	
6-10	5X	Skip	1
11-20	E10.0	X-coordinate (for axisymmetric, R-coordinate): X(N).	
21-30	E10.0	Y-coordinate (for axisymmetric, Z-coordinate): Y(N).	
31-40	E10.0	Z-coordinate: Z(N).	2
41-45	I5	Increment in number of points to be generated (INC).	1

Notes:

1. Nodal points are generated in increments of INC from *the previous node input to the current node*. The coordinates are obtained by linearly interpolating all coordinates between the present ones and the ones input on the previous card. Care must be taken such that there are an integer number of generated nodes between the present node number and the one input on the previous card. Note that, as the program calculates and fills these nodes, they should not be input elsewhere.
2. For MONT2D, this coordinate will be read but ignored.

3.7 Surface Data

Cols.	Format	Entry	Note
1-5	I5	Surface number (N).	
6-10	I5	Node N_1: NODES (1,N).	1
11-15	I5	Node N_2: NODES (2,N).	
16-20	I5	Node N_3: NODES (3,N).	2
21-25	I5	Node N_4: NODES (4,N).	2
26-30	5X	Skip.	
31-35	I5	Number of surfaces to be generated after current surface: NMISS.	
36-40	I5	Increment of generation: INC.	3
41-45	5X	Skip.	
46-50	I5	Surface material number: MATNUM(N).	
51-60	10X	Skip.	

61-65	I5	Number of photons (NPHT).	4
		(DEFAULT: NPHT = 0)	
66-70	I5	Photon increment (INCP).	5
71-80	E10.0	Convergence tolerance for surface: ERRMAX(N).	6
		(DEFAULT for surface: ERRMAX(N) = ERRDEF)	7,8

Notes:

1. The outward normal must be such that: for 2-D, as one proceeds from node N_1 to node N_2, the outward normal points to the right; for 3-D, the right-hand rule as explained in section 2.3 applies. Unpredictable and erroneous errors may result if this convention is not adhered to for all surfaces.
2. For MONT2D, N_3 and N_4 will be read but ignored.
3. NMISS surfaces are generated by successively incrementing surface numbers by 1 and all 4 node numbers by INC.
4. NPHT different from 0 overrides NPHTON on control Card 1, the number of photons per sub-surface division. If NPHT is negative, no photons will be emitted from the surface.
5. Similar to INC above. For each missing surface, i, (i = 1 to NMISS) generated, i*INCP extra photons will be added to it.
6. This is used as explained in section 2.8 to loop over full surface emissions until either the specified number of full surface emissions have occurred, or convergence to this tolerance, C_i defined in equation (2.3), is achieved, whichever comes first.
7. If no value is input (or a value of 0 is read), then this defaults to the global ERRDEF value input as described in section 3.5.
8. This value is read but ignored if NPLOOPS on Control Card 2 is 0, whence NPLOOPS is set to 1 and ERRMAX(N) is set to $1x10^{10}$.

3.8 Wavelength Band Data

CARDS 1 to (NBANDS-1)/8

Condition NBANDS > 1, (omit otherwise)

Cols.	Format	Entry	Note
1-10	E10.0	Wave breakpoint number 2 (micrometers).	1
11-20	E10.0	Wave breakpoint number 3 (micrometers).	1
21-30	E10.0	Wave breakpoint number 4 (micrometers).	1
		.	
		.	
		.	
71-80	E10.0	Wave breakpoint number 8 (micrometers).	2

Notes:

1. The first and last wave breakpoints are assumed to be 0.0 and ∞ (1×10^{10}) in micrometers and should not be input. Only breakpoints between 0 and ∞ should be input.
2. No more than eight breakpoints should be input per card. The total number of cards should be

(NBANDS - 1)/8.

3.9 Material Type Data

Material type data cards are input by band for each material in order. For example, if 3 materials are used in two bands, then all material type data are read in for material 1, band 1; then material 1 band 2; then material 2, band 1; etc. MN and IB are only used as checks to make sure the values are being entered in the right order. If the information is not entered in the correct order the program quits. All of these cards must be input before any material property curves are input (these cards control which material property curves are read).

Cols.	Format	Entry	Note
1-5	I5	Material type: MTYPE(N,IB): domain - [-4,2]	1
			2,3
			4,5
6-15	E10.0	Outgoing cone angle θ_o for beam radiation: THSET(MN,IB)	1
16-25	E10.0	Outgoing surface azimuth angle ϕ_o for beam radiation: PHISET(MN,IB)	1
26-35	E10.0	R dependance of diffuse reflectance: RDIFFR(MN,IB)	4,5
		(DEFAULT: RDIFFR = 1)	6
36-45	E10.0	R dependance of diffuse transmittance: RDIFFT(MN,IB)	4,5
		(DEFAULT: RDIFFT = 1)	6
46-50	I5	Material number (MN)	4,5
51-55	I5	Wavelength band number (IB)	4,5

Notes:

1. THSET and PHISET are ignored if MTYPE is other than 1.

2.If MTYPE(N) = 2, then emission occurs according to the function *fcn* in *subroutine getang*.

3. Material property curves are required for all material types except -3 and -4.

4. If the values of MN and IB are 0 (or blank) for the first band of a material then RDIFFR and RDIFFT are set to 1 (standard diffuse re-emission) and the material type data values are used for all bands in that material and no other material type data cards are read in for that material. This has been done for compatibility with earlier versions which do not support r dependance or multiple material types.

5. If MN is not 0 but IB is for the first band of the material then it is assumed that the material type data are constant for this material for all bands and no other material type data cards are read in for this material.

6. If either RDIFFR or RDIFFT = 0, then it will be set to 1 (standard diffuse re-emission.) If either RDIFFR or RDIFFT are < 0, then it will be set to 0 (isotropic re-emission.)

3.10 Material Property Curves

Material property curves are input by band for each material, except for materials of type -3 and -4, where no curves are input. For example, if 3 materials are used in two bands, then all curves are read in for material 1, band 1; then material 1 band 2; then material 2, band 1; etc. MN and IB are only used as checks to make sure the values are being entered in the right order. If the information is not entered in the correct order the program quits.

3.10.1 Specular Transmittance

Conditions: MTYPE = 2,1,0, or -2 (omit otherwise)

CARD 1

Cols.	Format	Entry	Note
1-16	2A8	Name of curve (e.g., specular trans): (CN1, CN2)	
17-21	I5	Material number (MN)	1
22-26	I5	Wavelength band number (IB)	1
27-31	I5	Curve number (NJ)	1, 2
32-36	I5	Number of points to be input for this material (NP)	3

CARDS 2 to NP+1

Cols.	Format	Entry	Note
1-10	E10.0	Cone angle - θ	3
11-20	E10.0	Specular transmittance: $\rho_s(\theta)$	

Notes:

1. These values are used to check that the data are input properly by material number, band number, and curve number.
2. A curve number of 1 must be input for specular transmittance.
3. NP number of cards must be input for the curve.

3.10.2 Diffuse Transmittance

Conditions: MTYPE = -1 or -2 (omit otherwise)

CARD 1

Cols.	Format	Entry	Note
1-16	2A8	Name of curve (e.g., diffuse trans): (CN1, CN2)	
17-21	I5	Material number (MN)	1
22-26	I5	Wavelength band number (IB)	1
27-31	I5	Curve number (NJ)	1, 2
32-36	I5	Number of points to be input for this material (NP)	3

CARDS 2 to NP+1

Cols.	Format	Entry	Note
1-10	E10.0	Cone angle - θ	3
11-20	E10.0	Diffuse transmittance: $\tau_d(\theta)$	

Notes:

1. These values are used to check that the data are input properly by material number, band number, and curve number.
2. For diffuse transmittance, curve number = 1 for MTYPE = -1 and curve number = 2 for MTYPE

= -2.

3. NP number of cards must be input for the curve.

3.10.3 Specular Reflectance

Conditions: MTYPE ≠ -3 or -4 (omit otherwise)

CARD 1

Cols.	Format	Entry	Note
1-16	2A8	Name of curve (i.e. specular reflectance): (CN1, CN2)	
17-21	I5	Material Number (MN)	1
22-26	I5	Wavelength band number (IB)	1
27-31	I5	Curve number (NJ)	1, 2
32-36	I5	Number of points to be input for this material (NP)	3

CARDS 2 TO NP+1

Cols.	Format	Entry	Note
1-10	E10.0	Cone angle - θ	3
11-20	E10.0	Specular Reflectance: $\rho_s(\theta)$	

Notes:

1. These values are used to check that the data are input properly by material number, band number, and curve number.
2. For specular reflectance, curve number = 2 for MTYPE = 2 to -1 and curve number = 3 for MTYPE = -2.
3. NP number of cards must be input for the curve.

3.10.4 Diffuse Reflectance

Conditions: MTYPE ≠ -3 or -4 (omit otherwise)

CARD 1

Cols.	Format	Entry	Note
1-16	2A8	Name of curve (e.g. diffuse reflectance): (CN1, CN2)	
17-21	I5	Material number (MN)	1
22-26	I5	Wavelength band number (IB)	1
27-31	I5	Curve number (NJ)	1, 2
32-36	I5	Number of points to be input for this material (NP)	3

CARDS 2 TO NP+1

Cols.	Format	Entry	Note
1-10	E10.0	Cone angle - θ	3
11-20	E10.0	Diffuse reflectance: $\rho_d(\theta)$	

Notes:

1. These values are used to check that the data are input properly by material number, band number, and curve number.
2. For diffuse reflectance, curve number = 3 for MTYPE = 2 to -1 and curve number = 4 for MTYPE = -2.
3. NP number of cards must be input for the curve.

3.11 User Grid Input

Conditions: NSHADE < 0 and NGX < 0

Cols.	Format	Entry	Note
1-80	8E10.0	X-grid coordinates XG(N)	1
1-80	8E10.0	Y-grid coordinates YG(N)	2
1-80	8E10.0	Z-grid coordinates ZG(N)	3

Notes:

1. For N = 1 to NGX+1. More than one card may be required.
2. For N = 1 to NGY+1. More than one card may be required.
3. For N = 1 to NGZ+1. More than one card may be required. For MONT2D, these cards are not read.

4. EXCHANGE MATRIX OUTPUT FILES

4.1 General Description

The exchange matrix files contain all of the information necessary to solve for the net radiative exchange as described in Chapter 1.

Familied files, identified by the names entered by the user, are generated for the absorption exchange matrices. The header of each file contains control information and the surface areas. This is followed by the surface emittances and the exchange number matrix for each band, ordered from lowest band number to highest.

4.2 Output Format for Exchange Matrix Files

The output format for both files is identical. To be compatible with TACOxD, these files may need to be processed with the program SMOOTH [Burns and Loehrke, 1992], which smooths the full matrix of exchange numbers into an upper triangle of numbers which obey reciprocity. All (un-smoothed and smoothed) files of exchange numbers are constant record length, direct access, binary files. All quantities are 32 bit, with integers stored as binary in powers of 2, and reals stored in IEEE standard 32-bit floating point format.

4.3 Header Cards

1 Record of Length NSURF (padded with zeroes)

Format	Entry	Note
Integer*4	Geometry code (NDIM)	1
Integer*4	Number of surfaces (NSURF)	
Integer*4	Factor code (IFACT)	2
Integer*4	Number of wavelength bands (NBANDS)	
Integer*4	Number of materials (NUMMAT)	

Notes:

1. NDIM is 1 for 2-D axisymmetric geometries, 2 for 2-D Cartesian geometries, and 3 for 3-D geometries.
2. IFACT is set to 2, and indicates that the file contains exchange numbers as opposed to view factors.

4.4 Surface Areas

1 Record of Length NSURF

Format	Entry
Real*4	Surface areas from 1 to NSURF

4.5 Data for Each Wavelength Band

(Repeated for each band, from k = 1 to NBANDS)

4.5.1 Surface Emittances

For Every Wavelength Bands

1 Record of Length NSURF

Format	Entry	Note
Real*4	Average surface emittances, $\bar{\varepsilon}_i^K$, from 1 to NSURF	1, 2

Notes:

1. The surface emittances are averaged over direction (cone angle), i.e., the values are hemispherical emittances.
2. For each band, the average surface emittances are written, followed by the exchange number matrix.

4.5.2 Photon Number Matrix

NSURF Records Each of Length NSURF

Numbers of Absorbed Photons by surface *i* for all *j*

Format	Entry	Note
Integer*4	Numbers of absorbed photons	1, 2

Notes:

1. One matrix, preceded by the average surface emittances, is written for each wavelength band.
2. The data are presented by row *i* for all columns *j*. That is, the first row of numbers $N_{1,(1 \rightarrow NSURF)}$ are presented, then for row 2, etc.
3. These files contain the full exchange matrices. The program SMOOTH [Burns and Loehrke, 1992] may be used to process these files into an upper triangle (including the diagonal) of numbers of photons which have been smoothed to obey reciprocity.

5. SAMPLE INPUT FILES

5.1 General Description

This section contains three sample input files to demonstrate the MONTE input format. The first is a 2-D triangle with diffusely reflecting inner surfaces, written for MONT2D. The second is a 3-D shoebox (rectangular prism) with specularly reflecting inner surfaces, written for MONT3D. The third input file, *frust.in*, illustrates the axisymmetric features of the 2-D program. Close examination of these files should assist the user in understanding how to generate a file.

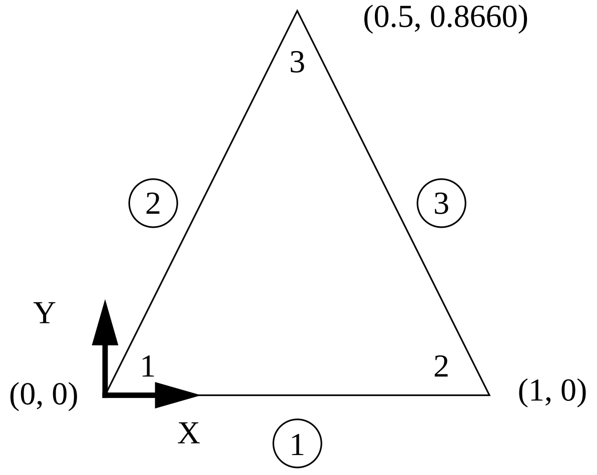


Figure 5.1 2-D Triangle of File ‘‘tri.in’’

5.2 2-D Prismatic Triangle

Figure 5.1 shows the triangle with its nodal coordinates, nodal numbering, and surface numbering. The figure actually represents a triangular prism extending infinitely far into (and out of) the paper. The input file “tri.in” is given below. Note the sequence of node point numbering, following the right-hand rule explained in section 2.3. Surface numbers are contained within circles. The inner surface of the prism is composed of a material with a diffuse reflectance of 0.2, independent of incident angle.

```
2-D DIFFUSE TRIANGLE TEST PROBLEM
&NDIM      NUMEL   NBANDS     NREFS     NLOST        INSEED
&    NUMNP     NUMMAT     NPHOTS     NWARNS      NINCR
    2    3    3    1    1  100  100  100   10    1         0
&IGEOM   NUMCAT      NDIVX     NSHADE      NCPU
&       IPRNT    IDATA      NDIVY    ITRACES    NPLOOPS
    110000    0    0  200    0    1    0    0   20
&   XSCALE    XSHIFT    YSCALE    YSHIFT    ZSCALE    ZSHIFT      DELT
        1.        0.        1.        0.       0.0        0.      0.01
& GRID NUMBERS (READ IF NSHADE < 0)
& NGX  NGY  NGZ
&   5    5    5
& ERRDEF (READ IF NPLOOOPS >0)
```

```
     1.e-3
& NODES (2D FORMAT)
&   N            X        Y          Z   NINC
    1           0.0       0.0       0.0    0
    2           1.0       0.0       0.0    0
    3           0.5    0.8660       0.0    0
& ELEMENTS
&   N   N1   N2   N3   N4       NMISS INC        MN            NPHT INCP  ERRMAX
    1    2    1    0    0          0    0         1              0    0     0.00
    2    1    3    0    0          0    0         1              0    0     0.00
    3    3    2    0    0          0    0         1              0    0     0.00
& WAVELENGTH BAND BREAKPOINTS (READ IF NBANDS > 1)
&      BP1       BP2       BP3       BP4       BP5       BP6       BP7       BP8
&      5.0
& MATERIALS
& MTYPE  THTSET    PHISET    RDIFFR    RDIFFT   MN   IB
    0       0.0       0.0       1.0       1.0    1    1
&         NAME      MN   IB   NJ   NP
TRANS SURF1         1    1    1    3
&     ANGLE    VALUE
       0.0       0.0
      45.0       0.0
      90.0       0.0
RHO S SURF1         1    1    2    3
&     ANGLE    VALUE
       0.0       0.0
      45.0       0.0
      90.0       0.0
RHO D SURF1         1    1    3    3
&     ANGLE    VALUE
       0.0       0.2
      45.0       0.2
      90.0       0.2
& USER DEFINED GRID (READ IF NGX < 0)
& USER GRID X
&        0       0.1       0.2       0.5       0.8      1.01
& USER GRID Y
&    -0.01      0.05       0.1       0.9      0.95      1.01
```

5.3 3-D Box

Figure 5.2 shows a 3-D geometry for analysis by MONT3D. Comparison of the picture with the input file “box.in” shown below illustrates the right-hand rule for 3-D surfaces. The reflectance of the inner surfaces is mixed specular-diffuse in this case, and varies with incident angle as shown in Figure 5.3.

```
3-D BOX TEST PROBLEM
&NDIM     NUMEL    NBANDS     NREFS     NLOST         INSEED
&    NUMNP    NUMMAT    NPHOTS    NWARNS     NINCR
    3    8    6    1    1   10  100  100   10    1         0
&IGEOM   NUMCAT     NDIVX    NSHADE     NCPU
&    IPRNT     IDATA     NDIVY  ITRACES   NPLOOPS
    11000     0     0    10    10     1     0     0 10
```

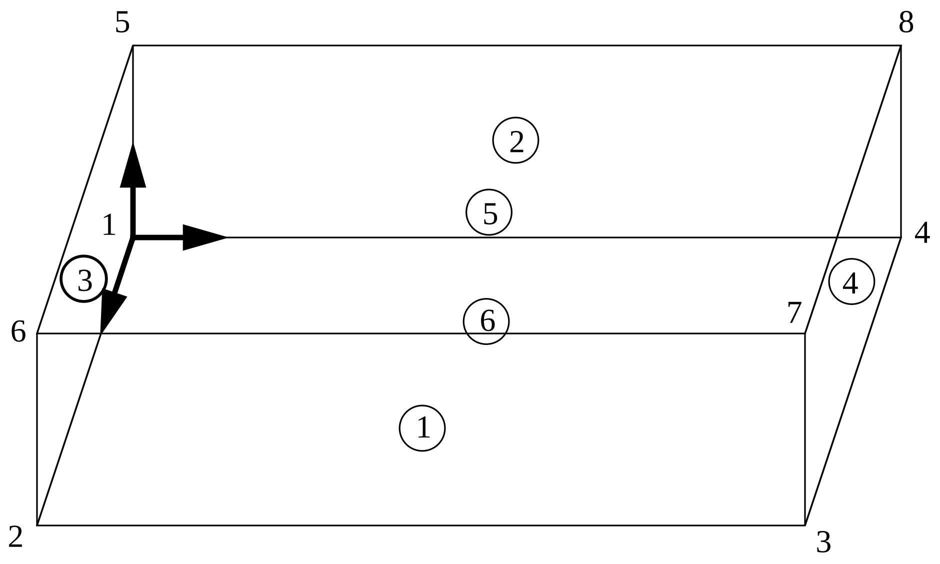


Figure 5.2 3-D Geometry of File “box.in”

```
&   XSCALE    XSHIFT    YSCALE    YSHIFT    ZSCALE    ZSHIFT      DELT   SPLITOL
       1.0       0.0       1.0       0.0       0.0       0.0      0.01    0.0001
& GRID NUMBERS (READ IF NSHADE < 0)
& NGX  NGY  NGZ
&   5    5    5
& ERRDEF (READ IF NPLOOOPS >0)
    1.0e-2
& NODES (3D FORMAT)
&   N  INC                  X                   Y                   Z
    1    0                0.0                 0.0                 0.0
    2    0               20.0                 0.0                 0.0
    3    0               20.0                30.0                 0.0
    4    0                0.0                30.0                 0.0
    5    0                0.0                 0.0                10.0
    6    0               20.0                 0.0                10.0
    7    0               20.0                30.0                10.0
    8    0                0.0                30.0                10.0
& ELEMENTS
&   N   N1   N2   N3   N4      NMISS INC         MN           NPHT INCP    ERRMAX
    1    1    2    3    4          0    0          1              0    0      0.00
    2    5    8    7    6          0    0          1              0    0      0.00
    3    1    5    6    2          0    0          1              0    0      0.00
    4    3    7    8    4          0    0          1              0    0      0.00
    5    1    4    8    5          0    0          1              0    0      0.00
    6    2    6    7    3          0    0          1              0    0      0.00
& WAVELENGTH BAND BREAKPOINTS (READ IF NBANDS > 1)
&      BP1       BP2       BP3       BP4       BP5       BP6       BP7       BP8
&      5.0
& MATERIALS
& MTYPE  THTSET    PHISET    RDIFFR    RDIFFT   MN   IB
    0       0.0       0.0       1.0       1.0    1    1
&        NAME      MN   IB   NJ   NP
```

```
TRANSMISSIVITY      1    1    1    3
&     ANGLE    VALUE
       0.0       0.0
      45.0       0.0
      90.0       0.0
RHO SPECULAR        1    1    2    6
&     ANGLE    VALUE
       0.0      0.40
      30.0      0.38
      50.0      0.35
      70.0      0.20
      80.0      0.20
      90.0      0.95
RHO DIFFUSE         1    1    3    3
&     ANGLE    VALUE
       0.0      0.00
      45.0      0.00
      90.0      0.00
& USER DEFINED GRID (READ IF NGX < 0)
& USER X GRID
&    -0.01      0.01         1         5     16.0      20.01
& USER Y GRID
&    -0.01      0.01         1         3        9      20.01
& USER Z GRID
&    -0.01      0.01         1      1.01        7      20.01
```

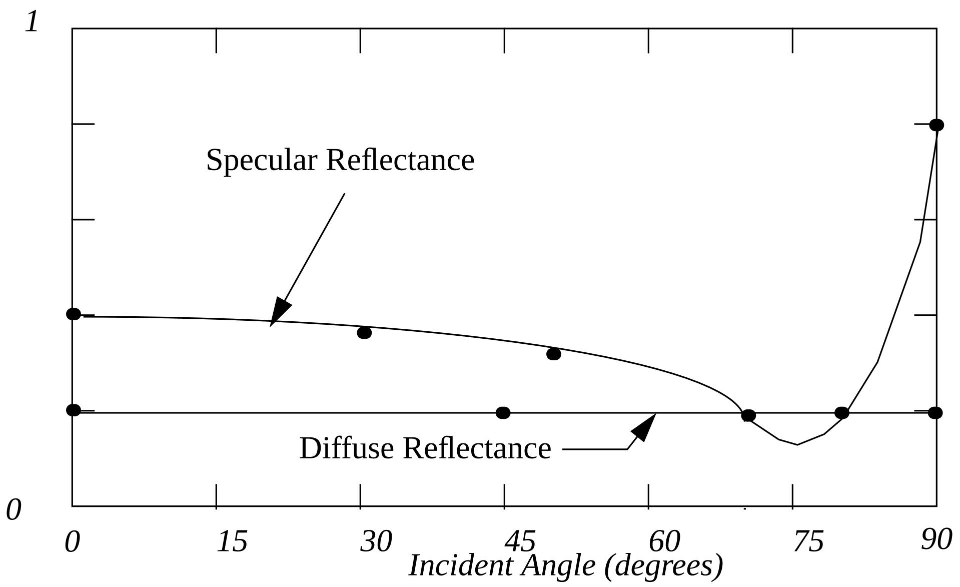


Figure 5.3 Material Property Curves for File ‘‘box.in’’

5.4   2-D Axisymmetric Frustrum

The file “frust.in” below is for the axisymmetric frustrum shown in Figure 5.4. Note that the geometry code is now 0, indicating that the axisymmetric case is being simulated. Note also that the surfaces do not close on themselves, but rather close on the axis of revolution (R = 0). Finally, the material properties are mixed specular-diffuse and vary with incident angle.

```
2-D DIFFUSE FRUSTRUM
&NDIM     NUMEL   NBANDS      NREFS     NLOST         INSEED
&    NUMNP    NUMMAT    NPHOTS    NWARNS     NINCR
    2    4    3    1    1  100  100  100   10    0         0
&IGEOM   NUMCAT     NDIVX    NSHADE      NCPU
&    IPRNT     IDATA     NDIVY   ITRACES   NPLOOPS
    010000    0    0   10    0    1    0    0   10
&   XSCALE    XSHIFT    YSCALE    YSHIFT    ZSCALE    ZSHIFT      DELT
        1.        0.        1.        0.       0.0        0.      0.01
& GRID NUMBERS (READ IF NSHADE < 0)
& NGX  NGY  NGZ
&   5    5    5
& ERRDEF (READ IF NPLOOOPS > 0)
     1.e-2
& NODES (2D FORMAT)
&   N            X         Y         Z    INC
    1          0.0       0.0       0.0    0
    2          1.0       0.0       0.0    0
    3          0.5       1.0       0.0    0
    4          0.0       1.0       0.0    0
& ELEMENTS
&   N   N1   N2   N3   N4       NMISS INC        MN           NPHT INCP  ERRMAX
    1    2    1    0    0         0    0         1             0    0      0.00
    2    3    2    0    0         0    0         1             0    0      0.00
    3    4    3    0    0         0    0         1             0    0      0.00
& WAVELENGTH BAND BREAKPOINTS (READ IF NBANDS > 1)
&      BP1       BP2       BP3       BP4       BP5       BP6       BP7       BP8
&      5.0
& MATERIALS
& MTYPE  THTSET    PHISET    RDIFFR    RDIFFT   MN   IB
    0       0.0       0.0       1.0       1.0    1    1
&        NAME      MN   IB   NJ   NP
TRANSMITTANCE       1    1    1    3
&     ANGLE    VALUE
        0.0      0.0
       45.0      0.0
       90.0      0.0
RHO SPECULAR        1    1    2    3
&     ANGLE    VALUE
        0.0      0.0
       45.0      0.0
       90.0      0.0
RHO DIFFUSE         1    1    3    3
&     ANGLE    VALUE
        0.0     0.20
       45.0     0.20
       90.0     0.20
& USER DEFINED GRID (READ IF NGX < 0)
& USER GRID R
&        0       0.1       0.2       0.5       0.8      1.01
& USER GRID Z
&    -0.01      0.05       0.1       0.9      0.95      1.01
```

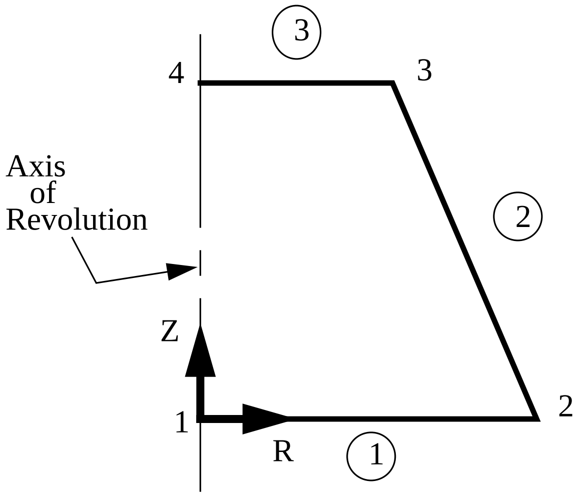


Figure 5.4 Axisymmetric Frustrum

6. PROGRAM EXECUTION

6.1 Execution of File “tri.in”

The following details the execution of the 2-D code with the file “tri.in.” The output below in section 6.2 is output to the screen, and has been captured using the Unix utility *script*. The Unix prompt is “pburns n%,” where n is the command number. All input from the user is shown bold and italicized. The script file created here is called “xxx.”

The executable file here is called *mont2d* (note that Unix is case sensitive), and is invoked by typing its name - shown at the first prompt after establishing the script - “pburns 1%.” All subsequent output is from the program MONT2D. The user is asked to type the prefix of the input file, which MUST have the extension “in.” In this case, the file is “tri.in,” and the user enters “tri.” The template of 8 a’s indicates that the user must type a prefix containing at most 8 alphanumeric characters. If the program cannot find the input file, or if some of the output files already exist, these errors are trapped by the code, and a diagnostic is printed (this hopefully prevents files being overwritten). Additional detail on the files used by the codes exists in Chapter 7.

The program proceeds with reading of the input file. The heading is printed, along with the version number (here, 2.6), the date of last modification (here, 02-03-94), and the system for which the code was compiled (here, Unix). Next, the control portion of the input is read. This establishes the numbers of nodes, surfaces (sometimes referred to as elements or surface elements), wavelength bands, materials and grids which are to be read by succeeding subroutines. Then, the input subroutines are executed one-by-one. As each subroutine is entered, the name of the subroutine is printed to the output file. This assists in debugging input files. For example, if the program “crashes” after the message “in nodin2” is printed, there is almost certainly a problem with the nodal input (or possibly, the card images are being read out of order). The successful completion of the entire input phase is then indicated by the message, “i n p u t   p h a s e   c o m p l e t e.”

The solution phase occurs next. After the specified number of full surface emission loops for each surface is completed, a message is printed indicating whether the exchange fractions have achieved the specified convergence tolerance. In addition, the wavelength band, surface number, total number of emitted photons (summed over all full surface emissions), and the calculated error are printed. Here, none of the surfaces converge to the stringent error tolerance of 0.001, but all come close. Finally, the overall program statistics are printed, including the total CPU time for the run (in seconds), the total number of photons emitted (and traced to absorption), and the performance indication in number of photons per CPU second.

Finally, upon completion of the run, the Unix prompt is issued. To exit the script, ensuring that the script file is saved, *exit* is typed. Then, the script file is saved. It may then be viewed using the Unix *more* command (i.e., *more xxx*), or using the Unix text editors (e.g., *vi xxx*). Additionally, the output file “tri.out” is available; the plot file “tri.plt” can be used with the MPLOT program to view the geometry and material properties, etc.

Section 6.3 is the screen output generated by a restart run of tri.in. The only modification to file “tri.in” was to change NINCR from 1 for the normal run to -1 for the restart run. Also note that before the restart run, the output files tri.out and tri01.out must be deleted or moved. The restart run will need to create files with the same names. As a safety feature to prevent the accidental loss of previous work, MONT2D will halt the run before these files are overwritten. Convergence was achieved in the restart run.

6.2 Output During Execution of File “tri.in”

```
pburns 14% script xxx
Script started on Fri Feb 4 10:06:36 1994
pburns 1% mont2d
  ****** mont2d ******
  enter input file prefix name -
             Note: input file MUST have extension .in
enter up to 8 alphanumeric characters according to the
template provided below
aaaaaaaa
tri
 2-D DIFFUSE TRIAGLE TEST PROBLEM               ver 2.6 02-03-94 unix

 in nodin2
 in elmin2
 in wavin
 in matin
 in curvin
 in cumdis
 in order
 in graf



         i n p u t   p h a s e   c o m p l e t e





                  band  surf.  iter.   npht.   error
 not converged -     1      1     20    400000   0.1334E-02
 restart file written
 not converged -     1      2     20    400000   0.1335E-02
 restart file written
 not converged -     1      3     20    400000   0.1335E-02
 restart file written




     normal termination




 total time charged for run =  0.44403E+03 (secs)
 total no. of photons emitted =  0.12000E+07
 photons per CPU second =  0.27095E+04 (photons/sec)
```

```
pburns 2% exit
pburns 3%
script done on Fri Feb 4 10:19:21 1994
```

6.3 Output During Restart Execution of File “tri.in”

```
pburns 15% script yyy
Script started on Fri Feb 4 13:33:10 1994
pburns 1% mont2d
  ****** mont2d ******
  enter input file prefix name -
             Note: input file MUST have extension .in
enter up to 8 alphanumeric characters according to the
template provided below
aaaaaaaa
tri
 2-D DIFFUSE TRIANGLE TEST PROBLEM                ver 2.6   02-03-94 unix

 in nodin2
 in elmin2
 in wavin
 in matin
 in curvin
 in cumdis
 in order
 in graf



         i n p u t   p h a s e   c o m p l e t e






                  band  surf.  iter.   npht.   error
     converged -     1      1     16    720000   0.9945E-03
     restart file written
     converged -     1      2     16    720000   0.9944E-03
     restart file written
     converged -     1      3     16    720000   0.9945E-03
     restart file written




     normal termination
 total time charged for run =  0.37175E+03 (secs)
 total no. of photons emitted =  0.21600E+07
 photons per CPU second =  0.58274E+04 (photons/sec)
ppburns 2% exit
pburns 2%
script done on Fri Feb 04 13:50:55 1994
```

6.4 Execution of Other Input Files

Execution of other input files produces similar output. The user interfaces of both the 2-D and 3-D programs have been designed to produce similar output.

7. IMPLEMENTATION

7.1 Compiling

7.1.1 Normal Execution

On Unix systems, the following shell file (delivered as file f*77m*) can be executed (it must be an executable file - see the Unix command *chmod*) to compile and link the code:

```
f77 -O -c -r8 m20s.f
f77 -O -c -r8 m21s.f
f77 -O -c -r8 m22s.f
f77 -o mont2d m20s.o m21s.o m22s.o
```

The first 3 lines generate the relocatable binary files for the: (1) control, (2) input and (3) solution phase routines, respectively. The final line invokes the linker, generating the absolute binary (executable) file *mont2d*. The source code files: *m20s.f, m21s.f* and *m22s.f* may be compiled separately. Additionally, linking may be done separately. The above is valid also for the 3-D code, provided that the names begin with *m3* instead of *m2*. Alternately the UNIX make utility can be used with the provided Makefile.

Finally, if a debug or a profiling run is to be done, the compile lines should all include the option -g (or -pg), which specifies that a symbol table be generated, and that source code line numbers be interspersed within the executable file. This option generates a much larger, slower executable, and should be exercised only when debugging. The resultant source code may be run under the control of the Unix debugging tool *dbx* (or *dbxtoo*l on Sun architectures).

7.1.2 Debug Execution

All of the above remains the same, except that the -xld option is added during the compile. This option causes source code lines beginning with a *d* in column 1 also to be compiled. Viz.:

```
f77 -O -c -xld -r8 m20s.f
f77 -O -c -xld -r8 m21s.f
f77 -O -c -xld -r8 m22s.f
f77 -o mont2d m20s.o m21s.o m22s.o
```

7.2 Common Blocks, Parameter Statements and the Size of Arrays

MONT2D has been changed to take more advantage of common blocks. This makes the code more robust and easier to edit. Unfortunately, this also requires limits to be set on the size of arrays. The sizes of various arrays are set in a parameter statement in the various subroutines. These parameters limit the number of various objects that can be used use such as number of elements, number of wavelength bands and number of materials. To increase these numbers, the parameters must be changed in EVERY subroutine in which they appear (and the main program) and the program must be recompiled. Note that the larger the arrays are, the more memory the program uses. If the arrays become too large, the program may become too large to be stored completely in available physical, memory and the program speed will severely degrade as the program is swapped to and from the disk.

The current parameters used to size arrays and there values are:

iblk	100
(maximum number of surfaces done before writing to a restart file)	
ibnd (maximum number of wavelength bands)	5
igrd (maximum number of segments in all grid cells)	7999
imat (maximum number of materials)	30
inod (maximum number of nodes):	5000
isrf (maximum number of surfaces):	3500

7.3 Distribution Files

The following files are included:

Source Code Files:
m20s.f, *m21s.f* and *m22s.f* for the 2D code, and *m30s.f*, *m31s.f* and *m32s.f* for the 3D code.
Input Files:
tri.in, *box.in* and *frust.in*.
Output Files:
all files generated by *tri.in*, *box.in* and *frust.in*.
Shell Files (for compiling and linking):
f77m.
Makefiles
Makefile containing instructions for the UNIX command make to allow automatic compilation.
Release Notes File:
The *README* files contain descriptions of the release. The actual release notes of the changes made are rnotes2d.txt and rnotes3d.txt in text format and rnotes2d.ps and rnotes3d.ps in Postscript format.
Users' Manual Files:
manual.ps (in printable PostScript format - can only be printed on a PostScript printer).
Test Files:
Various input files that can be used to test the various features have been included in a directory test_files for each program. A description of these input files can be found in the file testfiles.txt.

7.4 Problems

If problems with the code are encountered, one of the authors should be contacted.

7.5 Distribution

The codes are distributed by Jim Maltby of the Methods Development Group at Lawrence Livermore National Laboratory. (Ph: (510) 422-2688, e-mail: maltby1@llnl.gov)

7.6 Precision

The example problems are compiled and run using 64-bit floating point precision. For larger geometries, it is desirable to use 64-bit precision. This is especially so for the 3-D code, as the specification of co-planar surfaces is particularly susceptible to precision errors. To compile in 64 bits, the flag -r8 is used on Sun architectures. For smaller problems (geometries), the user may wish to compile and execute in 32-bit precision as execution is nearly twice as fast on most machines. *Caveat emptor*!

The exchange fraction files are always written in 32-bit format (note that the only real numbers written are the surface areas and emittances), as this saves storage space. When the codes are compiled in 64-bit format, the subroutines *rdabsf64* and *wrabsf64* are used. When executing in 32-bit mode, the subroutines *rdabsf* and *wrabsf* are used instead. This change must be done manually in the proper part of each code.

8. FILES AND FILE USAGE

This chapter briefly describes the file usage. For specificity, a prefix of *fn* is used. Table 8.1 gives the files used (either pre-existing or generated during execution) by MONTE.

Table 8.1: MONTE Files

Unit	Name	Function
1	fn.scr	Input file with comment cards “stripped away.” File is generated by code and deleted before completion of run. This is the file actually read during the input phase.
2	fn.rst	Restart file in binary format. Contains numbers of photons emitted per surface, and current seed for the random number generator.
3	fn.plt	Plot file, to be used with the program MPLOT [Shivaswamy, Nagesh and Burns, 1994]. Contains geometrical and material property information.
4	fn.lst	Contains lost photon trajectories, if any. For large geometries and many photon emissions, there is potential that many photons may be lost. This information is written to this separate file, so as not to clutter the output file.
5	stdin	Standard input (keyboard)
6	stdout	Standard output (screen).
7	fn.in	Input file with comment cards, in a format described in section 3. This file is used to generate the file of unit 1.
8	fnxx	Familied absorption exchange factors. These are stored in binary format as described in section 4. Note: xx is the number of the familied output file.
9	fn.trc	Trajectory file (written if ITRACES $\neq 0$, see section 3.2.2). To be used with MPLOT [Shivaswamy, Nagesh and Burns, 1994] to plot particle trajectories.
10	fnxx.out	Familied output file. Contains echo of all input and other information as determined by IPRINT. (see section 3.2.2)
12	fn.lks	Leaks file. To be used with MPLOT [Shivaswamy, Nagesh and Burns, 1994] to identify potential leaks. (3-D code only)

Files for unit numbers 1, 3-7, 9,10 and 12 are ASCII files, and may be read, typed, and printed. Files for unit numbers 2, and 8 are binary files, and require to be read by other programs. For example, unit 2 is used only by the MONT2D and MONT3D codes. Unit 8 is written by MONT2D and MONT3D, and read by the exchange factor smoothing program SMOOTH [Dolaghan et al., 1992]. The UNIX utility *od* (octal dump) may be used to examine binary files (do a *man* on *od* for instructions).

8.1 Specifying File Names

MONTE must have the names specified for all of the files shown in Table 8.1. This section explains the naming conventions and methods of specifying these file names.

There are two methods of specifying the file names; default and command line. If no name(s) are specified on the command line, the code will query the user via the console for a base file name (*fn*). This base name is used as the prefix for all files in the run (see above). The command line method of specifying the file names allows the user more flexibility in defining file names. Each file name can be specified independently according to the conventions in Table 8.2.

Table 8.2: Command Line File Control

File name to be specified	Preceded on command line by
Restart file name	-r, -R, r=, or R=
Plot file name	-p, -P, p=, or P=
Lost photon trajectory file name	-m, -M, m=, or M=
Input file name	-i, -I, i=, or I=
Absorption exchange factor file name	-e, -E, e=, or E=
Trajectory file name	-t, -T, t=, or T=
Output file name	-o, -O, o=, or O=
Leaks file name	-l, -L, l=, or L=
Family file name	-f, -F, f=, or F=

Several conventions bear emphasizing. If even one file name is specified on the command line, the code will not query the user for a name. As a minimum the names of the input file, output file and exchange factor file must be specified independently on the command line with the other files deriving their base name from the input file name. If the f option is used, any file not explicitly specified will assume the naming convention indicated in Table 8.1. Those file names explicitly specified will override this default. In all cases, serial numbers will be appended to the familied absorption exchange factor files and output files. The maximum length of any base file name is 8 characters. In the case of familied files, the last two characters of the name are used for the serial number.

To help clarify the command line conventions, the following examples and explanations are offered.

Example 1: %mont3d -i alakazam -Okaboom e= ardvark.w M=toasted

MONTE will expect the input file named *alakazam* to exist. MONTE will create the following files; the restart file named *alakazam.rst,* the plot file named *alakazam.plt,* the lost photon trajectory file named *toasted,* the absorption exchange factor files named *ardvar, ardvar01, advar02* ... up to the number required, the trajectory file (if required) named *alakazam.trc*, the MONTE output files named *kaboom, kaboom01, kaboom02* .. up to the number required and the leaks file (if required) named *alakazam.lks*.

Example 2: %mont3d -f calendar

MONTE will expect the input file *calender.in* to exist MONTE will create the following files; the restart file named *calender.rst*, the plot file named *calander.plt*, the lost photon trajectory file named *calender.lst*, the absorption exchange factor files named *calend*, *calend01*, *calend02* ... up to the number required, the trajectory file (if required) named *calender.trc*, the MONTE output files named *calend.out*, *calend01.out*, *calend02.out* .. up to the number required and the leaks file (if required) named *calender.lks*,

Example 3: %mont3d -f tri -e mi5run

MONTE will expect the input file *tri.in* to exist MONTE will create the following files; the restart file named *tri.rst*, the plot file named *tri.plt*, the lost photon trajectory file named *tri.lst*, the absorption exchange factor files named *mi5run, mi5run01, mi5run02* ... up to the number required, the trajectory file (if required) named *tri.trc*, the MONTE output files named *tri.out*, *tri01.out*, *tri02.out* .. up to the number required and the leaks file (if required) named *tri.lks*,

9. BATCH EXECUTION USING SCRIPTS

This section describes the use of a UNIX shell script to submit multiple MONTE runs to be executed sequentially. This has the advantage of running only one job at a time, thereby avoiding the overhead involved with swapping jobs in memory in and out of the CPU (our tests have shown that the overhead under UNIX in so doing is prohibitively large). The shell script in Figure 9.1 is an example which runs MONT2D three times for three separate problems (input files). In this example line one executes mont2d and redirects standard output to the file *run1*. The second line causes the script to pause until line 1 has finished. When the first run has completed the process continues with line 3, etc.

```
mont2d -f tc1 > run1
wait
mont2d -f tc2 > run2
wait
mont2d -f tc3 > run3
```

Figure 9.1 Script “submit”

The script file can be created with any ASCII editor and given any valid UNIX name (here the file is named *submit*). The files referred to in the script (here *tc1, run1*, etc.) can also have any valid UNIX name. The files *tc1, tc2,* and *tc3* must be the MONTE input files (without the .in extension) to be used for the run.

Once the script is created it must have execute permission before it can be run. This is accomplished with the UNIX command *chmod*. This is accomplished as follows:

```
%chmod 744 submit
```

where % is the UNIX prompt.

The script can be run in background by typing the script name followed by “&.” Alternately, it can be submitted using the UNIX commands *at* or *batch*. The syntax of any of the UNIX commands can be obtained by referring to the man(ual) pages on your system.

REFERENCES

Anderson, Stuart L., 1990. "Random Number Generators on Vector Supercomputers and Other Advanced Architectures," *SIAM Review, 32*, pp.221-251.

Burns, Patrick J. and Pryor, Daniel V., 1989. “Vector and Parallel Monte Carlo Radiative Heat Transfer,” *Numerical Heat Transfer, Part B: Fundamentals, 16*, pp. 20-42.

Burns, Patrick J., Maltby, James D. and Christon, Mark A., 1990. “Large-Scale Surface to Surface Transport for Photons and Electrons via Monte Carlo,” *Computing Systems in Engineering*, *1*(1), pp. 75-99.

Burns, Patrick J, Loehrke, Richard I., Dolaghan, John S. and Maltby, James D., 1992. “Photon Tracing in Axisymmetric Enclosures,” *Developments in Radiative Heat Transfer, HTD-Vol. 203*, pp. 93-100, ASME, New York.

Crockett, David V., Maltby, James D. and Burns, Patrick J., 1990. “User’s Manual for MONT3E - A Three-Dimensional Electron-Tracing Code with Non-Uniform Magnetic Field, Release 5.0,” Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Dolaghan, John S., Loehrke, Richard I., and Burns, Patrick J., 1992. “User’s Manual for SMOOTH,” Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Dolaghan, John S., 1991. *A Monte Carlo Simulation of Molecular Redistribution in an Enclosure due to Sputtering*, M.S. Thesis, Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Maltby, James D., 1987. *Three-Dimensional Simulation of Radiative Heat Transfer by the Monte Carlo Method*, M.S. Thesis, Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Maltby, James D., 1990. *Analysis of Electron Heat Transfer via Monte Carlo Simulation*, Ph.D. Dissertation, Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Maltby, James D. and Burns, Patrick J., 1991. “Performance, Accuracy and Convergence in a Three-Dimensional Monte Carlo Radiative Heat Transfer Simulation,” *Numerical Heat Transfer, Part B; Fundamentals,* 16, pp. 191-209.

Maltby, James D., Burns, Patrick J. and Winn, C. Byron, 1986. “Monte Carlo Simulation of Radiative Heat Transport in Passive Solar Buildings,” *Proceedings of the 1986 American Solar Energy Society Conference,* Boulder, Colorado (June 9-11, 1986).

Margolies, Dave,1986. Personal communication, LLNL.

Pryor, Daniel V. and Burns, Patrick J., July 21-25 1986. “A Parallel Monte Carlo Model for Radiative Heat Transfer,” presented at the 1986 SIAM Meeting, Boston, MA.

Shapiro, Arthur B., 1986. “TOPAZ2D - A Two-Dimensional Finite Element Code for Heat Transfer Analysis, Electrostatic and Magnetostatic,” Lawrence Livermore National Laboratory, UCID-20824.

Shapiro, Arthur B., 1985. “TOPAZ3D - A Three-Dimensional Finite Element Heat Transfer Code,” Lawrence Livermore National Laboratory, UCID-20484.

Shapiro, Arthur B., 1983. “FACET - A Radiation View Factor Computer Code for Axisymmetric, Two-Dimensional Planer, and Three-Dimensional Geometries with Shading,” Lawrence Livermore National Laboratory, UCID-19887.

Shivaswamy, Kiran A., Nagesh, Sukhi and Burns, Patrick J., 1994. “User’s Manual for the Program MPLOT,” Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.

Statton, E. Scott, 1983. *MONTE - A Two-Dimensional Monte Carlo Radiative Heat Transfer Code*, M.S. Thesis, Department of Mechanical Engineering, Colorado State University, Fort Collins, CO 80523.