

2 mif

X-324-73-272

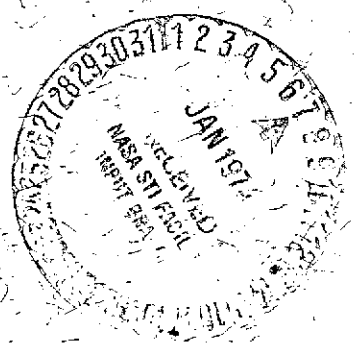
PREPRINT

NASA TM X- 70538

VIEW FACTOR COMPUTER PROGRAM (PROGRAM VIEW) USER'S MANUAL

EDWARD F. PUCCINELLI

JULY 1973



GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND



(NASA-TM-X-70538) VIEW FACTOR COMPUTER
PROGRAM (PROGRAM VIEW) USER'S MANUAL
(NASA) 127-p HC \$8.50 CSSL 20K


116

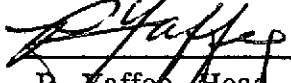
N74-13638

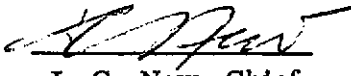
Unclas
G3/32 20655

VIEW FACTOR COMPUTER PROGRAM
(PROGRAM VIEW)
USER'S MANUAL

Edward F. Puccinelli
Test and Evaluation Division

Reviewed by 
A. J. Villasenor

Reviewed by 
P. Yaffee, Head
Electronics Test Branch

Approved by 
J. C. New, Chief
Test and Evaluation Division

July 1973

GODDARD SPACE FLIGHT CENTER
Greenbelt, Maryland

CONTENTS

	<u>Page</u>
1. INTRODUCTION	1-1
2. VIEW PRIMER	2-1
2.1 Computing View Factors	2-1
2.2 Available Finite Element Shapes	2-2
2.3 Describing Elements using NASTRAN-Type Data	2-3
2.3.1 Input Format	2-3
2.3.2 Dimensions and Locations of Elements	2-4
2.4 Describing Elements Using RAVFAC-Type Data	2-7
2.4.1 Input Format	2-7
2.4.2 Dimensions and Locations of Surfaces	2-8
2.5 Ordering Input and Job Control Cards	2-10
2.6 General	2-10
3. MODELING	3-1
3.1 Modeling Considerations	3-1
3.2 Coordinate Systems	3-5
4. NASTRAN ELEMENTS	4-1
5. RAVFAC SURFACES	5-1
6. INPUT	6-1
6.1 Organization	6-1
6.2 Control Cards	6-1
6.2.1 Title Card	6-2
6.2.2 Case Control Card	6-3
6.2.3 Format Type Card	6-6
6.2.4 ENDDATA Card	6-6
6.2.5 -1 Card	6-6
6.2.6 ENDCASE Card	6-6
6.2.7 -1 Card (For Local Coordinate Systems)	6-6

PRECEDING PAGE BLANK NOT FILMED

CONTENTS (Continued)

	<u>Page</u>
6.3 NASTRAN-Formatted Input Data Cards	6-7
6.3.1 CHBDY Input Data Card	6-8
6.3.2 CORD1C Input Data Card	6-12
6.3.3 CORD1R Input Data Card	6-14
6.3.4 CORD1S Input Data Card	6-16
6.3.5 CORD2C Input Data Card	6-18
6.3.6 CORD2R Input Data Card	6-20
6.3.7 CORD2S Input Data Card	6-22
6.3.8 GRID Input Data Card	6-24
6.3.9 GRDSET Input Data Card	6-25
6.3.10 PHBDY Input Data Card	6-26
6.3.11 \$VIEW Input Data Card	6-27
6.4 RAVFAC-Formatted Input Data Cards	6-29
6.4.1 RAVFAC Card Number 1 - Format I5, 2I1, 43X, 74A, A2	6-31
6.4.2 RAVFAC Card Number 2 - Format 5X, 5I5, 5E10.5	6-32
6.4.3 RAVFAC Card Number 3 - Format 5X, I5, 10X, 6E10.5	6-33
6.4.4 Local Coordinate System Card-Format 5X, I5, 10X, 6E10.5	6-35
6.5 360/95 Job Control Cards	6-36
6.5.1 Region Request	6-37
6.5.2 Time Estimation	6-37
7. OUTPUT	7-1
7.1 RADLST Output Data Card	7-1
7.2 RADMTX Output Data Card	7-2

CONTENTS (Continued)

	<u>Page</u>
8. ERROR AND WARNING MESSAGES	8-1
8.1 Explanations of Error Messages	8-1
8.2 Explanations of Warning Messages	8-4
9. RESTART	9-1
APPENDIX A - Sample Problem Using Only NASTRAN-Type Data . . .	A-1
APPENDIX B - Sample Problem Using NASTRAN and RAVFAC- Type Data	B-1
APPENDIX C - Sample Problem Demonstrating the Use of Local Coordinate Systems.	C-1

ILLUSTRATIONS

<u>Figure</u>		<u>Page</u>
2.1	Example of View Factor Computation	2-1
2.2	Fields of a NASTRAN-Type Data Card	2-3
2.3	Continuation of a NASTRAN CHBDY Card.	2-4
2.4	Basic Coordinate System and Two Grid Cards	2-5
2.5	CHBDY Card Used to Define an AREA4 Element	2-6
2.6	CHBDY Card Used to Define an AREA3 Element	2-6
2.7	RAVFAC-Type Rectangular Surface Divided into Six Elements	2-7
2.8	Relationship of Basic and Surface Coordinate System	2-8
2.9	Specification of a Rectangular Surface	2-9
2.10	Specification of a Cylindrical Surface	2-9
2.11	Data Deck Organization by Cases	2-10
2.12	Deck Organization of More Than One Case Utilizing Both Types of Input Data.	2-11
2.13	Indicating the Active Side of a RAVFAC Rectangular Element	2-11
2.14	Indicating the Active Side of a RAVFAC Parabolic Element	2-12
2.15	Direction of V Indicates Active Side.	2-12
2.16	Order of Grid Points Indicate Active Side.	2-13
3.1	View Factor Computation Between Two Elements	3-2
3.2	Computing a Nonzero View Factor	3-2
3.3	Computing the View Factor for a Cylinder	3-3
3.4	Subelement Mesh on a Plate	3-3
3.5	Spherical Element B Shades Subelements A and C	3-4
3.6	Computing View Factor for a Fine Mesh of Subelements	3-4
3.7	Cylinder Modeled With Eight-by-Four Mesh	3-5
3.8	Various Methods of Describing Locations of Elements of a Model.	3-6

ILLUSTRATIONS (Continued)

<u>Figure</u>		<u>Page</u>
4.1	Circular Plate With Two-by-Four Mesh of Subelements	4-1
4.2	Rectangular Plate With Four-by-Three Mesh of Subelements	4-2
4.3	Cylinder With Two-by-Six Mesh of Subelements	4-2
4.4	Triangular Plate With Seven-by-Three Mesh of Subelements	4-3
4.5	Nonrectangular Quadrilateral Plate With Specified Four-by-Three Mesh of Subelements	4-3
4.6	Mesh Formed if AREA4 Element is Rectangular	4-4
5.1	Surface Type ± 1 Rectangle	5-2
5.2	Surface Type ± 2 Disk	5-2
5.3	Surface Type ± 3 Trapezoid	5-2
5.4	Surface Type ± 4 Cylinder	5-3
5.5	Surface Type ± 5 Cone	5-3
5.6	Surface Type ± 6 Sphere	5-4
5.7	Surface Type ± 7 Circular Paraboloid	5-4
5.8	Four-by-Three Element Mesh With Three-by-Four Subelement Mesh	5-5
6.1	Input Data Deck	6-1
6.2	Organization of a Case Consisting Only of NASTRAN Data	6-2
6.3	Organization of a Case Consisting Only of RAVFAC Data	6-2
6.4	Organization of a Case Consisting of Both NASTRAN and RAVFAC Data	6-3
6.5	Choosing a View Factor Computation Method	6-4
6.6	Format and Example CHBDY Card	6-8
6.7	POINT Element—Flat Disk Shape	6-10
6.8	LINE Element—Flat Rectangular Plate	6-10

ILLUSTRATIONS (Continued)

<u>Figure</u>		<u>Page</u>
6.9	REV Element—Three-Dimensional Cone	6-11
6.10	AREA3 Element—Flat Triangular Plate	6-11
6.11	AREA4 Element—Flat Quadrilateral Plate	6-11
6.12	Cylindrical Coordinate System	6-12
6.13	Format and Example CORD1C Card	6-12
6.14	Rectangular Coordinate System	6-14
6.15	Format and Example CORD1R Card	6-14
6.16	Spherical Coordinate System	6-16
6.17	Format and Example CORD1S Card	6-16
6.18	Cylindrical Coordinate System	6-18
6.19	Format and Example CORD2C Card	6-18
6.20	Rectangular Coordinate System	6-20
6.21	Format and Example CORD2R Card	6-20
6.22	Spherical Coordinate System	6-22
6.23	Format and Example CORD2S Card	6-22
6.24	Format and Example GRID Card	6-24
6.25	Format and Example GRDSET Card	6-25
6.26	Format and Example PHBDY Card	6-26
6.27	Format and Example \$VIEW Card	6-27
6.28	Location of LINE Elements For DISLIN < 0	6-28
6.29	RAVFAC Input With No Local Coordinate Systems Specified	6-29
6.30	RAVFAC Input With Local Coordinate Systems Specified	6-30
6.31	Basic Coordinate System Rotation	6-34
6.32	Rotations Order	6-34
7.1	Format and Example RADLST Card	7-2
7.2	Format and Example RADMTX Card	7-2

TABLES

<u>Table</u>		<u>Page</u>
2.1	Element Shapes	2-2
2.2	NASTRAN Element Names	2-4
4.1	NASTRAN Elements	4-1
6.1	Sample Runs/Times	6-38

1. INTRODUCTION

The purpose of program VIEW is to compute view factors between specified surfaces and to be compatible with level 15.5 of the NASTRAN¹ structural analysis program. Program VIEW is a modification of a (finite element) view factor computation program called RAVFAC². VIEW is designed to run on an IBM System/360 operating under OS (operating system), with a minimum region size of 110 K bytes. The actual computation of view factors is still performed exactly as it was in the original version of RAVFAC. In developing VIEW, RAVFAC was modified to satisfy the following compatibility requirements:

- Accept finite element input which can also be used as input to NASTRAN.
- Produce output (view factors) in a format which can be used as input to NASTRAN.
- Follow NASTRAN program design so that in the future VIEW can be incorporated into NASTRAN as a subroutine.

The VIEW program permits computation of the view factors between surfaces, taking into account the presence of any intermediate surfaces. VIEW also computes these view factors either by contour integration or by finite difference (double summation) methods. The first method is more accurate, but the second method is faster. Either method may be selected or a criterion may be specified which causes the program to select the best method based upon the geometry of the problem.

Important features of the program include:

- A restart capability, which protects a user against having to rerun an entire problem should a computer failure occur.

¹NASTRAN is a structural analysis program developed by NASA at GSFC and is available from COSMIC. Level 15.5 of NASTRAN includes thermal analysis capabilities.

²RAVFAC was developed by J. K. Lovin and A. W. Lubkowitz, Lockheed Missile and Space Co., Huntsville, Alabama, November 1969, Contract NAS8-30154. RAVFAC is available from COSMIC under No. MFS-21075.

- The ability to dynamically allocate available core space, thus allowing the user to request the amount of space in the computer required for his problem by using the region parameter on the job card. There is no maximum number of elements to which the user is limited, except that the computer's capacity may not be exceeded.
- The ability to accept one or a combination of two input formats. A surface may be modeled by describing elements to program VIEW exactly as elements are described to RAVFAC or NASTRAN. Therefore, the VIEW program has RAVFAC-type and NASTRAN-type inputs.
- The ability to run several problems in sequence in one job submission. Each problem run is referred to as a "case".

Programming information may be found in the "Programmer's Manual for VIEW a Modification of the RAVFAC View Factor Program for Use with the NASTRAN Thermal Analyzer on IBM-360 Series Computers," GSFC Document No. X-322-73-120, dated March 1973.

2. VIEW PRIMER

This section contains a general discussion on how to compute view factors, describe the shape of elements, and describe the dimensions and locations of elements using RAVFAC- and NASTRAN-type data. The user should read this section to get an understanding of what type of information is to be input. This section does not explain how cards should be punched. The ordering of input and job control cards is also discussed. General information concerning the use of the VIEW program is presented in paragraph 2.6.

2.1 COMPUTING VIEW FACTORS

The computation of view factors is accomplished as follows:

- The user constructs a finite element model of the surfaces for which he wishes to have the view factors computed.
- The input data describes the location and shape of the finite elements relative to a basic coordinate system.
- The program outputs the view factor from each element to each other element, taking into consideration the case when one or more elements block the view between two elements.

An example of view factor computation is shown in Figure 2.1. The user describes the shape and location of elements 1, 2, and 3. The program computes view factors from E1 to E3, E3 to E1, E3 to E2, E2 to E3, E1 to E2, and E2 to E1, taking into account that E3 partially shades E1 from E2. The equation used for computing these view factors may be found in Appendix E of the Programmer's Manual. (See page 1-2.)

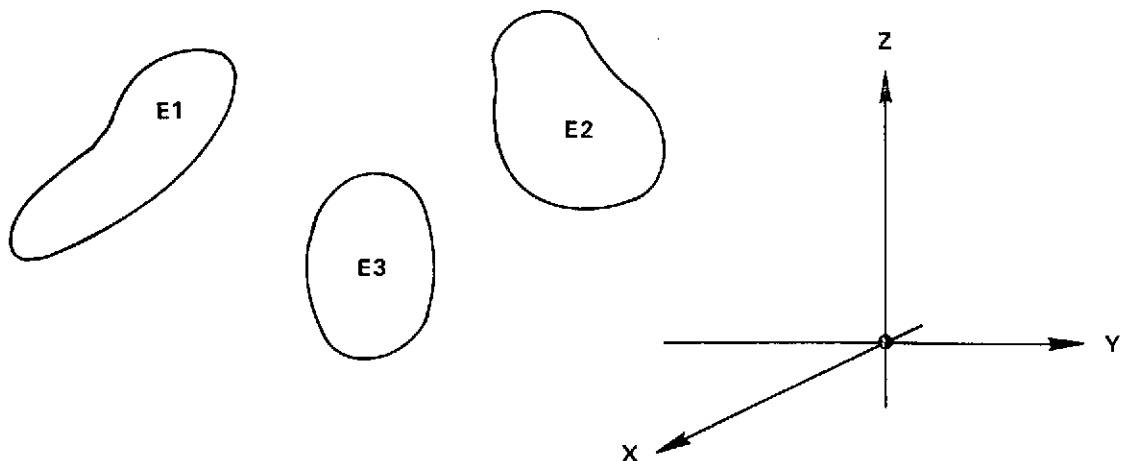


Figure 2.1. Example of View Factor Computation

2.2 AVAILABLE FINITE ELEMENT SHAPES

The VIEW program allows the user to employ either NASTRAN-type input, RAVFAC-type, or a combination of both.

Using only the NASTRAN-type input, the input data cards and punch output are compatible with the input to the NASTRAN program. Thus, a thermal analysis may be made using both the NASTRAN and VIEW programs, with data cards common to both. The availability of RAVFAC-type input is advantageous in some cases because it allows for more element shapes than are available with NASTRAN-type data. To be completely flexible, VIEW allows the use of both types of data in making a single model. The element shapes which may be described to the program according to the type of data that is used are given in Table 2.1.

Table 2.1
Element Shapes

NASTRAN-Type Data	RAVFAC-Type Data
Rectangular plate	Rectangular plate
Circular plate	Circular plate or portions
Triangular plate	Trapezoidal plate
Cylindrical shell	Cylindrical shell or portions
Conical shell	Conical shell or portions
Quadrilateral plate	Spherical shell or portions
	Circular parabolic shell or portions

These elements are described and illustrated in detail in Sections 4 and 5.

The user specifies an element shape on a NASTRAN-type data card called CHBDY (see paragraph 6.3). If RAVFAC-type data is employed, then the element shape is specified on a RAVFAC-type data card number 2 (see paragraph 2.4).

2.3 DESCRIBING ELEMENTS USING NASTRAN-TYPE DATA

2.3.1 Input Format

The NASTRAN-type data cards used by VIEW are a subset of the cards known as "bulk data" cards in the NASTRAN program documentation. The NASTRAN-type input to the VIEW program must correspond to the following rules:

- Cards are divided into ten eight-column fields as shown in Figure 2.2.

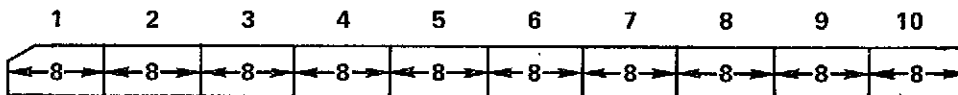


Figure 2.2. Fields of a NASTRAN-Type Data Card

- A mnemonic must be punched in field 1 beginning in column 1 to identify the card type. The mnemonics are listed in Section 6 and are comprised of names such as GRID, CHBDY, and CORD1S.
- Fields 2 through 9 are for data items. All data items must lie completely within a designated field, have no imbedded blanks, and must be of the proper type, that is, blank, integer, real, or BCD. All real numbers, including zero, must contain a decimal point. A blank will be interpreted as a real zero or integer zero as appropriate. Real numbers may be encoded in various ways. For example, the real number 7.0 may be encoded as 7.0, .7E1, 0.7 + 1, 70. -1, .70 + 1. Only the mnemonic in field 1 need be left-justified.
- Normally field 10 is reserved for optional user identification. However, the data required for a few types of cards do not always fit in fields 2 through 9. In this case, the remaining data is punched on a continuation card. In continuation cards, field 10, except column 73 which is not referenced, is used in conjunction with field 1 of the continuation card as an identifier and hence must contain a unique entry. The continuation card must contain the symbol + in column 1, followed by the same seven characters that appeared in columns 74 through 80 of field 10 of the card that is being continued. The continuation card need not physically follow the card it continues in the NASTRAN-type data deck; however, run time will be reduced if it does. Figure 2.3 shows an example of a CHBDY card and its continuation card.

	1	2	3	4	5	6	7	8	9	10
Card 1	CHBDY	X	X	X	X	X	X	X	X	+ABC
Continuation of card 1	+ABC	X	X	X	X	X	X	X	X	

Figure 2.3. Continuation of a NASTRAN CHBDY Card

- Any NASTRAN-type data card which is mispunched in field 1 is ignored by VIEW. A message is printed out warning the user that "x" input cards were not recognized. The user should always check this message.

2.3.2 Dimensions and Locations of Elements

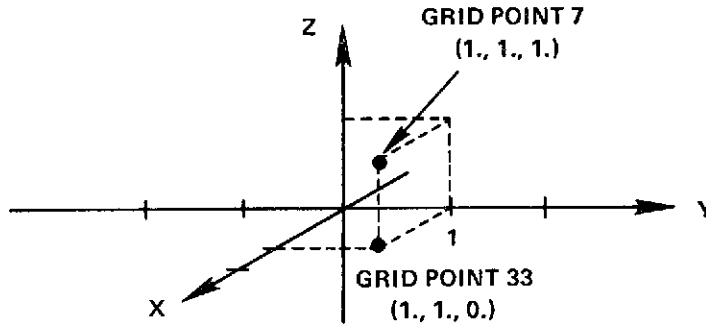
There are only five basic element shapes which may be described by using NASTRAN-type data. Each of these elements has a NASTRAN name as shown in Table 2.2.

Table 2.2
NASTRAN Element Names

Element Shape	NASTRAN Name
Circular plate	POINT element
Rectangular plate	LINE element
Conical or cylindrical shell	REV element
Triangular plate	AREA3 element
Quadrilateral plate	AREA4 element

Describing the dimension and location of these five elements is done by using grid points. Grid points are points in the three-dimensional space whose coordinates are specified on a NASTRAN-type data card called GRID (see paragraph 6.3). The coordinates of a grid point may be specified relative to the basic rectangular coordinate system, or they may be specified relative to a local coordinate system which in turn is related to the basic coordinate system by use of NASTRAN-type data CORD cards. The local coordinate system may be either rectangular, spherical, or cylindrical (see paragraph 6.3).

Example—The following grid points are located in space relative to a basic rectangular coordinate system and described to the program by the data cards shown in Figure 2.4.



GRID	7		1.	1.	1.				
GRID	33		1.	1.	0.				

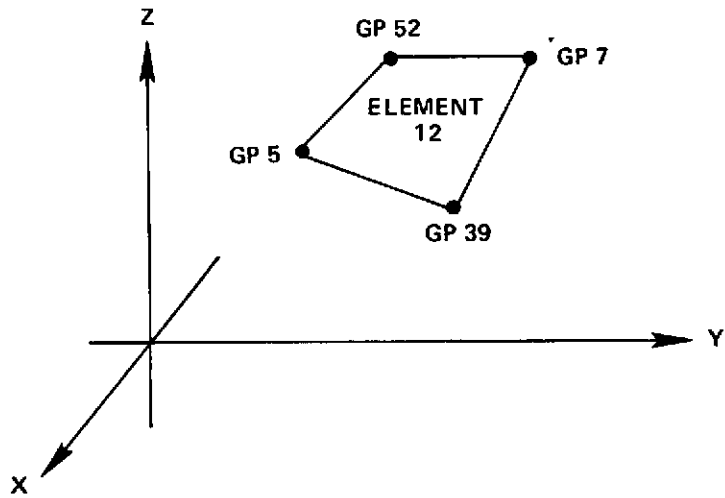
Figure 2.4 Basic Coordinate System and Two Grid Cards

The name and identification number of an element is punched on a NASTRAN-type data card called CHBDY. On this card the user also punches the grid point identification numbers of the grid points which determine the dimension and location of element 12. The specific format of the CHBDY card can be found in paragraph 6.3.

Example—The user defines the size and location of a quadrilateral plate element he wishes to designate as number 12 by using the CHBDY card shown in Figure 2.5.

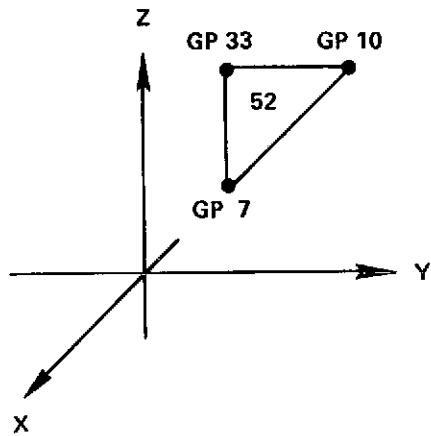
The location of the grid points are specified on GRID cards. The grid points listed on the CHBDY card must be planar and must be listed sequentially. (The direction of ordering determines the active side of the element, as will be explained later.)

Example—The same comments in the previous example apply for the triangular element as shown in Figure 2.6 but only three grid points need be on the CHBDY card.



CHBDY	12		AREA 4	5	39	7	52	X	
-------	----	--	--------	---	----	---	----	---	--

Figure 2.5. CHBDY Card Used to Define an AREA4 Element



CHBDY	52		AREA 3	7	10	33			
-------	----	--	--------	---	----	----	--	--	--

Figure 2.6. CHBDY Card Used to Define an AREA3 Element

The remaining elements cannot be completely defined using only grid points. Other information must appear on the CHBDY card. Paragraph 6.3 explains in detail how to specify all the elements using NASTRAN-type data.

2.4 DESCRIBING ELEMENTS USING RAVFAC-TYPE DATA

2.4.1 Input Format

RAVFAC-type data allows the user to specify a surface shape which is divided into a mesh of elements by the program. (View factors are output on an element-to-element basis.) Thus, by specifying a surface shape, its location, dimensions, and an m-by-n mesh size, the user actually describes mn elements simultaneously. This saves the trouble of specifying mn separate shapes, locations, and dimensions.

Three RAVFAC-type data cards are required to define each surface: cards number 1, 2, and 3. In card number 1 the user specifies the identification number of the surface, any desired comment, plus shading information. (Shading information is explained in Section 3.1). Card 2 is used to define the surface shape, element mesh size and surface dimensions relative to a surface coordinate system. Card 3 is used to relate the surface coordinate system to the basic coordinate system. The specific format of each card is given in Section 6.4.

Example—Figure 2.7 shows a RAVFAC-type data rectangular surface for which a two-by-three element mesh was specified. The program recognizes six adjoining rectangular elements, numbers them as described in the last part of Section 5, and then calculates the view factors to and from each one. A surface may be made one element by specifying a one-by-one mesh.

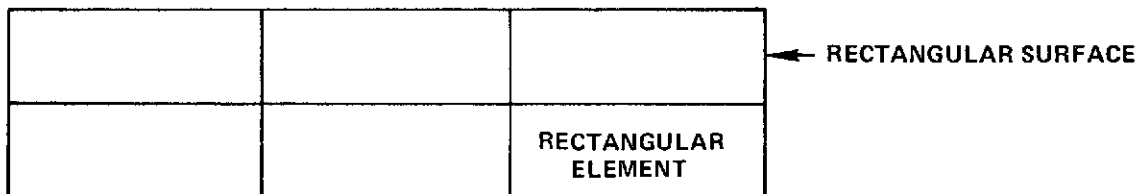


Figure 2.7. RAVFAC-Type Rectangular Surface Divided into Six Elements

2.4.2 Dimension and Location of Surfaces

As discussed in paragraph 2.4.1, the user specifies surfaces and element meshes on each surface. Knowing how an element mesh is defined over a surface, the user knows where each element is located once he has specified the location and dimensions of a surface. (See Section 5.) This paragraph will be concerned only with the question of how surfaces are specified.

Each of the seven different RAVFAC surfaces has a surface coordinate system associated with it. Drawings of this concept can be seen in Section 5. Each surface is located in space by relating the surface coordinate system to the basic coordinate system of the model, or by relating it to a local coordinate system which in turn must be related to the basic coordinate system.

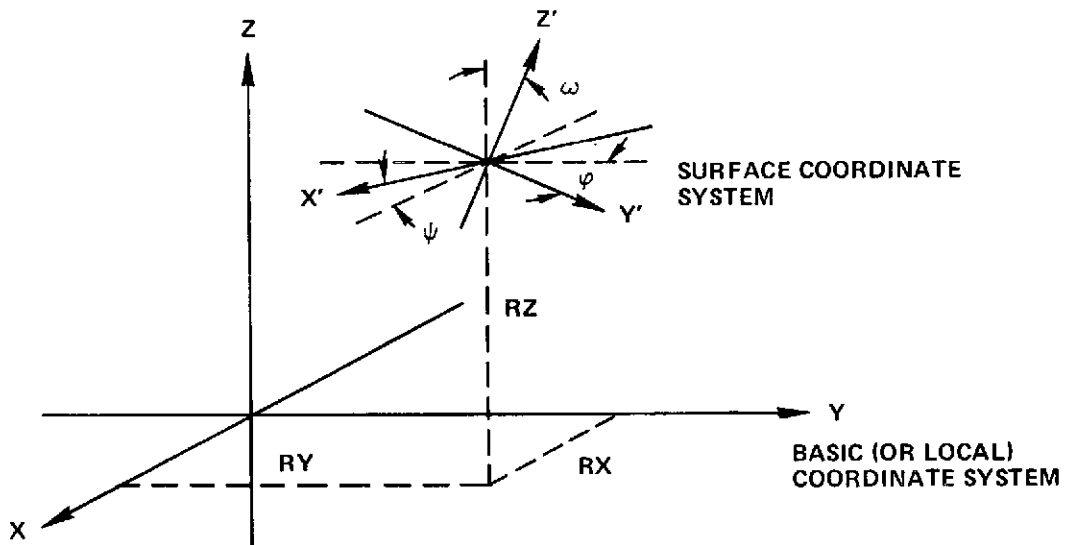


Figure 2.8. Relationship of Basic and Surface Coordinate System

Figure 2.8 shows the distances RX, RY, and RZ between the origins of the two coordinate systems. These distances, as well as PHI, PSI, and OMEGA, the angles which represent the rotation of the basic (or local) coordinate system into the surface coordinate system, must be specified on RAVFAC-type data card 3. The rotations must be listed in the order:

PHI = rotation of Y toward X about Z in degrees
 PSI = rotation of X toward Z about Y in degrees
 OMEGA = rotation of Y toward Z about X in degrees

The size of a surface is specified by using the five parameters ALPHA, BMIN, BMAX, GMIN, and GMAX. These parameters define different surfaces in different ways as Figures 2.9 and 2.10 show.

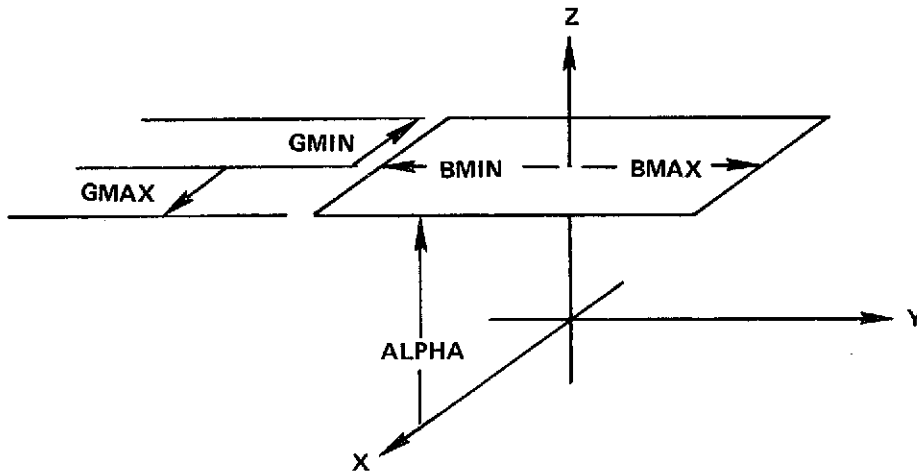


Figure 2.9. Specification of a Rectangular Surface

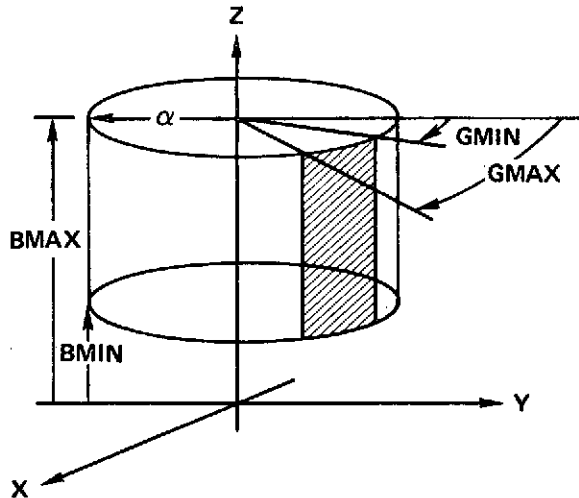


Figure 2.10. Specification of a Cylindrical Surface

See Section 5 for specification details of all surfaces.

2.5 ORDERING INPUT AND JOB CONTROL CARDS

Only a brief discussion on this subject will be made here. Detailed explanations are provided in paragraphs 6.1, 6.2, and 6.5.

The set of input cards, of either one or both data types, which allows the program to compute the view factors of a given model is called a "case." The program allows the user to input several cases back-to-back, so that once the view factors for one case are found the program proceeds to the next case (Figure 2.11).

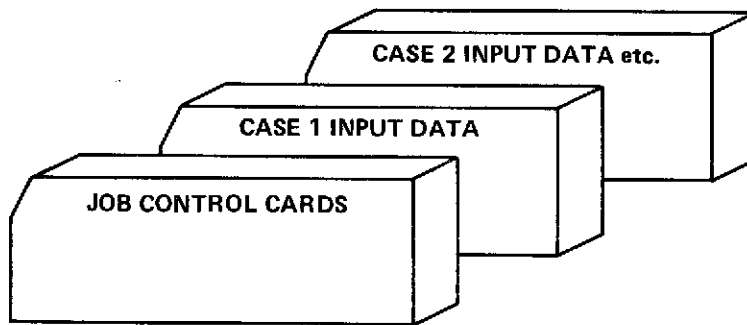


Figure 2.11. Data Deck Organization by Cases

The cards which comprise the Job Control Cards deck in Figure 2.11 are described in paragraph 6.5. Each case is comprised of either five or seven case control cards and the NASTRAN- and/or RAVFAC-type input data. The format of the control cards is explained in paragraph 6.2.

If a case has both types of input data, it must be ordered as shown in Figure 2.12. If a case has only one type of input data, there would be only that type of data and its control cards. A detailed explanation is given in paragraph 6.1.

2.6 GENERAL

Although the method for describing the size and location of elements using either type of input data has been discussed, the program requires some additional information. All element shapes are two-dimensional. Although each element has two sides, the program will only compute view factors to and from one side. Therefore, the user must specify which side is active, that is,

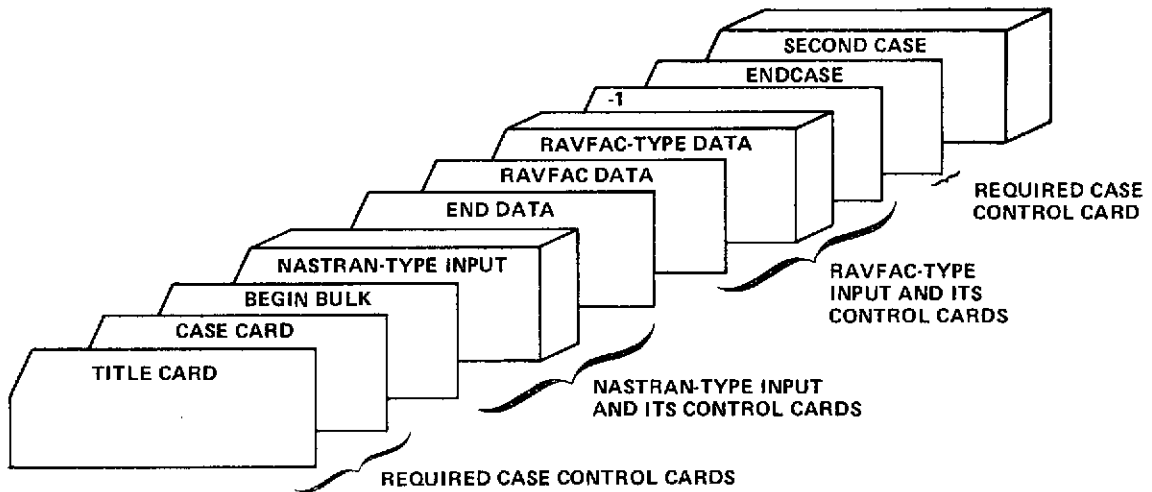


Figure 2.12. Deck Organization of More Than One Case Utilizing Both Types of Input Data

to and from which side the view factors are to be computed. (The nonactive side can still cause shading.)

If the user is employing RAVFAC-type data, specify the active side on card type number 2 when indicating the surface shape using the ILK parameter. Refer to paragraph 6.4.2. For planar surfaces, a plus indicates the + Z side is active and a minus indicates the -Z side is active (Figure 2.13).

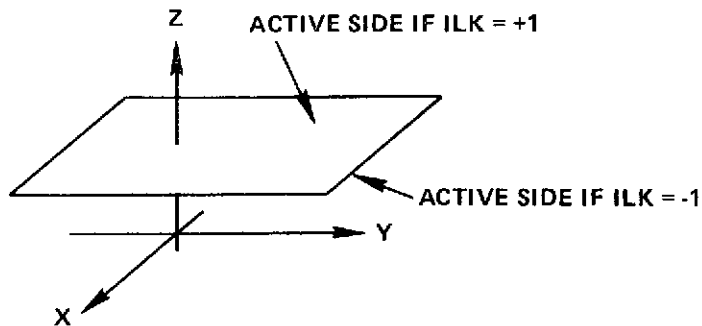


Figure 2.13. Indicating the Active Side of a RAVFAC Rectangular Element

For nonplanar surfaces, a plus indicates the outside is active while a minus indicates the inside is active (Figure 2.14).

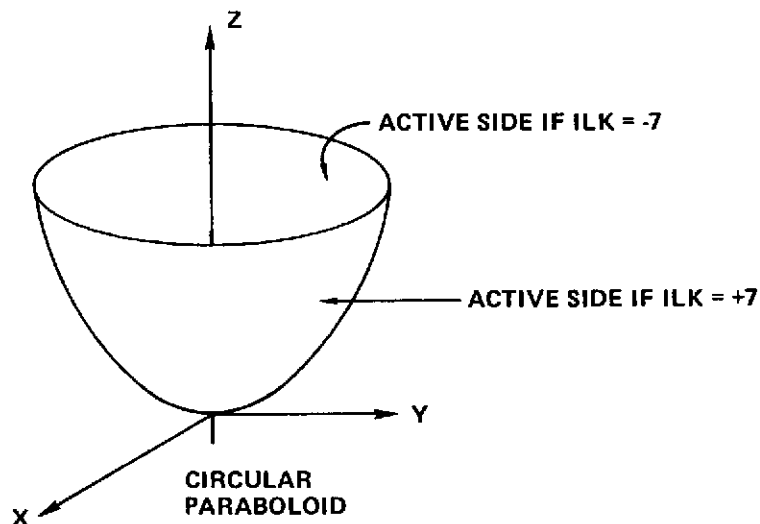


Figure 2.14. Indicating the Active Side of a RAVFAC Parabolic Element

If NASTRAN-type data is being employed, indicate which side of an element is active on a CHBDY card. For the REV, AREA3, and AREA4 elements, the active side is specified by the ordering of the grid points. (Refer to the description of CHBDY cards in paragraph 6.3.) For POINT and LINE elements, the active side is specified by the V vector on the CHBDY card (Figure 2.15).

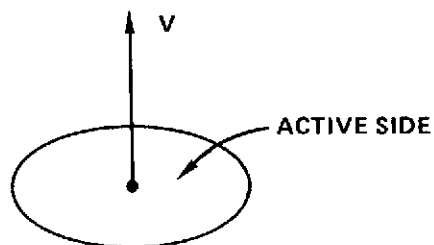


Figure 2.15. Direction of V Indicates Active Side

In Figure 2.16 if grid point numbers are listed on the CHBDY and in the order 1-2-3, 2-3-1, or 3-1-2, the active side is the one shown. A counter-clockwise ordering makes the other side active. Thus, for AREA3 and AREA4 elements the righthand rule gives the active side.

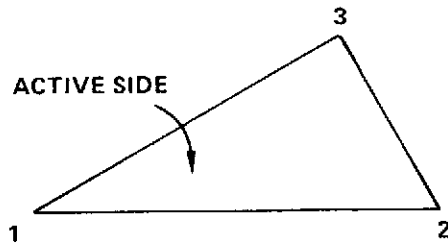


Figure 2.16. Order of Grid Points Indicate Active Side

A time-saving feature of the program should be emphasized. The program is capable of detecting whether one element causes shading between two other elements, as shown in Figure 2.1. However, a significant amount of computer time can be saved if the user tells the program which surfaces can never cause shading and/or never be shaded.

Using RAVFAC-type data, this specification is made on a card type number 1 (see paragraph 6.4 for details). Using NASTRAN-type input data, this specification is made on a \$VIEW card. The number of the \$VIEW card is given in field 9 of the CHBDY card (see paragraph 6.3 for further details).

The program also has the facility for creating an n-by-m mesh of subelements on a single element. View factors are calculated between subelements and internally recombined to print out the view factors between elements. Although this lengthens the run time it significantly increases the accuracy of answers, especially when shading occurs between two elements. This subelement mesh size is specified on a \$VIEW card if NASTRAN-type data is being used. It is specified on card type number 2 if RAVFAC-type data is used. At least a two-by-two subelement mesh is recommended on all elements until the user becomes quite familiar with the program (see Section 3 and paragraphs 6.3 and 6.4 for further details). A subelement mesh covers an element in the same manner that an element mesh covers a surface. It is recommended that the user look over the problems given in Appendixes A and B of this document.

VIEW also has a restart capability. This means that a continuous record can be made on a magnetic tape while view factor computations are being made. Should some machine disaster cause termination before all computations have been completed, then the user may employ the magnetic tape to pick up his job from the stopping point. This saves the time and cost of having to restart a job from the beginning. (See paragraphs 6.2 and 6.5 for further details.)

The remainder of this document contains detailed information about each of the topics covered in this "primer" section.

3. MODELING

The term modeling, as in modeling of a structure, is defined to mean the description of a given structure to program VIEW, by approximating the structure using any combination of the following seven element shapes, and by specifying their size and location in space.

- Triangular plate.
- Quadrilateral plate.
- Circular plate.
- Cylindrical shell.
- Conical shell.
- Spherical shell.
- Circular parabolic shell.

NOTE

The last two elements can be described only by using RAVFAC-type formatted input.

As discussed in Section 2, element shapes may be described to the program by using either RAVFAC- or NASTRAN-type input. It is important to realize that program VIEW needs to know only element shapes, their geometrical location, and which side of the element is to be considered for view factor calculation. Hence it is possible to compute view factors between elements which do not form a valid structural or thermal model.

3.1 MODELING CONSIDERATIONS

The user must specify which of the two sides of an element is to be considered active. That is, to and from which side are view factors to be computed. The inactive side will cause shading as well as the active side.

Besides the possibility of modeling incorrectly in a structural or thermal sense, there are also some models which lead to erroneous view factor calculations for geometric reasons. This can best be explained by a brief description of how the program determines when a view factor between two elements is to be computed.

In Figure 3.1, V_A is a vector normal to the centroid of element A, and points in the direction which the user specifies that element A can "see" and "be seen" by other elements; similarly for V_B . A vector is then drawn from the centroid of B to the centroid of A, forming the angles α and β . Two conditions must be

satisfied simultaneously if there is to be a nonzero view factor between A and B:

- $-90^\circ < \beta < +90^\circ \Rightarrow \cos \beta > 0$
- $+90^\circ < \alpha < +180^\circ$ or $-90^\circ > \alpha > -180^\circ \Rightarrow \cos \alpha < 0$

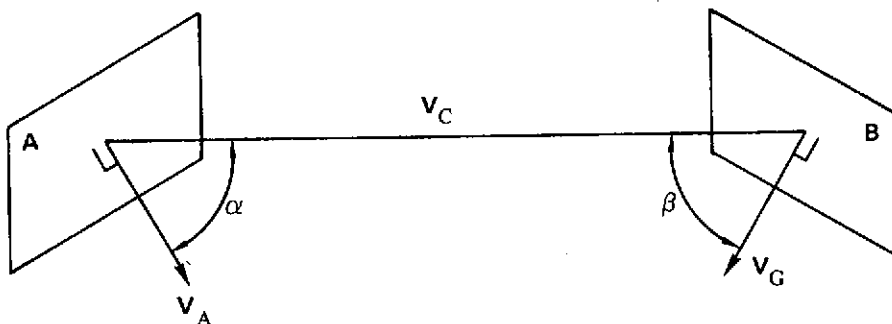


Figure 3.1. View Factor Computation Between Two Elements

This is equivalent to stating that the active sides must see each other. Hence, whenever possible, the user should avoid having two elements meet anywhere except at a common edge. For example, if a structure included a wall meeting another wall to form a T-shape, it should not be modeled using three quadrilateral elements, two (back-to-back) meeting the other at its center. Rather, four quadrilateral elements should be used, all meeting at a common edge.

On the left in Figure 3.2, element C, being right of center of element A, will have a view factor calculated from all of C to part of A. No view factor would be calculated between D and A.

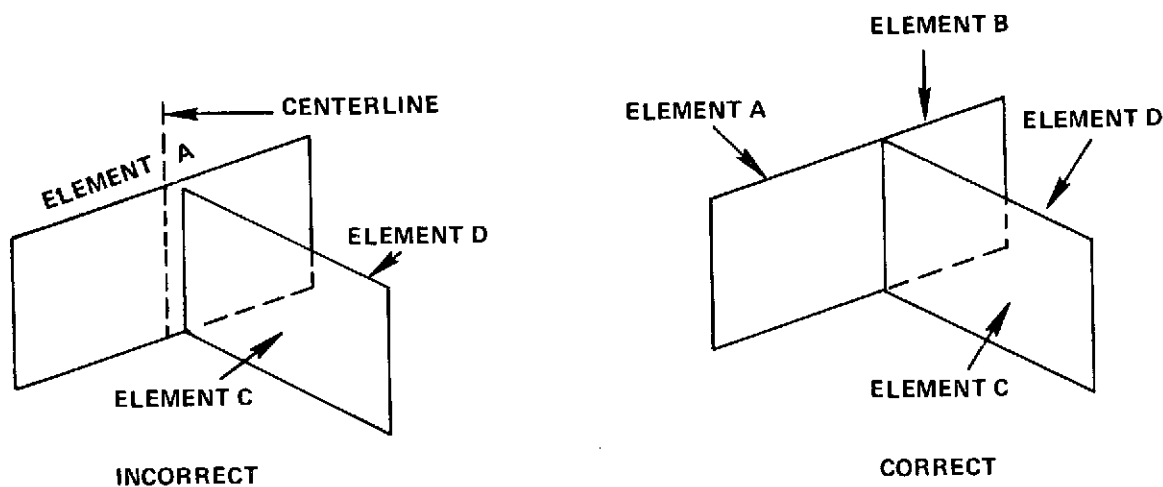


Figure 3.2. Computing a Nonzero View Factor

In some cases it is impossible to satisfy this rule. For example, suppose the structure involves a plate which separates two cylinders, as shown in Figure 3.3. The cylinder can and should be modeled as two separate cylinders which meet at the plate so that the previous rule is at least partially satisfied. The rule cannot be fully satisfied using the element shapes previously described.

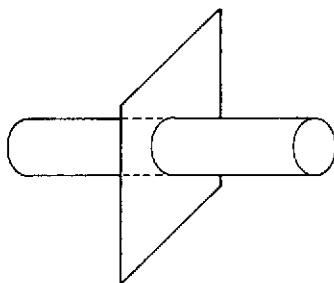


Figure 3.3. Computing the View Factor for a Cylinder

For such cases, the user should employ a feature of program VIEW which allows definition of a mesh of subelements* of any size on a particular element (Figure 3.4). The view factors are calculated from these subelements and automatically recombined to produce a view factor for the element which they comprise. In fact, the program performs the "can I see" check described in Figure 3.1 between every set of two subelements.

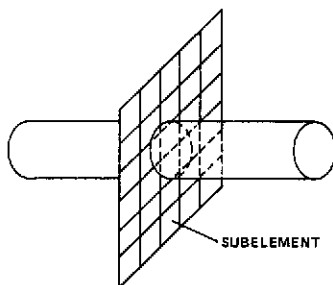


Figure 3.4 Subelement Mesh on a Plate

By using this approach, inaccurate view factors may occur only between the cylinders and the subelements they touch. The remaining view factors between other subelements and the cylinders are found accurately. Thus, the smaller

*For the user familiar with the RAVFAC program, the concept of subelements corresponds to elements in the RAVFAC documentation, while elements in this document correspond to RAVFAC nodes. In general, the terminology used in this document will follow that defined in NASTRAN.

the mesh the more accurate the view factors between the cylinders and elements, but the greater the run time.

A subelement is shaded from another subelement if the line connecting the center of the two subelements is intersected by any part of at least one other element (Figure 3.5). VIEW will automatically determine such situations if desired. However, the user may substantially reduce the problem run time if the elements which can shade and/or be shaded by other elements are specified.

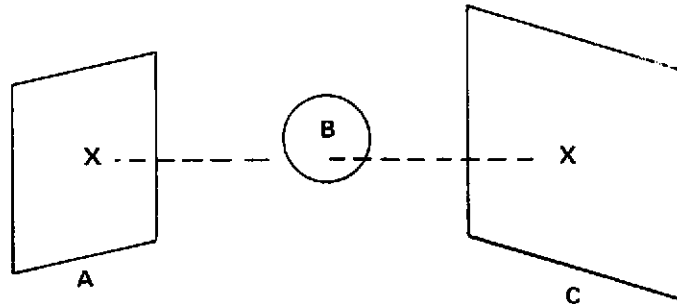


Figure 3.5. Spherical Element B Shades Subelements A and C

It is extremely important to realize that view factors are calculated on a subelement-to-subelement basis and then recombined to print out a view factor from element to element. In Figure 3.5, if the user specifies a one-by-one mesh of subelements on the elements A and C, a zero view factor would be calculated between A and C, although from the drawing this is not the true view factor. If the user specifies a finer mesh of subelements on elements A and C, as shown in Figure 3.6, then sphere B, which shades mostly subelement A2 from C2, would cause a zero view factor to be computed only between those two subelements. A nonzero view factor would be computed from each subelement of A to each subelement of C (except between A2 and C2), and then recombined to give a view factor between elements A and C. The user should note from this example that again, the finer the mesh of subelements the more accurate the view factors.

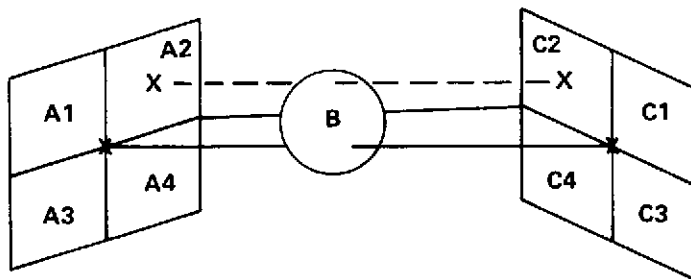


Figure 3.6. Computing View Factor for a Fine Mesh of Subelements

A subelement mesh must be specified for all nonplanar elements. The subelements for nonplanar elements are recognized by the program as flat rectangular plates and view factors are calculated from and to these plates. A cylinder modeled with an eight-by-four mesh, as shown on the left in Figure 3.7, is recognized by the program as shown on the right in Figure 3.7.

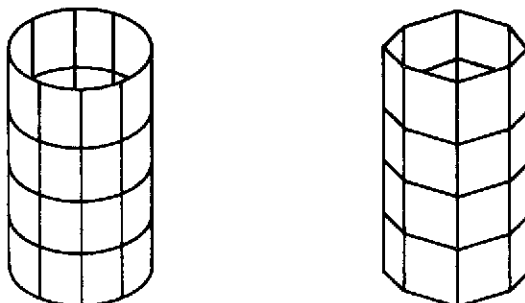


Figure 3.7. Cylinder Modeled With Eight-by-Four Mesh

While a fine mesh of subelements will produce more accurate answers than a coarse mesh, it will also increase computer run time. When not needed, a fine mesh of subelements should be avoided. An example of such a case is Figure 3.6 if no element shaded A from C. Here a one-by-one mesh of subelements on elements A and C should be adequate.

3.2 COORDINATE SYSTEMS

VIEW allows the user the convenience of specifying the location of different surfaces in different coordinate systems. In describing elements using NASTRAN input, coordinate systems are described and used just as in the NASTRAN program. Similarly, in describing surfaces using RAVFAC input, coordinate systems are described and used just as in the RAVFAC program.

Both types of input allow the user to specify local coordinate systems for convenience. That is, the user is allowed to locate other coordinate systems in space by specifying their orientation and displacement from the basic coordinate system. By using NASTRAN data, the user has the added choice of selecting the type of local coordinate system he desires. It may be either rectangular, spherical, or cylindrical.

A limitation imposed by VIEW which is not present in NASTRAN is that each local coordinate system must reference the basic coordinate system. One local coordinate system may not be specified relative to another local coordinate system. Also, the program will allow the use of, at most, forty NASTRAN local coordinate systems of type CORD 1 and forty of type CORD 2.

Figure 3.8 shows all the basic methods of describing the location of elements of a model. Each model must have one and only one basic coordinate system. For NASTRAN-type input each element of the model may be located in space by specifying the coordinates of bounding grid points relative to the basic coordinate system. For RAVFAC-type input, the user relates the location and orientation of an element coordinate system to the basic coordinate system.

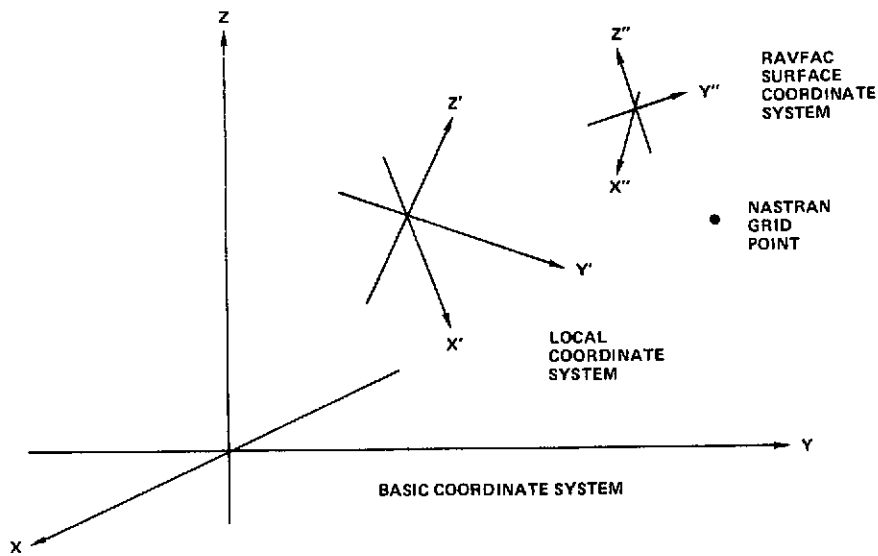


Figure 3.8. Various Methods of Describing Locations of Elements of a Model

As an extra feature of the program, the user may relate grid points or surface coordinate systems (depending on which type of input he is using) to one or more local coordinate systems. He then specifies the location and orientation of the local coordinate system to the basic coordinate system.

4. NASTRAN ELEMENTS

There are five element shapes which may be described to VIEW by using NASTRAN data (Table 4.1). More shapes are available using RAVFAC data (see Section 5). Either side of an element, but not both sides, may be designated as active, that is, the side to and from which view factors are computed. Specification of the active side is described in paragraph 6.3.

Table 4.1
NASTRAN Elements

Element Shape	NASTRAN Name
Circular plate	POINT
Rectangular plate	LINE
Cone or cylinder	REV
Triangular plate	AREA3
Quadrilateral plate	AREA4

There is one additional element shape which may be described to NASTRAN for heat transfer analysis purposes, but which is not available in VIEW, that is, the elliptic cylinder. A circular cylindrical shape is available by using the REV element. Figures 4.1 through 4.6 illustrate the element shapes (broad lines) and an example of their division into subelements (fine lines).

- POINT Element

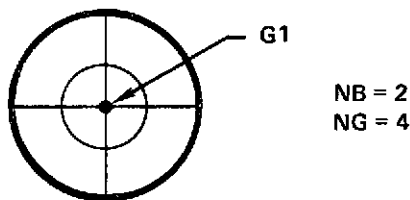


Figure 4.1. Circular Plate With Two-by-Four Mesh of Subelements

In Figure 4.1, NB is the division in circular sections while NG divides the disk into pie-shaped sections.

- LINE Element

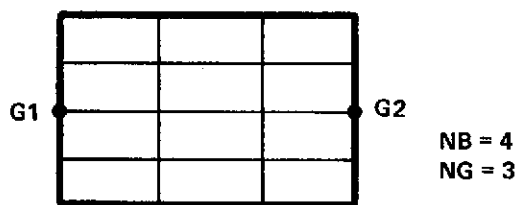


Figure 4.2. Rectangular Plate With Four-by-Three Mesh of Subelements

In Figure 4.2, NB is the division perpendicular to the line connecting the two grid points needed to give the location of the LINE element. NG is the division along the line.

- REV Element

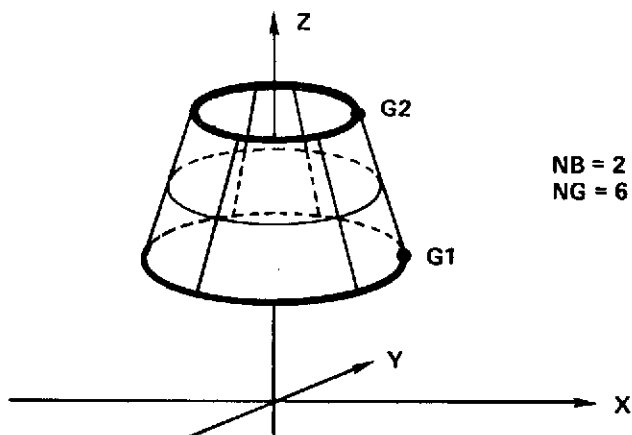


Figure 4.3. Cylinder With Two-by-Six Mesh of Subelements

In Figure 4.3, NB is the division along the axis of the element, while NG is the division around the surface. NASTRAN requires that grid points G1 and G2 be in the X-Z plane of the basic coordinate system.

- AREA3 Element

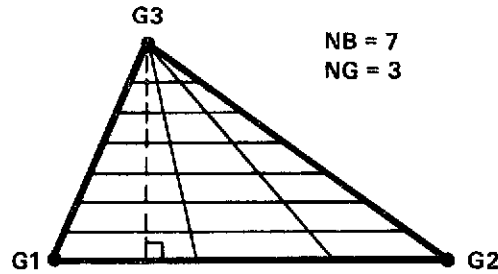


Figure 4.4. Triangular Plate With Seven-by-Three Mesh of Subelements

In Figure 4.4, the dashed line is a perpendicular from G3 to the line connecting G1 and G2. NB is the division along this line. NG is the division along the line connecting G1 and G2.

- AREA4 Element

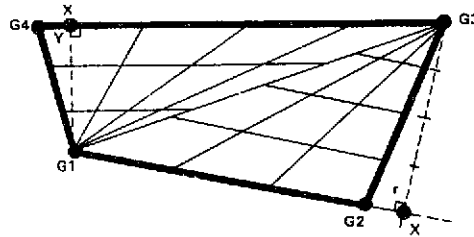


Figure 4.5. Nonrectangular Quadrilateral Plate With Specified Four-by-Three Mesh of Subelements

The complexity of Figure 4.5 is due to the fact that the element is divided into two triangular elements by connecting grid points G1 and G3. The subelement mesh size specified by the user (in this case $NB_1 = 4$ and $NG_1 = 3$) is applied to triangle G1, G2, G3, just as described for the AREA3 element. NG divisions are made along the line connecting G1 and G2, while NB divisions are made along a perpendicular to this line. However, for triangle G3, G4, G1, the subelement mesh size is chosen proportionate to that specified for the first triangle in the following manner:

$$\frac{NG_1}{\text{length (G1, G2)}} = \frac{NG_2}{\text{length (G3, G4)}}$$

$$\frac{NB_1}{\text{length (G3, X)}} = \frac{NB_2}{\text{length (G1, Y)}}$$

If either NB_2 or NG_2 are calculated to be zero, then both NB_2 and NG_2 are set to one and a warning message is printed. Note that in counting the number of elements in the problem, each nonrectangular AREA4 element should be counted as two elements. If the AREA4 element is rectangular, then it is not divided into two triangular elements by the program.

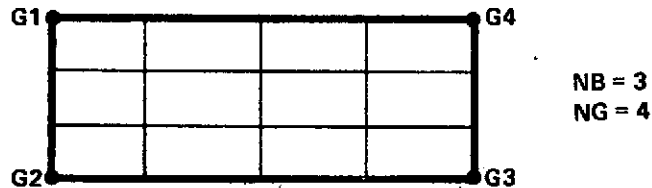


Figure 4.6. Mesh Formed if AREA4 Element is Rectangular

In Figure 4.6, NB divisions are again made along the line connecting G1 and G2, while NG divisions are made along the line connecting G2 and G3. In this case, the AREA4 element is counted as one element, as it is not broken into two triangular elements.

5. RAVFAC SURFACES

The VIEW program also accepts input in the same format as the RAVFAC program. (See Section 2.) The input format is described in detail in paragraph 6.4. While NASTRAN-type input allows the user to describe elements and the mesh of subelements into which each element is divided, RAVFAC-type input is somewhat more general.

The user may describe a surface comprised of a mesh of elements, each of which is comprised of a mesh of subelements. Hence, where with NASTRAN input the user might model a wall with 100 rectangular elements (10 along the base and 10 along an edge), with RAVFAC-type input the user may form the same model by specifying one surface comprised of a ten-by-ten mesh of elements. View factors are still output on an element-to-element basis.

Another advantage of RAVFAC-type input is that more surface shapes may be described than by using NASTRAN-type input. These surface types are:

- Rectangular plates.
- Circular plate or section thereof.
- Trapezoidal plate.
- Cylindrical shell or portion thereof.
- Conical shell or portion thereof.
- Spherical shell or portion thereof.
- Circular parabolic shell or portion thereof.

Each element is described by specifying its location and active side relative to the element coordinate system as shown in Figures 5.1 through 5.8. The element coordinate system is then related to the local or basic coordinate system. Refer to Section 3 and paragraph 6.4 for further details on specification of the active side.

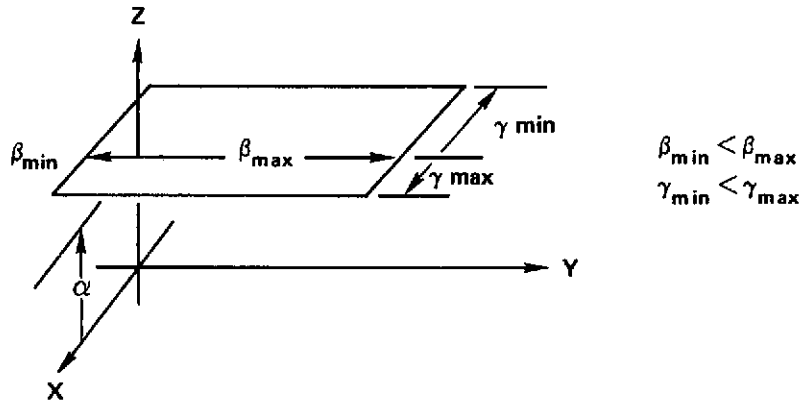


Figure 5.1. Surface Type ± 1 Rectangle

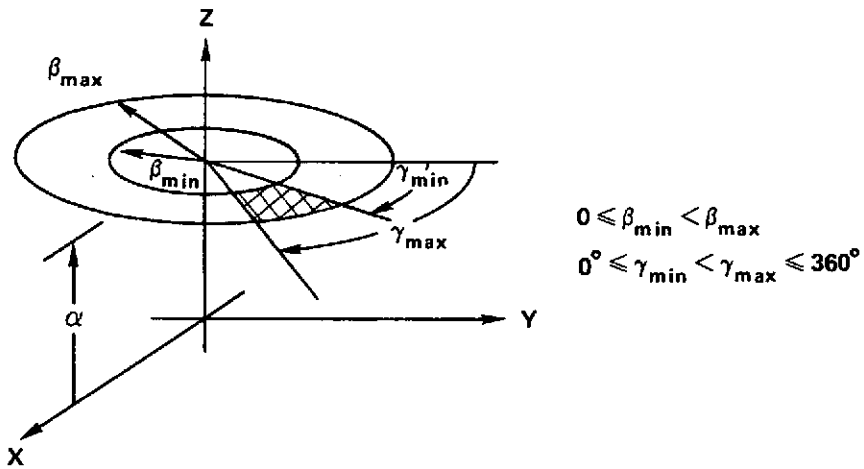


Figure 5.2. Surface Type ± 2 Disk

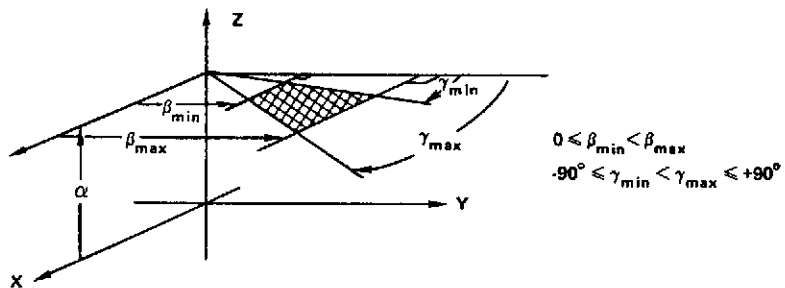


Figure 5.3. Surface Type ± 3 Trapezoid

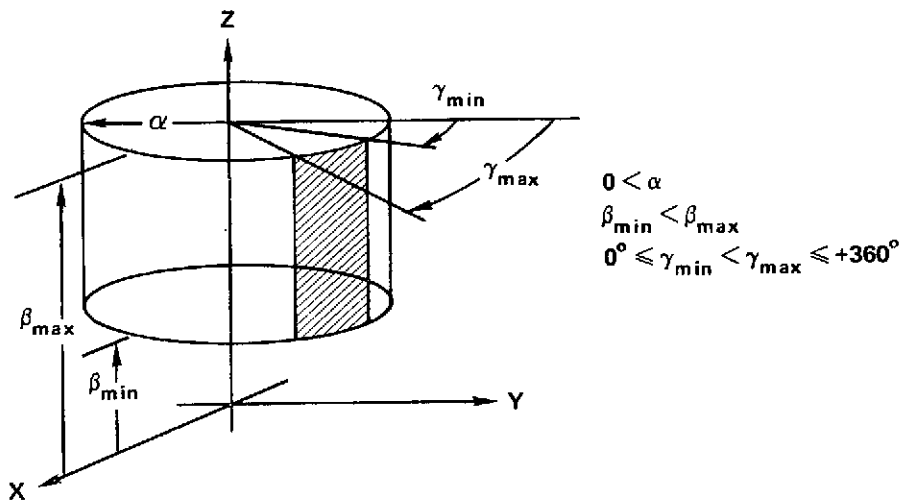


Figure 5.4. Surface Type ± 4 Cylinder

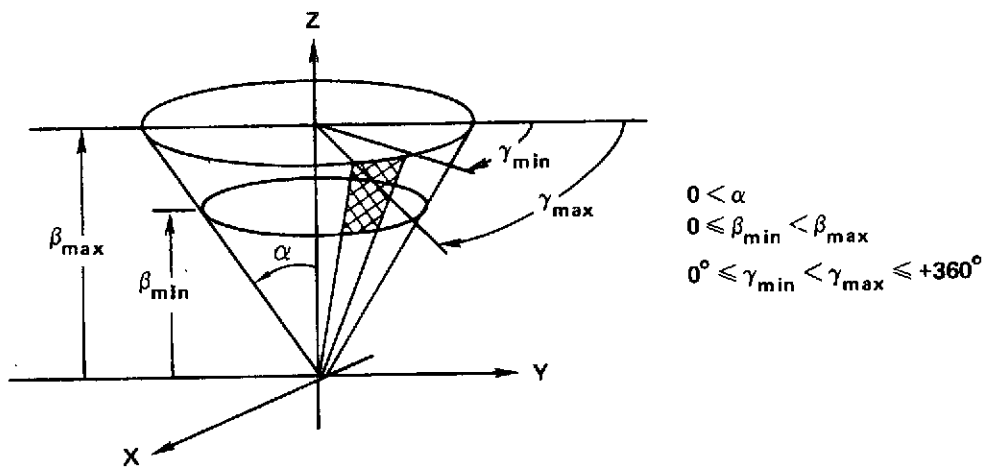


Figure 5.5 Surface Type ± 5 Cone

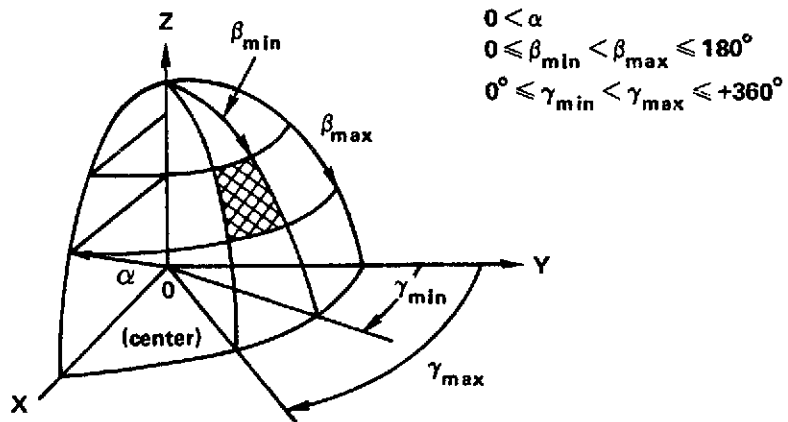


Figure 5.6. Surface Type ± 6 Sphere

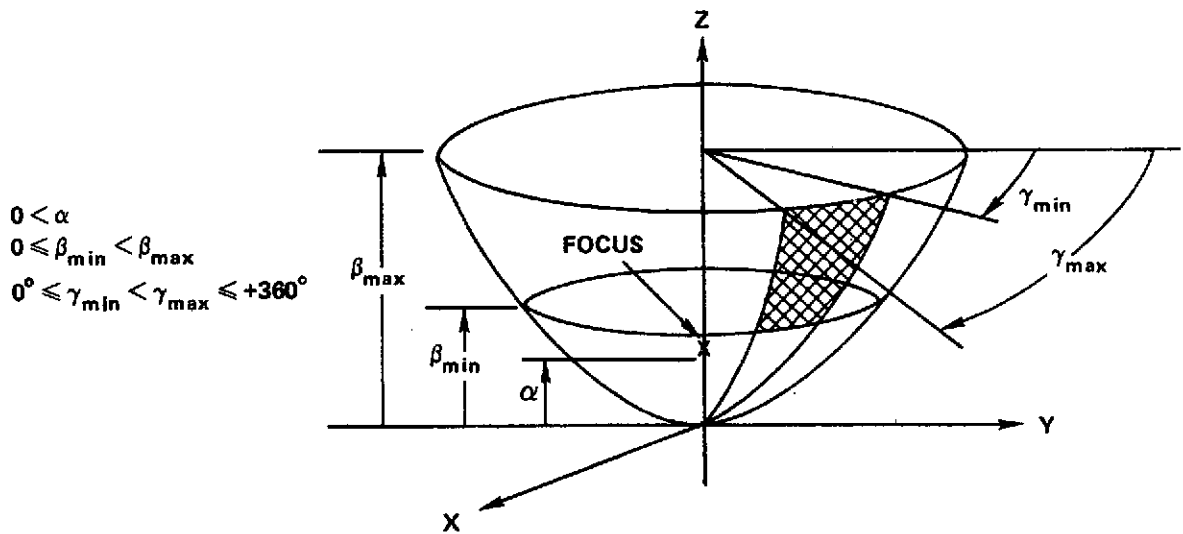


Figure 5.7. Surface Type ± 7 Circular Paraboloid

Each surface may be divided into NVB elements in the β direction and NVG elements in the γ direction. In turn each element may be divided into NB subelements in the β direction and NG subelements in the γ direction.

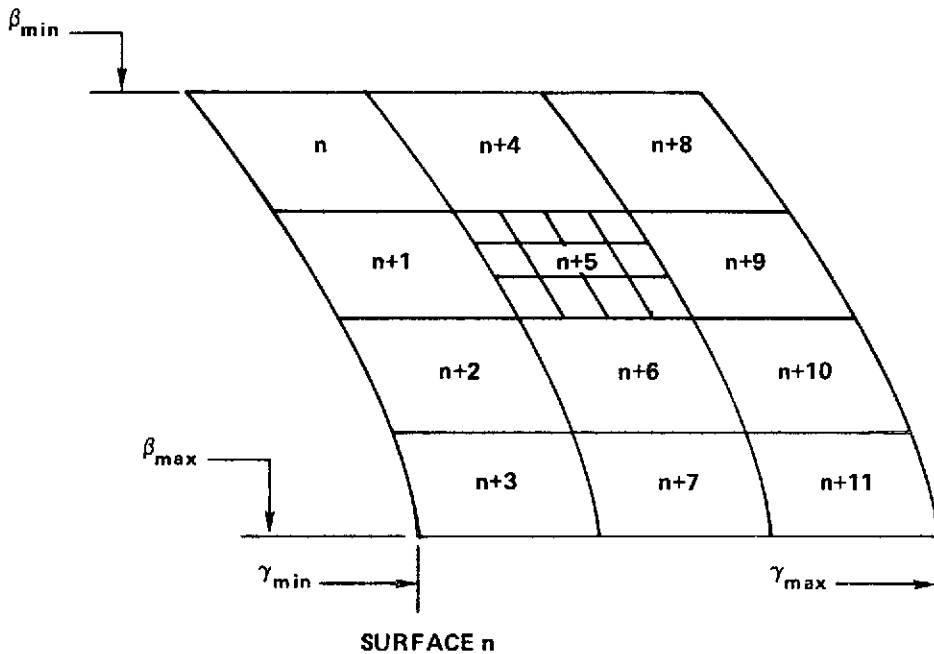


Figure 5.8. Four-By-Three Element Mesh with Three-By-Four Subelement Mesh

On element $n + 5$ of Figure 5.8 is an example of a three-by-four subelement mesh. The user may label the surface any number n , but the program will number the elements as indicated.

6. INPUT

6.1 ORGANIZATION

Several view problems may be run at one time. Each problem is referred to as a case. A typical input data deck would appear as shown in Figure 6.1.

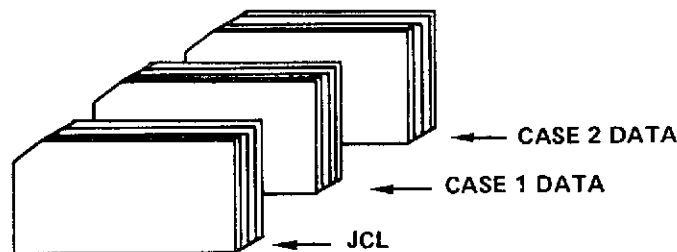


Figure 6.1. Input Data Deck

Any case must always begin with a title card, after which must come a case control card followed by a format-type card declaring NASTRAN input or RAVFAC input is to follow. If NASTRAN input is used, after the NASTRAN data must come an ENDDATA card (Figure 6.2). If RAVFAC data is used, the end of data is signified by inserting a card with -1 in columns 4 and 5 (Figure 6.3). If the user desires to run a problem using both formats, NASTRAN data must appear before the RAVFAC data (Figure 6.4). Every case must end with an ENDCASE card so that the program may distinguish between cases.

Figures 6.2 through 6.4 describe the only ways that the control cards comprising a case may be organized. Any data card may be punched on either the 026 or 029 key punch.

6.2 CONTROL CARDS

The control cards portion of a case is described in the following subparagraphs.

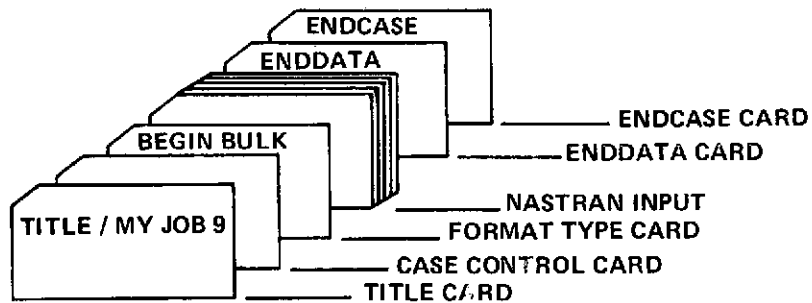


Figure 6.2. Organization of a Case Consisting Only of NASTRAN Data

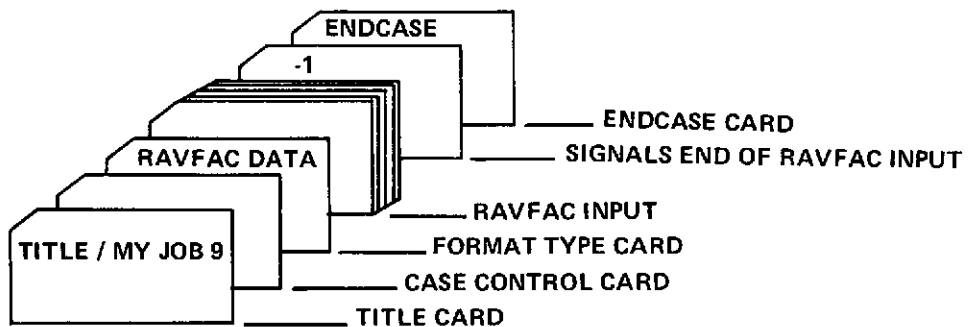


Figure 6.3. Organization of a Case Consisting Only of RAVFAC Data

6.2.1 Title Card

This is the first card in a case. It must have TITLE punched in columns 1 to 5. Beginning in column 9, the user may punch any 71 characters, and those characters are printed at the beginning of the output to identify the case. See sample output in appendices.

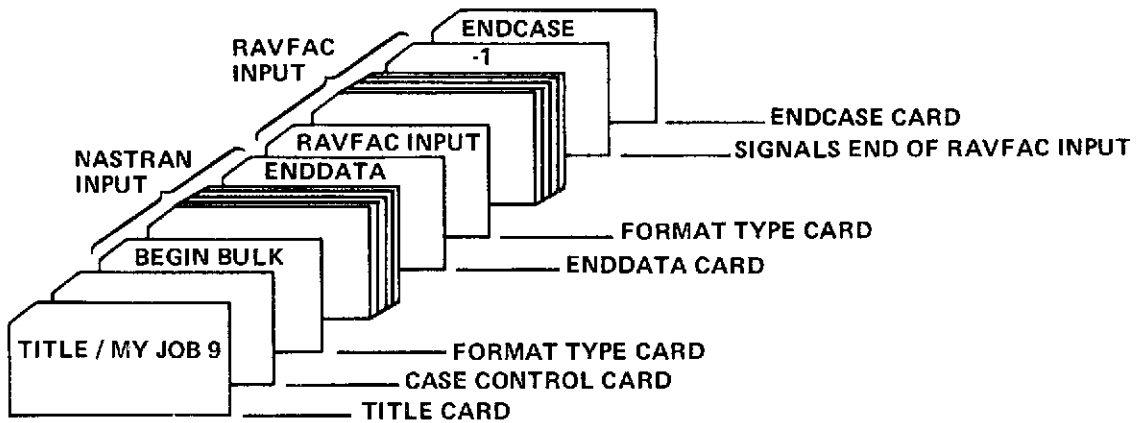


Figure 6.4. Organization of a Case Consisting of Both NASTRAN and RAVFAC Data

6.2.2 Case Control Card

This is the card which immediately follows the title card and is used to pass information to the program regarding restart, type of integration to be used, and various other features as described in the following format description. The format of this card is as follows:

- Columns 1 to 5 — User must punch CASE=. This identifies the card as a case control card.
- Columns 9 to 16 — Enter integer anywhere in columns 9 to 16, which is the same number of minutes CPU time shown on the JCL job card. Punch H if the time estimate on the job card is one-half minute. Defaults to 100,000, but will not affect results unless job times out, in which case all available output may not be printed out. If the proper CPU time is punched in this field and the job times out then all computed view factors will be printed out.
- Columns 17 to 24 — An integer less than or equal to zero in these columns indicates that no restart tape is to be

produced. Any integer greater than zero indicates that a restart tape will be produced on unit 2. In this case the user must supply the computer operators with a tape and specify the number of the tape on the EXEC card in the JCL (paragraph 6.5).

Columns 25 to 32—If any integer greater than zero appears in columns, then all view factor computations are made using the finite difference technique. An integer less than zero causes contour integration to be used in computing all view factors. If zero is punched in columns 25 through 32, or if they are left blank, the program selects which of the two methods it will use between two elements, based on the parameter RMAX specified in columns 49 through 56. Consider the situation in Figure 6.5.

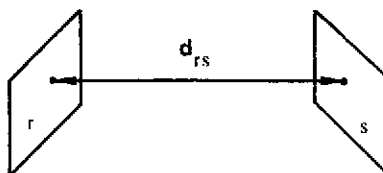


Figure 6.5. Choosing a View Factor Computation Method

The finite difference method becomes inaccurate when $d_{rs}^2 > A_s/0.1$ where A_s is the area of a subelement s , and d_{rs} is the distance between two subelements for which view factors are being computed. The solution to this problem is not to use contour integration for all calculations because it requires more computer time. By specifying a zero in any one of columns 25 through 32, the program selects the most efficient techniques between two elements by checking whether:

$$\frac{A_s}{d_{rs}^2} > RMAX$$

where RMAX is specified in columns 49 through 56.

If this relation is satisfied, contour integration is used. If not, finite difference is used. This check requires more time than if the user had requested the finite difference technique, but less time than if contour integration was requested.

Columns 33 to 40—Any integer greater than zero causes the subelement mesh size for each element to be set to one-by-one, overriding information given on the input data. For large problems a short checkout run of the data validity should be made. Using this option to set a one-by-one subelement mesh for each element will allow for the problem to be processed in a relatively short time. A zero, blank, or negative integer tells the program that the mesh sizes will be specified on the input data. This should be used once the user is sure of the model.

Columns 41 to 48—Any integer less than zero indicates that all shading is to be neglected. This will save computer time, but the user must be sure that no shading occurs in the model. An integer greater than or equal to zero tells the program to consider shading, using the information which describes each surface in the input data. There the user can specify whether or not a surface can shade any other surfaces or be shaded by any other surfaces.

Columns 49 to 56—This is where the user specifies the floating point number RMAX. If the columns are left blank, RMAX is set to 0.1 by the program. A value of .01 will provide results as accurate as usually required. A value greater than .1 will probably produce view factors which are too large. If the user selects contour integration or finite difference integration in columns 17 to 24 rather than allowing the program to choose the method, then any number in these columns is ignored.

6.2.3 Format Type Card

This is the card which always appears after the case control card and sometimes after an ENDDATA card (see Figure 6.4). Its purpose is to identify the format type of the data which follows. Punch BEGIN BULK in columns 1 to 10 to indicate NASTRAN-type input will follow. Punch RAVFAC DATA in columns 1 to 11 to indicate that RAVFAC-type data will follow.

6.2.4 ENDDATA Card

This card immediately follows the last card input in NASTRAN format. It signifies the end of NASTRAN-formatted input for the case. The word ENDDATA is punched beginning in column 1.

6.2.5 -1 Card

This card immediately follows the last card input in RAVFAC format. It signifies the end of the RAVFAC-formatted input for the case. Punch -1 in columns 4 and 5.

6.2.6 ENDCASE Card

This card follows either an ENDDATA card or a -1 card. It signifies the end of the case. The word ENDCASE is punched beginning in column 1.

6.2.7 -1 Card (For Local Coordinate Systems)

This card follows the last RAVFAC local coordinate system card (paragraph 6.4.4). It is used only if RAVFAC local coordinate system cards are used. Punch -1 in columns 9 and 10.

6.3 NASTRAN-FORMATTED INPUT DATA CARDS

Once the program has read a format-type card declaring BEGIN BULK, it knows that it will be reading input data cards in a format which is also acceptable to the NASTRAN program. At this point, VIEW is similar to NASTRAN. Some of the read subroutines used in VIEW were taken directly from the NASTRAN program.

The NASTRAN input data cards may be arranged in any order, but processing time will be decreased if continuation cards follow directly behind the cards which they continue. The free field format feature is also available in VIEW. That is, integer, decimal, alphanumeric, and exponential numbers may be punched starting in any of the columns of fields 2 through 10.* Alphanumeric characters must be left-justified in field 1. The large field option available in NASTRAN for the input of double precision is not available in VIEW. Cards may be punched on an 026 or 029 keypunch. See paragraph 2.3.1 for further details.

Once VIEW realizes it is to read NASTRAN-formatted data, it will accept any type of card without causing an error until it reaches an ENDDATA card. However, VIEW will only use and print out the following types of cards:

- CHBDY
- PHBDY
- GRID
- CORD1R, S, C
- CORD2R, S, C
- GRDSET
- \$VIEW

NOTE

If any other types of cards are read a message is printed out on the last page of output stating:
X LOGICAL CARDS DETECTED OF TYPES NOT
RECOGNIZED BY VIEW.

*A NASTRAN input data card is divided into ten fields, each consisting of eight columns. A continuation to a card is organized in the same fashion and its fields are numbered 11 to 20.

6.3.1 CHBDY Input Data Card

This card defines an element, numbers it, references, and gives data concerning its shape and geometric location. Figure 6.6 is the format and an example of a CHBDY card.

1	2	3	4	5	6	7	8	9	10
CHBDY	EID	PID	TYPE	G1	G2	G3	G4	IVIEW	+abc
CHBDY	413	29	LINE	107	108			14	+BD27

1	2	3	4	5	6	7	8	9	10
+abc	GA1	GA2	GA3	GA4	V1	V2	V3		
+BD27					1.00	0.0	0.0		

Figure 6.6. Format and Example CHBDY Card

The contents of each CHBDY input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CHBDY	CHBDY should be punched left-justified to identify the card type.
2. EID	Element identification number (Integer > 0)
3. PID	Integer ≥ 0 or blank. Only if the element being specified is either a POINT or LINE element must the user reference a PHBDY card. The PHBDY card gives the area of a POINT element or the width of a LINE element.
4. TYPE	Defines the shape of the element. Punch one of the names POINT, LINE, REV, AREA3, AREA4.
5 to 8, G1, G2, G3, G4	Integer ≥ 0 or blank. Grid point identification numbers. The boundaries of the element will be determined by the grid points listed here.

<u>Field</u>	<u>Contents</u>
9. IVIEW	Integer >0. \$VIEW card identification number. The element will have the characteristics given by the \$VIEW card with this number.

This card must be continued only if it defines a POINT or LINE element. If one of these two elements is defined and the card is not continued, the element will be ignored by the program and will not appear in the view factor calculations. If the card is to be continued, the following fields must be filled in:

<u>Field</u>	<u>Contents</u>
10, 11. +abc	Field 10 of the CHBDY card (except column 73 which is not referenced) is used in conjunction with field 11 of a continuation card as an identifier, and hence must contain a unique entry. The continuation card must contain the symbol + in column 1, followed by the same seven characters that appeared in columns 74 to 80 of field 10 of the card that is being continued. See paragraph 2.3.1 for further details.
12 to 15.	Anything punched in these fields is ignored.
16 to 18. V1, V2, V3	In these fields, V1, V2, and V3 are the coordinates of the vector in the coordinate system referenced in field 7 of grid card G1. The direction of the vector orients the elements and indicates the active side as described on the next page.

The following are descriptions of the element types. As seen in Figure 6.7, the POINT element shape is a flat disk with the center at grid point G1. Its active side and orientation in space are given by the vector, \vec{V} , specified in fields 16, 17, and 18 of the CHBDY continuation card. The \vec{V} is given in the coordinate system referenced in field 7 of grid card G1. Specify the disk area by using a PHBDY card.

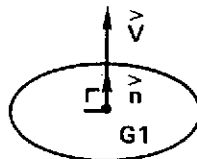


Figure 6.7. POINT Element—Flat Disk Shape

The LINE element shape, as seen in Figure 6.8, is a flat rectangular plate. Grid points G1 and G2 are located at the center of two opposite sides. The specified vector \vec{V} and the vector \vec{T} form a plane. The LINE element is oriented perpendicular to this plane. The active side of the element is given by the normalized vector $\vec{n} = (\vec{T} \times (\vec{V} \times \vec{T})) / |\vec{T} \times (\vec{V} \times \vec{T})|$. Again, \vec{V} is given in the coordinate system referenced in field 7 of grid card G1. Specify the width perpendicular to \vec{T} by using a PHBDY card.

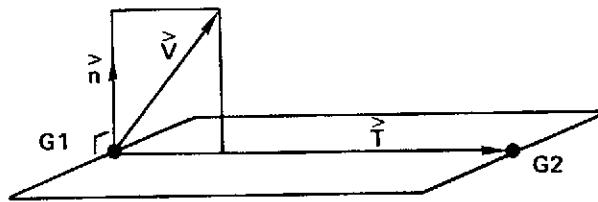


Figure 6.8. LINE Element—Flat Rectangular Plate

As seen in Figure 6.9, the REV element shape is a three-dimensional cone or cylinder. The grid points G1 and G2 must be in the X-Z plane of the basic coordinate system. A vector \vec{T} from G1 to G2 connects them. The element is generated by revolving \vec{T} about the Z axis 360 degrees. The active side is given by the normalized vector $\vec{n} = (\vec{e}_y \times \vec{T}) / |\vec{e}_y \times \vec{T}|$. There is no need to specify \vec{V} or a surface area.

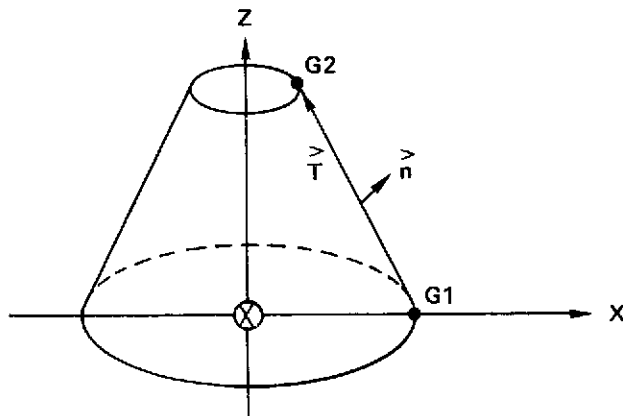


Figure 6.9. REV Element—Three-Dimensional Cone

The AREA3 element shape is a flat triangular plate, as shown in Figure 6.10. It is oriented in space by the locations of grid points G1, G2, and G3. Its active side is specified by the normalized vector $\vec{n} = (\vec{T}_{12} \times \vec{T}_{13}) / |\vec{T}_{12} \times \vec{T}_{13}|$ (that is, the right-hand rule is applied to G1, G2, G3). There is no need to specify \vec{V} or a surface area.

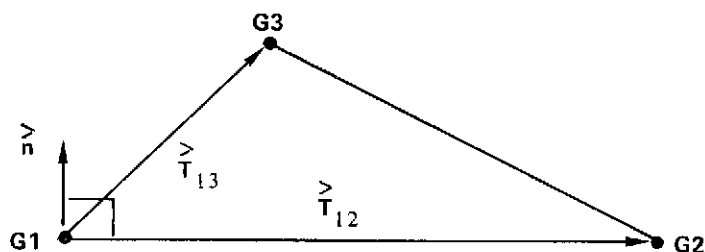


Figure 6.10. AREA3 Element—Flat Triangular Plate

As seen in Figure 6.11, the AREA4 element shape is a flat quadrilateral plate. It is oriented in space by the locations of G1, G2, G3, and G4. The grid points must all lie in one plane. They must be entered so that the lines $\overline{G1G2}$, $\overline{G2G3}$, $\overline{G3G4}$, and $\overline{G4G1}$ do not intersect except at grid points. The active side is determined in the same manner as for the AREA3 element. There is no need to specify \vec{V} or a surface area.

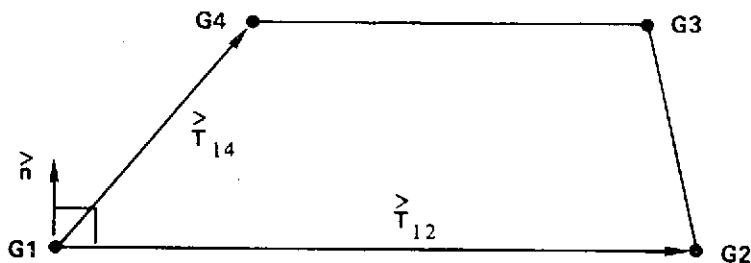


Figure 6.11. AREA4 Element—Flat Quadrilateral Plate

6.3.2 CORD1C Input Data Card

The CORD1C card defines a cylindrical coordinate system by reference to three grid points. These points must be defined in the basic coordinate system. The first point is the origin, the second lies on the Z-axis, and the third lies in the plane of the azimuthal origin. Figure 6.12 shows a cylindrical coordinate system and Figure 6.13 is the format and an example of a CORD1C card.

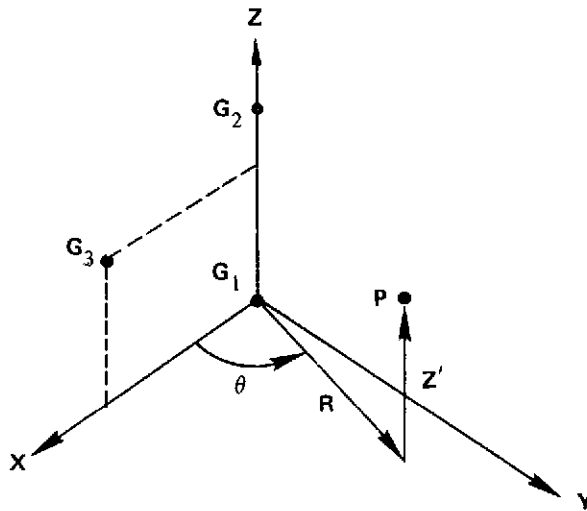


Figure 6.12. Cylindrical Coordinate System

1	2	3	4	5	6	7	8	9	10
CORD1C	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1C	3	16	32	19					

Figure 6.13. Format and Example CORD1C Card

The contents of each CORD1C input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD1C	Punch CORD1C beginning in column 1
2. CID	Coordinate system identification number (Integer > 0)
3 to 5. G1, G2, G3	Grid point identification numbers (Integer > 0); G1 ≠ G2 ≠ G3)

Coordinate system identification numbers on all CORD1C cards must be unique. The three points G1, G2, G3 must be noncollinear and in the basic coordinate system. The location of a grid point (P in Figure 6.12) in this coordinate system is given by R, θ , Z' where θ is measured in degrees. One or two coordinate systems may be defined on a single card.

6.3.3 CORD1R Input Data Card

The CORD1R card defines a rectangular coordinate system by reference to three grid points. These points must be defined in the basic coordinate system. The first point is the origin, the second lies anywhere on the +Z-axis, and the third lies anywhere in the X-Z plane. Figure 6.14 shows a rectangular coordinate system and Figure 6.15 is the format and an example of a CORD1R card.

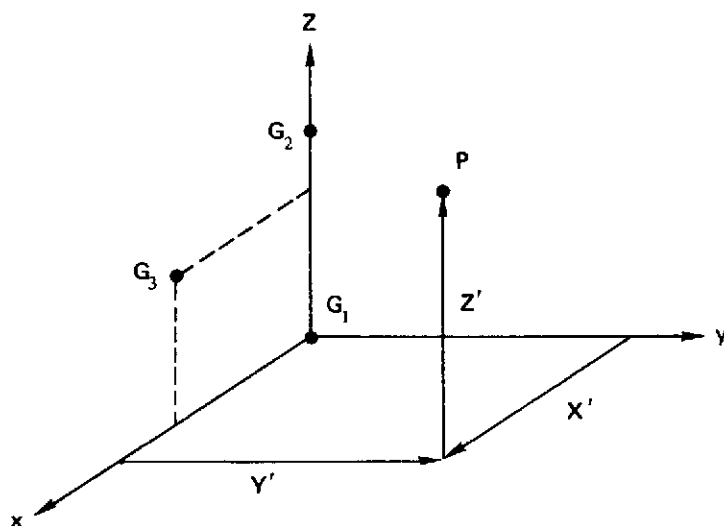


Figure 6.14. Rectangular Coordinate System

1	2	3	4	5	6	7	8	9	10
CORD1R	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1R	3	16	32	19					

Figure 6.15. Format and Example CORD1R Card

The contents of each CORD1R input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD1R	Punch CORD1R beginning in column 1
2. CID	Coordinate system identification number (Integer > 0)
3 to 5. G1, G2, G3	Grid point identification numbers (Integer > 0); G1 ≠ G2 ≠ G3)

Coordinate system identification numbers on all CORD1R cards must be unique. The three points G1, G2, G3 must be noncollinear and in the basic coordinate system. The location of a grid point (P in Figure 6.14) in this coordinate system is given by X' , Y' , Z' . One or two coordinate systems may be defined on a single CORD1R card.

6.3.4 CORD1S Input Data Card

The CORD1S card defines a spherical coordinate system by reference to three grid points. These points must be defined in the basic coordinate systems. The first point is the origin, the second lies on the +z-axis, and the third lies in the plane of the azimuthal origin. Figure 6.16 shows a spherical coordinate system and Figure 6.17 is the format and an example of a CORD1S card.

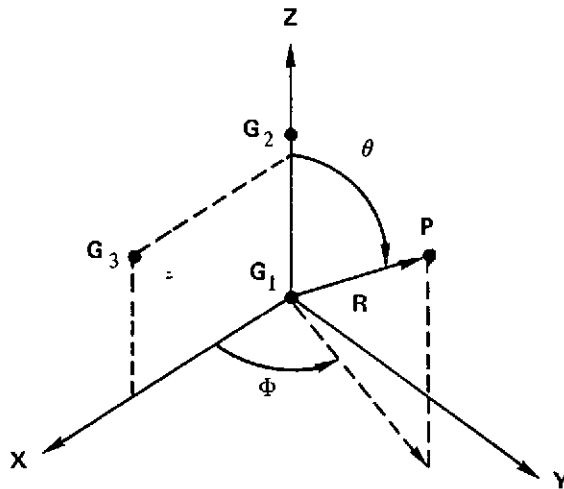


Figure 6.16. Spherical Coordinate System

1	2	3	4	5	6	7	8	9	10
CORD1S	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1S	3	16	32	19					

Figure 6.17. Format and Example CORD1S Card

The contents of each CORD1S input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD1S	Punch CORD1S beginning in column 1.
2. CID	Coordinate system identification number (Integer > 0)
3 to 5. G1, G2, G3	Grid point identification numbers (Integer > 0; G1 ≠ G2 ≠ G3)

Coordinate system identification numbers on all CORD1S cards must be unique. The three points G1, G2, G3 must be noncollinear and in the basic coordinate system. The location of a grid point (P in Figure 6.16) in this coordinate system is given by R, θ, ϕ where θ and ϕ are measured in degrees. One or two coordinate systems may be defined on a single CORD1S card.

6.3.5 CORD2C Input Data Card

The CORD2C card defines a cylindrical coordinate system by reference to the coordinates of three points. The first defines the origin, the second defines the direction of the +Z-axis, and the third lies in the plane of the azimuthal origin. Figure 6.18 shows a cylindrical coordinate system and Figure 6.19 is the format and an example of a CORD2C card.

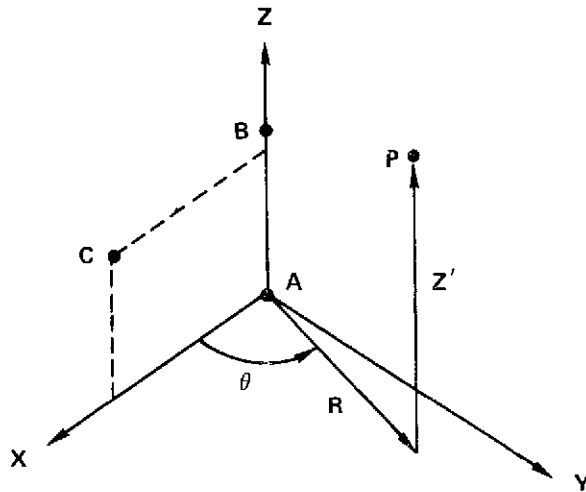


Figure 6.18. Cylindrical Coordinate System

1	2	3	4	5	6	7	8	9	10	
CORD2C	CID	RID	A1	A2	A3	B1	B2	B3	ABC	CARD 1
CORD2C	3		2.9	1.0	0.0	3.6	0.0	1.0	123	
+BC	C1	C2	C3							CARD 2
+23	5.2	1.0	-2.9							
11	11	13	14							

Figure 6.19. Format and Example of CORD2C Card

The contents of each CORD2C input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD2C	Punch CORD2C beginning in column 1
2. CID	Coordinate system identification number (Integer > 0)
3. RID	Blank
4-6. A1, A2, A3	Coordinates of three points in the basic coordinate system (Real)
7-9. B1, B2, B3	
12-14. C1, C2, C3	
10, 11. +ABC	Unique continuation symbol. (See 2.3.1)

The continuation card must be present. The three points (A1, A2, A3), (B1, B2, B3), (C1, C2, C3) must be unique and noncollinear. Noncollinearity is checked by the geometry processor. The points must be in the basic coordinate system. Coordinate system identification numbers on all CORD2C cards must be unique. The location of a grid point (P in Figure 6.18) in this coordinate system is given by R, θ , Z' where θ is measured in degrees.

6.3.6 CORD2R Input Data Card

The CORD2R card defines a rectangular coordinate system by reference to the coordinates of three points. The first point defines the origin, the second defines the direction of the +Z-axis, and the third defines a vector which, with the Z-axis, defines the X-Z plane. Figure 6.20 shows a rectangular coordinate system and Figure 6.21 is the format and an example of a CORD2R card.

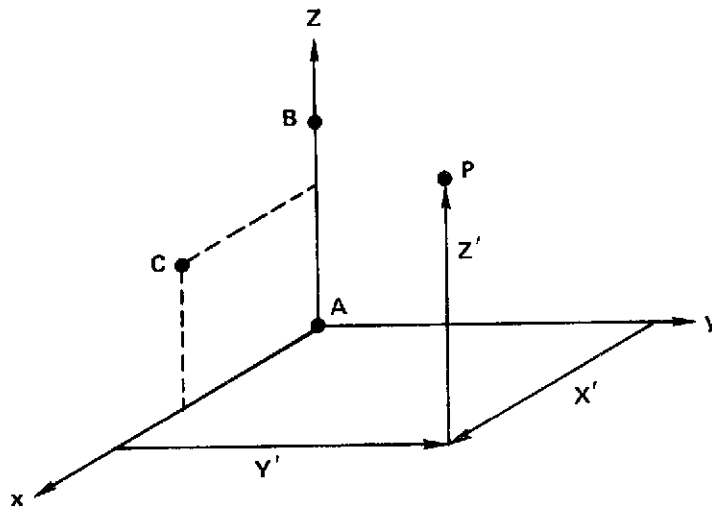


Figure 6.20. Rectangular Coordinate System

1	2	3	4	5	6	7	8	9	10	
CORD2R	CID	RID	A1	A2	A3	B1	B2	B3	ABC	CARD 1
CORD2R	3		-2.9	1.0	0.0	3.6	0.0	1.0	123	
+BC	C1	C2	C3							CARD 2
+23	5.2	1.0	-2.0							
11	12	13	14							

Figure 6.21. Format and Example CORD2R Card

The contents of each CORD2R input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD2R	Punch CORD2R beginning in column 1
2. CID	Coordinate system identification number (Integer > 0)
3. RID	Blank
4-6. A1, A2, A3	Coordinates of three points in the basic coordinate system (Real)
7-9. B1, B2, B3	
12-14. C1, C2, C3	
10, 11. +ABC	Unique continuation symbol. (See 2.3.1)

The continuation card must be present. The three points (A1, A2, A3), (B1, B2, B3), (C1, C2, C3) must be unique and noncollinear; noncollinearity is checked by the geometry processor. The points must be in the basic coordinate system. Coordinate system identification numbers on all CORD2R cards must be unique. The location of a grid point (P in Figure 6.20) in this coordinate system is given by X' , Y' , Z' .

6.3.7 CORD2S Input Data Card

The CORD2S card defines a spherical coordinate system by reference to the coordinates of three points. The first point defines the origin, the second defines the direction of the +Z-axis, and the third lies in the plane of the azimuthal origin. Figure 6.22 shows a spherical coordinate system and Figure 6.23 is the format and an example of a CORD2S card.

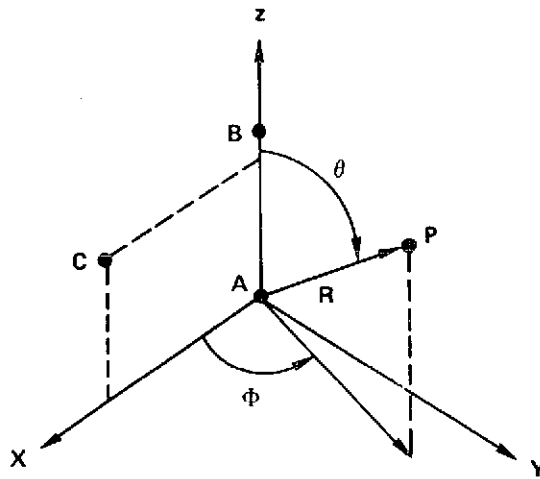


Figure 6.22. Spherical Coordinate System

1	2	3	4	5	6	7	8	9	10	
CORD2S	CID	RID	A1	A2	A3	B1	B2	B3	ABC	CARD 1
CORD2S	3		2.9	1.0	0.0	3.6	0.0	1.0	123	
+BC	C1	C2	C3							CARD 2
+23	5.2	1.0	-2.9							
11	12	13	14							

Figure 6.23. Format and Example CORD2S Card

The contents of each CORD2S input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. CORD2S	Punch CORD2S beginning in column 1
2. CID	Coordinate system identification number (Integer > 0)
3. RID	Blank
4-6. A1, A2, A3	Coordinates of three points in the basic coordinate system (Real)
7-9. B1, B2, B3	
12-14. C1, C2, C3	
10, 11. +ABC	Unique continuation symbol. (See 2.3.1)

The continuation card must be present. The three points (A1, A2, A3), (B1, B2, B3), (C1, C2, C3) must be unique and noncollinear; noncollinearity is checked by the geometry processor. Coordinate system identification numbers on all CORD2S cards must be unique. The location of a grid point (P in Figure 6.22) in this coordinate system is given by R , θ , ϕ where θ and ϕ are measured in degrees.

6.3.8 GRID Input Data Card

GRID cards define the location of a geometric grid point of the model. Figure 6.24 is the format and an example of a GRID card.

1	2	3	4	5	6	7	8	9	10
GRID	ID	CP	X1	X2	X3	CD	PS		
GRID	107	6	1.0	0.0	-4.5				

Figure 6.24. Format and Example GRID Card

The contents of each GRID input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. GRID	GRID should be punched left-justified to identify the card type.
2. ID	Grid point identification number (Integer >0).
3. CP	Identification number of the coordinate system in which the location of the grid point is defined (Integer >0).
4, 5, 6. X1, X2, X3	Location of the grid point in coordinate system CP (any real number).
7. CD	Identification number of a coordinate system in which the V vector on a CHBDY card is defined. Used only for Point and LINE elements (Integer >0).
8 to 10.	Any numbers punched in these fields are ignored by VIEW.

The meaning of X1, X2, and X3 depend on the type of coordinate system defined by CORD card CP. The system may be rectangular, cylindrical, or spherical coordinates.

6.3.9 GRDSET Input Data Card

This card defines default options for fields 3, 7, and 8 of all grid cards. Figure 6.25 is the format and an example of a GRDSET card.

	1	2	3	4	5	6	7	8	9	10
GRDSET			CP				CP	PS		
GRDSET			16				32			

Figure 6.25. Format and Example GRDSET Card

The contents of each GRDSET input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. GRDSET	GRDSET should be punched left-justified to identify the card type.
3. CP	Identification number of the coordinate system in which the location of the grid point is defined (Integer >0).
7. CD	Identification number of a coordinate system in which the V vector on a CHBDY card is defined (Integer >0).
8. PS	Not used by VIEW.

The contents of fields 3, 7 and 8 of this card are assumed for the corresponding fields of any grid card whose fields 3, 7 and 8 are blank. Only one GRDSET card may be used for each case.

6.3.10 PHBDY Input Data Card

The PHBDY card defines properties of elements specified on CHBDY cards, and is only needed for POINT and LINE elements. Figure 6.26 is the format and an example of a PHBDY card.

1	2	3	4	5	6	7	8	9	10
PHBDY	PID	MID	AF	E	ALPHA	R1	R2		
PHBDY	29		300.8						

Figure 6.26. Format and Example PHBDY Card

The contents of each PHBDY input data card are as follows:

<u>Field</u>	<u>Contents</u>
1. PHBDY	PHBDY should be punched left-justified to identify the card type.
2. PID	Property identification number (Integer >0).
4. AF	Area if referenced by a CHBDY card defining a POINT element. Width if referenced by a CHBDY card defining a LINE element. (Real ≥ 0.0).

Note: Any data punched in fields 3 and 5 to 10 are ignored by VIEW.

6.3.11 \$VIEW Input Data Card

This card is used only by the VIEW program and is not recognized by the NASTRAN program. It is used to give element characteristics which are not found on the other NASTRAN cards. Figure 6.27 is the format and an example of a \$VIEW card.

1	2	3	4	5	6	7	8	9	10
\$VIEW	IVIEW	KSHD	KBSHD	NB	NG	DISLIN			
\$VIEW	14	0	1	5	7	-.25			

Figure 6.27. Format and Example \$VIEW Card

The contents of each \$VIEW input data card are as follows:

	<u>Field</u>	<u>Contents</u>
1.	\$VIEW	\$VIEW should be punched left-justified to identify the card type.
2.	IVIEW	VIEW card identification number (Integer >0).
3.	KSHD	Can shade flag (0 = NO, 1 = YES).
4.	KBSHD	Can be shaded flag (0 = NO, 1 = YES).
5.	NB	Subelement mesh size in the beta direction (Integer >0).
6.	NG	Subelement mesh size in the gamma direction (Integer >0).
7.	DISLIN	Displacement of a surface perpendicular to the active side of the surface. Used only for the <u>LINE</u> element (Real).

The shading flags should be used carefully. When the user can identify a surface which cannot cause shading between any other surfaces, he can save computer time by identifying it. If in doubt as to whether or not it can shade, it should be labeled "can shade." Similarly, the user can save computer time by

identifying surfaces which cannot be shaded. If in doubt, they should be labeled "can be shaded." If the user wants the program to do all the work, all surfaces must be labeled $KSHD = 1$, $KBSHD = 1$.

The $DISLIN$ field should be used only for $LINE$ elements. It locates the surface $DISLIN$ units perpendicular to the place where the active side of the surface is located by the $CHBDY$ card (Figure 6.28).

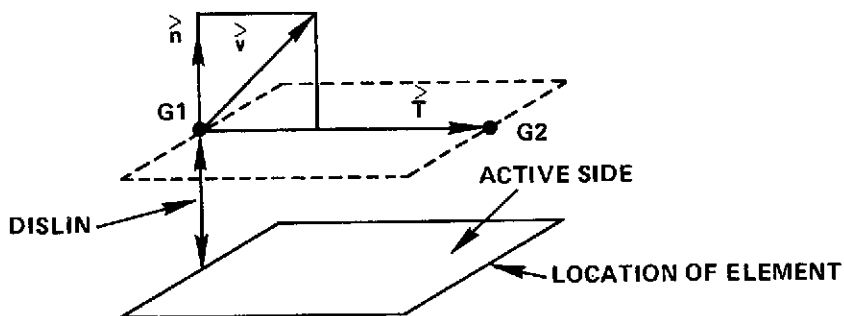


Figure 6.28. Location of $LINE$ Elements For $DISLIN < 0$

6.4 RAVFAC-FORMATTED INPUT DATA CARDS

The description of the geometry of a surface using RAVFAC-formatted input is totally different from that of an element using NASTRAN-formatted data. However, the information is essentially identical.

When using RAVFAC data, the user needs three distinct cards to describe each surface. These sets of three cards may be entered in any order; however, the ordering of the three cards within a set is required to follow a certain rule (refer to Figure 6.29).

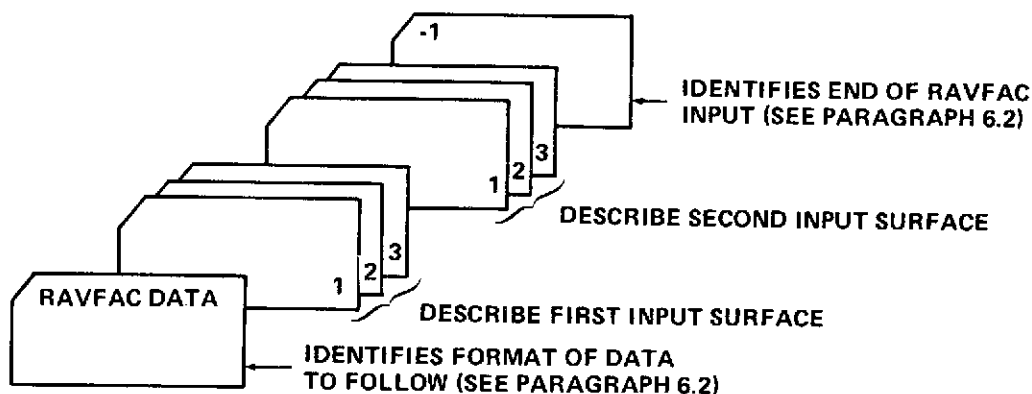


Figure 6.29. RAVFAC Input With No Local Coordinate Systems Specified

RAVFAC input also allows the user to specify an element relative to a local coordinate system. If one or more local coordinate systems are specified in the input, then one more card describing each one of these systems must be input. These cards relate each local coordinate system to the basic coordinate system. In this case the group of cards in RAVFAC input would be arranged as seen in Figure 6.30, as opposed to the manner shown in Figure 6.29 where no local coordinate systems were specified.

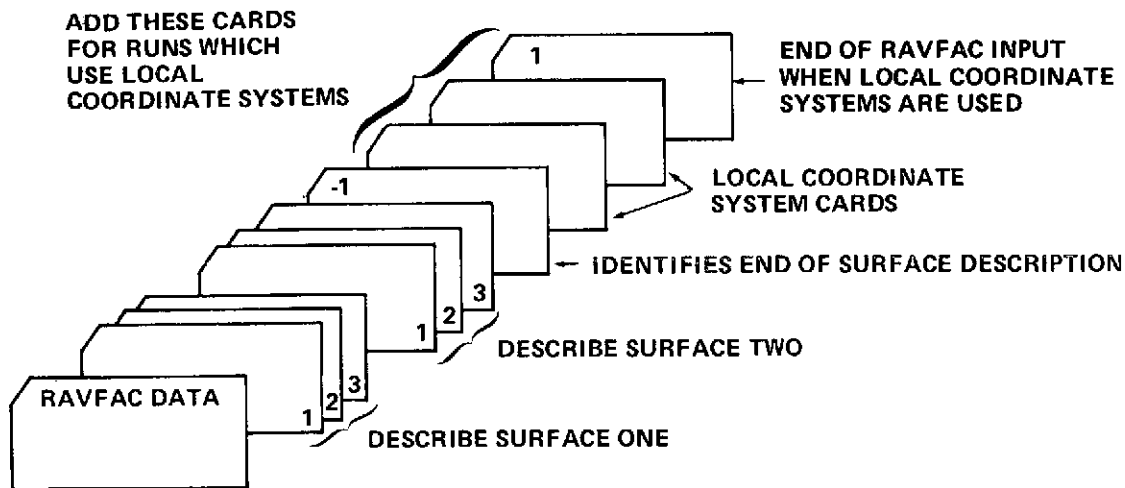


Figure 6.30. RAVFAC Input With Local Coordinate Systems Specified

After the last surface description card the user must insert a control card with -1 punched in columns 4 and 5 to identify the end of the surface descriptions. If the user describes one or more surfaces relative to a local coordinate system, then the program will expect a local coordinate system card for each different system referred to. After the last local coordinate system card, the user must insert another control card with -1 punched in columns 9 and 10.

6.4.1 RAVFAC Card Number 1—Format I5, 2I1, 43X, 7A4, A2

<u>Columns</u>	<u>Variable Name</u>	<u>Description</u>
1 to 5	ISF	Right-justified integer ≥ 0 . First element number on the surface. Each surface may be divided into several elements. If there are two or more elements on the surface, they will be numbered consecutively from ISF. The numbering scheme can be seen in Figure 5.8. Elements on different surfaces may be assigned the same number if desired. If ISF = 0 or is blank, the identification number will be one plus the identification number of the last element on the preceding surface.
6	KSHD	Integer ≥ 0 . If = 0 the program is told that each element of the surface cannot cause shading. If > 0 each element of the surface is flagged as "can shade."
7	KBSHD	Integer ≥ 0 . If = 0 the program is told that each element of the surface cannot be shaded. If > 0 each element of the surface is flagged as "can be shaded."
51 to 80	COMM	Any comment can be made in these columns to identify the surface.

6.4.2 RAVFAC Card Number 2—Format 5X, 5I5, 5E10.5

<u>Columns</u>	<u>Variable Name</u>	<u>Description</u>
9, 10	ILK	<p>Right-justified integer indicating the surface shape.</p> <p>= ±1 — Rectangle = ±5 — Cone = ±2 — Disk = ±6 — Sphere = ±3 — Trapezoid = ±7 — Circular = ±4 — Cylinder paraboloid</p> <p>See section 5 for figures of each surface. For the flat surfaces (±1, ±2, ±3), the positive value indicates the +z side is active while the negative value flags the -z side as active. For the surfaces of revolution (±4, ±5, ±6, ±7), the positive value flags the outside of the surface as active while the negative value flags the inside. If the user wants to specify both sides active he must input two separate surfaces, one looking each way.</p>
11 to 15	NVB	Right-justified integer > 0. Number of elements in the β direction.
16 to 20	NVG	Right-justified integer > 0. Number of elements in the γ direction.
21 to 25	NB	Right-justified integer > 0. Number of subelements per element in the β direction.
26 to 30	NG	Right-justified integer > 0. Number of subelements per element in the γ direction.
31 to 40	A	Surface dimension α (see figures in Section 5).
41 to 50	BMIN	Surface dimension β_{\min} (see figures in Section 5).
51 to 60	BMAX	Surface dimension β_{\max} (see figures in Section 5).
61 to 70	GMIN	Surface dimension γ_{\min} (see figures in Section 5).
71 to 80	GMAX	Surface dimension γ_{\max} (see figures in Section 5).

6.4.3 RAVFAC Card Number 3--Format 5X, I5, 10X, 6E10.5

<u>Columns</u>	<u>Variable Name</u>	<u>Description</u>
6 to 10	NCS	Right-justified integer. If $NCS > 0$ then the surface coordinate system is related to a local coordinate system numbered NCS. (Described on later cards.) If $NCS \leq 0$, then the surface coordinate system is related to the basic coordinate system (see Section 3.2).
21 to 30	RX	Distance of the origin of the surface coordinate system to the origin of the basic coordinate system (or local coordinate system if $NCS > 0$) measured along the x-axis of the basic coordinate system (or local coordinate system).
31-40	RY	Same as RX except that the distance is measured along the y-axis.
41-50	RZ	Same as RX except that the distance is measured along the z-axis.
51 to 60	PHI	Coordinate rotation angle, ϕ , in degrees, for rotating the basic coordinate system (or the local coordinate system if $NCS > 0$) into the surface coordinate system. Rotate Y toward X about Z.
61 to 70	PSI	Same as PHI but rotates X toward Z about Y (given in degrees).
71 to 80	OMEGA	Same as PHI but rotates Y toward Z about X (given in degrees).

Note that the basic coordinate system (or local coordinate system) is rotated into the surface coordinate system. The rotations must be made in the order ϕ then ψ then ω (Figures 6.31 and 6.32).

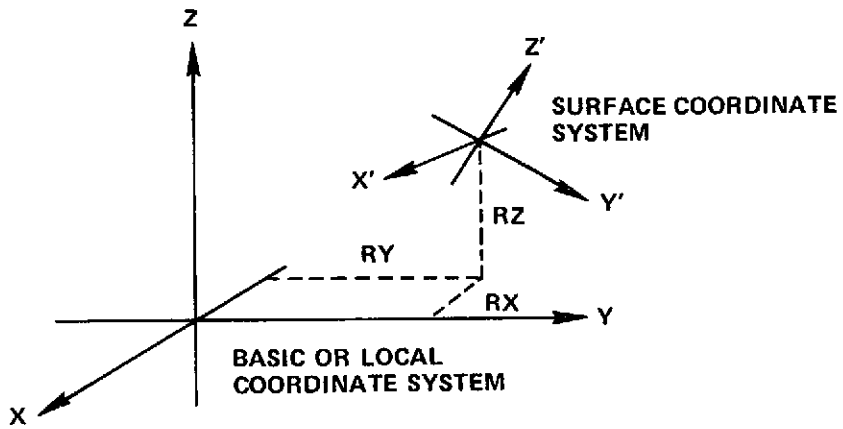


Figure 6.31. Basic Coordinate System Rotation

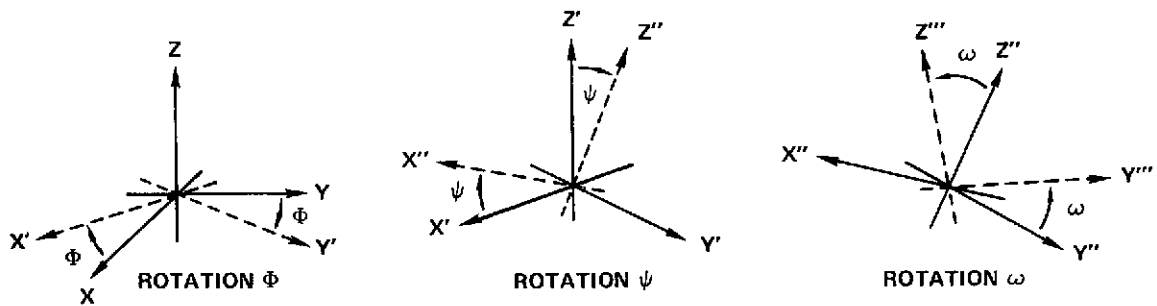


Figure 6.32. Rotations Order

6.4.4 Local Coordinate System Card—Format 5X, I5, 10X, 6E10.5

<u>Columns</u>	<u>Variable Name</u>	<u>Description</u>
6 to 10	ICS	Right-justified integer > 0. Local coordinate system number.
21 to 30	RXR	Distance of the origin of the local coordinate system to the origin of the basic coordinate system measured along the x-axis of the basic coordinate system.
31 to 40	RYR	Same as RXR but measured along the y-axis.
41 to 50	RZR	Same as RXR but measured along the z-axis.
51 to 60	PHIR	Coordinate rotation angle, ϕ , in degrees for rotating the basic coordinate system into the local coordinate system. Rotates Y toward X about Z.
61 to 70	PSIR	Same as PHIR but rotates X toward Z about Y.
71 to 80	OMEGAR	Same as PHIR but rotates Y toward Z about X.

Rotation of the basic coordinate system into the local coordinate system must be carried out in the order ϕ , ψ , ω .

6.5 360/95 JOB CONTROL CARDS

The program delivery tape contains a copy of the source deck, a load module, and a copy of the job control language (JCL) necessary to run the program. At the Goddard Space Flight Center where the program was developed, the JCL has been placed in the 360/95 procedure library, so that only the JOB and EXEC control cards need appear before the data.

For jobs where restart capability is not required, the EXEC card should be punched as

```
//bEXECbVIEW,REGION=XXXK
```

where b denotes a blank and XXX is a three-digit integer greater than 115, defining the requested core space. The amount of the request is discussed later in this section.

For jobs where restart capability is desired, the EXEC card should be punched as

```
//bEXECbVIEW,REGION=XXXK,TAPESER=YYYYYY,TAPEDSN=RESTART
```

The XXX explained previously must be identical to the region request of the run which generated the tape. YYYYYY is the tape number of the restart tape that the user will create. (Usually a blank tape is submitted with the job.)

If punched output is desired, the following JCL card must be inserted after the EXEC card:

```
//FT07F001bDDbDSN=&&CARDS,SYSOUT=B
```

6.5.1 Region Request

The following formula may be used as a guide for estimating the region request made on the EXEC card for a given case:

$$\begin{aligned} X &= 218 X \text{ (Number of elements in case) } + 115K \\ Y &= 48 X \text{ (Number of subelements in case) } + 115 X \text{ (number of} \\ &\quad \text{elements in case) } + 115K \\ Z &= 55 X \text{ (Number of grid points in case) } + 115K \\ \text{Request} &= \max (X, Y, Z) \end{aligned}$$

If several cases are being run at the same time, the user must make the region request sufficiently large to handle only the worst case, that is, the one which requires the most core space.

6.5.2 Time Estimation

There are several factors which determine the time necessary to process a case. Among these are:

- Method of integration used (see description of CASE control card columns 25 to 32).
- Use of shading flags (see description of columns 41 to 48 of the CASE control card, and description of the NASTRAN-type input data card \$VIEW, and RAVFAC-type input card 1).
- Subelement mesh sizes specified for each element (the overall number of subelements determines the number of view-factor calculations).
- Ordering of input data. The user can save time by ordering input so that the elements which cause the most shading appear first in the case data. Also, time will be saved if NASTRAN-type data continuation cards appear directly after the cards which they continue.

Since so many variables are involved in computing view factors, it is impossible to give the reader any accurate guidelines on estimating central processing unit (CPU) and input-output (IO) time requests. However, while the user is gaining experience, the use of the program's restart capability will ensure that runs which may time-out are not wasted effort.

A few examples of runs made at GSFC may serve as an indication of time requirements (refer to Table 6.1).

Table 6.1
Sample Runs/Times

Sub-Elements	Shading in Problem?	CPU Seconds GSFC 360/95 Slow Core	Comments
914	YES	994	Program asked to figure which elements shade
969	YES	234	
1187	YES	781	
721	YES	38	
504	YES	44	
1448	NO	58	Program told that there is no shading
1000	NO	62	
624	NO	12	
480	NO	9	
150	NO	3	

SECTION 7

OUTPUT

I

7. OUTPUT

The first page of output contains the case number, date and initiation time of run, the problem title from the case title card, an echo of the case control card, and a description of its variables. If the output is produced from a restart case the first page will be as described, while the next page will echo the case control card submitted with the restart tape (see Section 9).

A semisorted NASTRAN bulk data echo will begin on the next page if there is any NASTRAN-type formatted input. At its completion there will follow an input summary indicating the number of each of the card types (CORD1, CORD2, CHBDY, GRID and so on). Warning messages may also appear followed by a summary of any RAVFAC-type formatted data which was read.

The computed view factors are printed out on an element-to-element basis, along with the areas of each element and the sum of all the view factors seeing each element.

A summary of important program statistics is printed out. First, the user is told if the program was terminated normally or abnormally. In the latter case error messages appear. If the program terminated normally, the user is then informed of the CPU time used (in seconds) to process the input data, and the time used to compute view factors. Then the user is informed as to how many view factor computations were made by the contour integral techniques and how many were made by the finite difference technique. The user is next informed as to the number of elements, subelements, and grid points that the data generated, along with the maximum number of elements, subelements, and grid points that the region request will allow. Finally, the user is informed as to the number of bytes of core not used because of an excessive region request. Hence, if the problem should have to be rerun, or if another problem of similar size must be run, the region request can be reduced to the minimal necessary size.

The user can call for the program to produce punch output in the form of RADLST and RADMTX cards. The manner of producing this output is described in paragraph 6.5.

7.1 RADLST OUTPUT DATA CARD

This card contains a list of element identification numbers given in the same order as the columns of the RADMTX matrix. Figure 7.1 is the format and an example of a RADLST card.

	1	2	3	4	5	6	7	8	9	10
RADLST	EID1	EID2	EID3	EID4	EID5	EID6	EID7	EID8	abc	
RADLST	10	20	30	50	31	41	51	61	+1000000	
+bc	EID9	etc.								
+1000000	71									

Figure 7.1. Format and Example RADLST Card

The contents of each RADLST output data card are as follows:

<u>Field</u>	<u>Contents</u>
RADLST	The word RADLST is punched beginning in column 1.
EID _i	The element identification numbers of the elements, given in the order that they appear in the RADMTX matrix (paragraph 7.2)
abc	Sequential continuation numbers are punched out beginning with +1000000.

7.2 RADMTX OUTPUT DATA CARD

This card contains numbers, described in Figure 7.2 and the following paragraph, so as to form one column of a matrix of radiation exchange coefficients. Figure 7.2 is the format and an example of a RADMTX card.

	1	2	3	4	5	6	7	8	9	10
RADMTX	INDEX	$R_{i,i}$	$R_{i+1,i}$	$R_{i+2,i}$	$R_{i+3,i}$	$R_{i+4,i}$	$R_{i+5,i}$	$R_{i+6,i}$	abc	
RADMTX	3	0.	9.3	17.2	16.1	.1	0.	6.2	+5000000	
+bc	$R_{i+7,i}$	etc.								
+5000000	6.2									

Figure 7.2. Format and example RADMTX Card

The contents of each RADMTX output data card are as follows:

<u>Field</u>	<u>Contents</u>
RADMTX	The word RADMTX is punched beginning in column 1.
INDEX	The column number i of the matrix (Integer > 0).
R _{i+k, i}	The matrix values (Real), starting on the diagonal, continuing down the column. K = 0, 1, 2, ..., n - i. These values are the view factors from element i + k to element i multiplied by the area of element i + k.
abc	Sequential continuation numbers are punched out beginning with +5000000.

The INDEX numbers go from 1 through NA, where NA is the number of radiating elements. Since the radiation exchange coefficient matrix is symmetric, only the lower triangle is output. Column 1 is associated with the element first listed on the RADLST card. Column 2 is for the next, and so on.

$$\rho_i = \sum_{j=1}^{NA} R_{ij} q_j$$

$$\rho_i = \text{total irradiation at } i$$

$$q_j = \text{radiosity per unit area at } j$$

$$R_{ij} = \text{view factor from element } i \text{ to element } j \text{ multiplied by the area of element } i.$$

The following three RADMTX and one RADLST card would form the lower triangular portion of a symmetric matrix as shown:

RADMTX	1	0.0	8.1	2.2
RADMTX	2	3.2	1.7	
RADMTX	3	12.2		
RADLST	10	50	33	

$$R = \begin{bmatrix} 0.0 & 0 & 0 \\ 8.1 & 3.2 & 0 \\ 2.2 & 1.7 & 12.2 \end{bmatrix}$$

The first column describes the radiation exchange coefficients of element 10. The second column describes the radiation exchange coefficients of element 50, and the third column, the radiation exchange coefficient of element 33.

8. ERROR AND WARNING MESSAGES

Two types of messages may be printed out by the program at any point of the output. These are fatal errors and warning messages. A fatal error message terminates execution of the case being run. The program will usually try to run subsequent cases if they are present in the data deck. Warning messages apprise the user of a possible error in the data, but the program, being unsure of the user's intent, will continue processing.

In all cases, it has been attempted to make the messages sufficiently clear to explain the problem which caused them. However, since the explanations may be clear to the programmer, but may not be clear to the user, this section will expand on these explanations and/or refer the reader to the appropriate portions of this document.

8.1 EXPLANATIONS OF ERROR MESSAGES

<u>Fatal Error Number</u>	<u>Explanations</u>
1	See Section 4 and paragraph 6.5.
2	The CHBDY card defines a line element. See Figure 6.8 and associated comments.
3	Either the \$VIEW card number in field 9 of the CHBDY card is mispunched, or the \$VIEW card is mispunched (field 1 or 2) or misplaced.
5	Elements must be given positive areas.
6	Coincident grid points will not properly define the element shape specified in field 4 of the CHBDY card. See Figures 6.7 through 6.11 and associated comments.
7	See Figure 6.11 and associated comments.
8	Fields 6, 7, and 8 of the continuation card of the CHBDY card were all found to be zero. Thus, a vector $V = (0, 0, 0)$ was defined. The element type (either POINT or LINE) defined in field 4 of the CHBDY card requires V to be nonzero. See Figures 6.7 and 6.8 and associated comments.

Fatal
Error
Number

Explanations

- 9 Either the GRID card number on the CHBDY card is mispunched or the GRID card is mispunched (field 1 or 2) or misplaced.
- 10 All four grid points of AREA4 element must be coplanar. See Figure 6.11 and associated comments.
- 11 The first three grid points, having been found collinear, do not properly define an AREA3 or AREA4 element. See Figures 6.10 and 6.11 and associated comments.
- 12 The element type defined in the CHBDY card (either POINT or LINE) must reference a PHBDY card. The PHBDY card may have been mispunched or misplaced, or possibly the number in field 3 of the CHBDY card was mispunched or missing.
- 13 The CHBDY card does not define a REV element properly. See Figure 6.9 and associated comments.
- 14 The CHBDY card defines either a POINT or LINE element. To either element, a vector V must be associated. The vector V is in the coordinate system defined in field 7 of the GRID card G1 (specified in field 5 of the CHBDY card). That coordinate system was not defined by some type of CORD card. It was either missing or mispunched in fields 1 or 2.
- 15 CHBDY cards defining POINT or LINE elements must have PHBDY cards associated to them. See description of CHBDY card, field 3, paragraph 6.3.1.
- 21 Field 11 of a necessary continuation card was either misplaced or mispunched, or possibly field 10 of the card to be continued was mispunched.
- 22 Field 3 of the CORD2 card must be blank! See description of CORD2 cards, paragraphs 6.3.5 through 6.3.7.
- 23 User does not have the minimum amount of data necessary to define one element by NASTRAN-type input data. Perhaps the ENDDATA card was misplaced.

Fatal
Error
Number

Explanations

- 24 There must be an ENDDATA card if the case has NASTRAN-type input data. See paragraph 6.1. No subsequent cases are processed.
- 25 Too many elements have been defined and core space has run out. Increase the region request or lower the number of elements in the model. See paragraph 6.5.
- 26 More than 40 CORD cards of either type are not permitted.
- 27 Field 4 was probably mispunched.
- 28 In field 1 of the CORD card, CORDXY must be punched where X = 1 or 2 and Y = R, S, or C. The card had X or Y mispunched.
- 29 Field 11 of each continuation card must be unique. The deck has at least two with the same number.
- 40 Either field 3 of the GRID card has been mispunched or the CORD card which was intended to be referenced has been mispunched (field 1 or 2) or misplaced.
- 50)
51 } Probable mispunch.
52)
53)
- 54 No embedded blanks are allowed in data.
- 55)
56 } Should not occur. Seek a programmer's assistance.
- 60 Check the region request on the generating run. Use the same request on the restart deck JCL. Resubmit.
- 61)
62 } { May have a bad tape. Try once more.
 } { If not successful, start over again.
- 70 The grid points used to define coordinate systems by means of CORD1 cards must be in the basic coordinate system.

Fatal
Error
Number

Explanations

- 80 Enough space was requested to load the program, but not enough to store any data. Increase the request. See paragraph 6.5.
- 81 Error should not occur. If it does, there is an error in subroutine GETMA. You will need a programmer's assistance. Save the run to show the programmer.
- 90 The RAVFAC surface shape that was specified in columns 9 and 10 of a RAVFAC number 2 card is undefined. It must be either ± 1 , ± 2 , ± 3 , ± 4 , ± 5 , ± 6 , ± 7 .
- 93 The region request was not large enough for all the elements requested. Increase the region request or decrease element usage. See paragraph 6.5.
- 100 See paragraph 6.1.
- 101 The region request was not large enough for all the elements or subelements requested. See paragraph 6.5.
- 102 }
110 } See paragraph 6.1.
111 }
- 112 See description of TITLE card, paragraph 6.2.
- 113 See description of CASE card, paragraph 6.2.
- 114 The CPU time request must be increased. Consider using restart capability next time. See paragraph 6.5.

8.2 EXPLANATIONS OF WARNING MESSAGES

Warning
Number

Explanations

- 4 The VIEW card has a blank or zero in fields 5 and/or 6. This does not properly define a mesh of subelements so the program assumes a one-by-one mesh ($NB = 1$ and $NG = 1$) and continues processing. Check that this did not affect the answers.

Warning
Number

Explanations

- 16 See paragraph 2.5.
- 30 The program gets tired after checking 500 out-of-order continuation cards for duplicate ID fields. It refuses to check any remaining continuation cards. The user should check.
- 31 See description of GRDSET card in paragraph 6.3.9.
- 32 If continuation cards do not appear directly behind the card they continue, then processing is slowed.
- 33 Some card types were mixed in the NASTRAN-type input data which were not used by VIEW. It is wise to check that these are not mispunched cards which were really intended for VIEW processing.
- 34 The NASTRAN-type input data element ELCYL is not recognized, by VIEW. Other element types will have to be used to approximate the ELCYL element.
- 35 In the CHBDY card, which specified a POINT or LINE element, the continuation field (10) was left blank. The program interprets this as implying that the user does not want this element to appear in his model for calculation of view factors.
- 91 The RAVFAC-type input card number 2 has a blank or zero in columns 11 to 15 and/or 16 to 20. This does not properly define a mesh of elements so the program assumes a one-by-one mesh (NVB and NVG = 1). Check that this did not affect the answers.
- 92 The RAVFAC-type input data card number 2 has a blank or zero in columns 21 to 25 and/or columns 26 to 30. This does not properly define a mesh of subelements so the program assumes a one-by-one mesh (NB and NG = 1). Check that this did not affect the answers.
- 103 No further cases are run.

9. RESTART

Restart is a program feature which saves completed work on a tape; should disaster occur, the user can resubmit his restart JCL with the restart tape and pick up where he left off.

For example, suppose a user has just put together a large model which he expects will take no longer than 30 CPU minutes and 10 I/O minutes to run. Assume that one of the following happens:

- The job really takes 31 CPU minutes to run.
- The job really takes 11 I/O minutes to run.
- The computer operator accidentally hits the emergency OFF switch.

Normally such events mean that whatever calculations have already been done are wasted because the whole job will have to be run over again. Restart solves this problem.

To create a restart tape during the original run the user should:

- Punch any positive integer anywhere in columns 17 to 24 of the case control card (see paragraph 6.2).
- Have the JCL EXEC card read

```
//bEXECbVIEW, REGION=XXXK, TAPESER=YYYYYY, TAPEDSN=RESTART
```

where b denotes a blank and XXX should be the region request and YYYYYY is the tape number (see paragraph 6.5).

To make a run from a restart tape the user should:

- Have the same EXEC card just described.
- Have as input data a single card with RESTART punched in columns 1 to 7, and anywhere in columns 9 to 16 an integer which is the same number as the CPU time estimate on the JOB card. If the time estimate on the job card is one-half minute, punch an H anywhere in columns 9 to 16.

- Submit the JCL, single data card, and the restart tape.

The data for more cases may be placed behind the previously described single data card.

APPENDIX A

SAMPLE PROBLEM USING ONLY NASTRAN-TYPE DATA

The physical structure to be modeled in this problem is shown in Figure A-1.

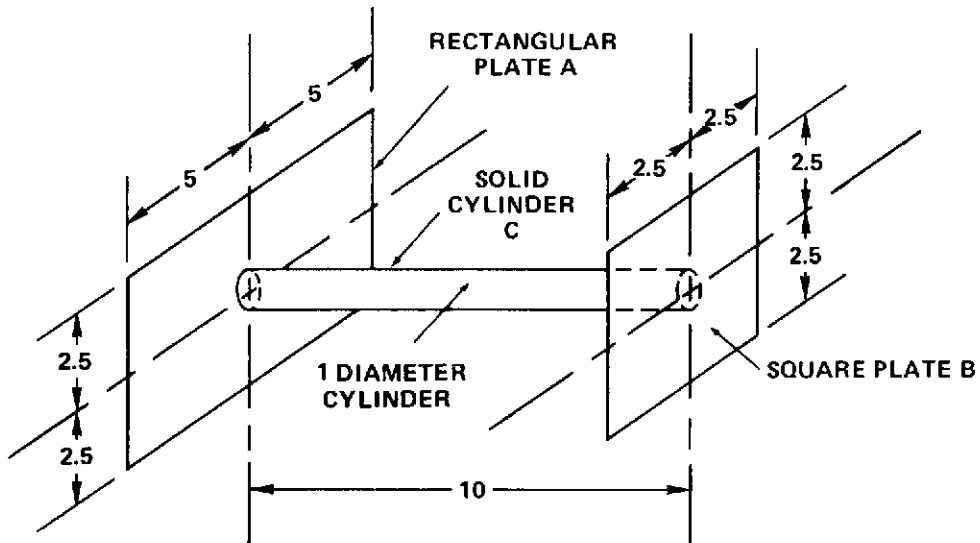


Figure A-1. Structure to Be Modeled

As shown in Figure A-1, nonzero view factors exist only between the two sides of plate A and plate B which face each other (call them face A and face B), and between the outside of the cylinder (face C) and faces A and B. Hence, the model, if desired, could consist of only five elements: two rectangular, one cylindrical, and two disks. The disks would cap the ends of the cylinder so that the two rectangles would not see each other through the cylinder.

Choosing the axis of the cylinder as the z-axis and the larger rectangle to be in the x-y plane, the element model in the basic coordinate system can be drawn as seen in Figure A-2.

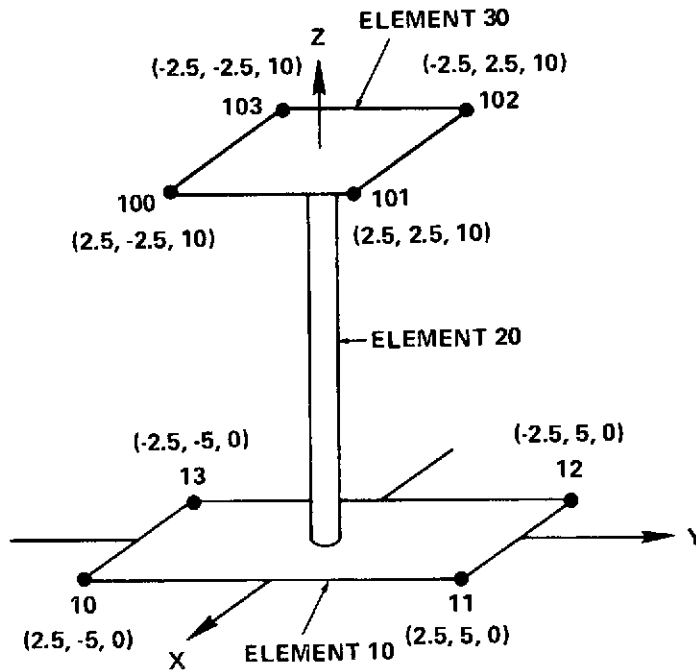


Figure A-2. Element Model in the Basic Coordinate System

The grid points are arbitrarily numbered as shown for the two rectangular plates. Figures A-3 and A-4 show more clearly the grid points used to define the cylinder and two disks.

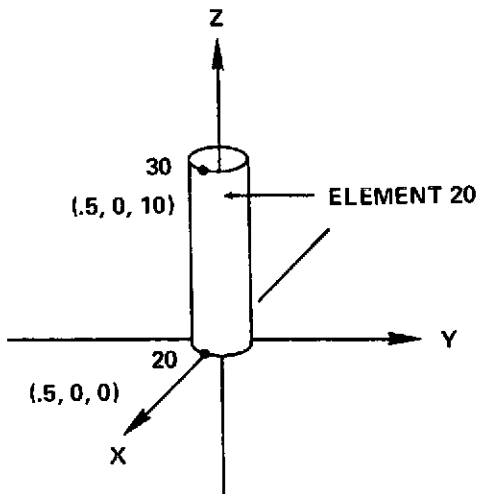


Figure A-3. Grid Points Used to Define Cylinder

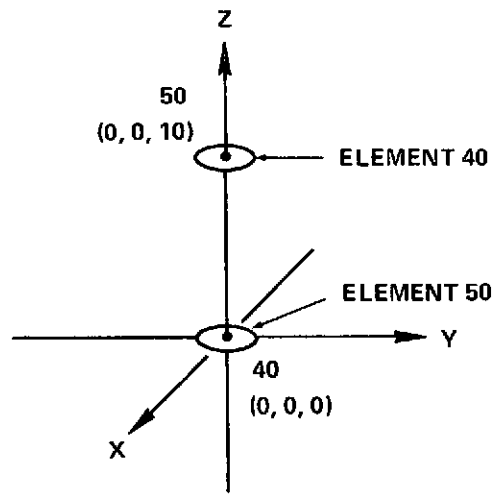


Figure A-4. Grid Points Used to Define Two Disks

Since shading is involved, a fine subelement mesh is needed on elements 10 and 30, and a less fine subelement mesh is needed on element 20. Since elements 40 and 50 are present for shading purposes only, we can specify a one-by-one mesh on each of these elements. Furthermore, we can specify that element 10 can be shaded but cannot cause shading. The same is true for element 30. However, element 20 can shade but cannot be shaded while 40 and 50 can shade and be shaded.

Figure A-5 is a listing of the input used to run this problem. It is followed by a copy of the program output.

```
//B9EFPT01 JOB (██████1831C,T,B00081,001H00),YYY,MSGLEVEL=(2,0)
// EXEC VIEW,REGION=175K
TITLE    *** SAMPLE PROBLEM NO. 1 ***
CASE=    1
BEGIN BULK
CHBDY    10      AREA4      10      11      12      13      50
CHBDY    20      REV        20      30
CHBDY    30      AREA4      100     103     102     101     60
CHBDY    40      10      POINT    50      0.0     0.0     1.     65     +C1
+C1
CHBDY    50      10      POINT    40      0.0     0.0     -1.0   65     +C2
+C2
$VIEW    50      0        1        25     15
$VIEW    55      1        0        10     20
$VIEW    60      0        1        15     15
$VIEW    65      1        1        5      5
PHBDY    10      .786
GRID     10      2.5     -5.0    0.0
GRID     11      2.5     5.0     0.0
GRID     12      -2.5    5.0     0.0
GRID     13      -2.5    -5.0    0.0
GRID     100     2.5     -2.5    10.0
GRID     101     2.5     2.5     10.0
GRID     102     -2.5    2.5     10.0
GRID     103     -2.5    -2.5    10.0
GRID     20      .5
GRID     30      .5      10.0
GRID     40
GRID     50      10.0
ENDDATA
ENDCASE
```

Figure A-5. Program Input for Sample Problem 1

DATE 7/26/73
TIME 14.44.40

* CASE 1 *

PROBLEM TITLE *** SAMPLE PROBLEM NO. 1 ***

CASE CONTROL CARD

COLUMNS	VARIABLE	VALUE	DESCRIPTION
9-16	IENDTM	1	ENTER CPU TIME USED ON JOB CARD (IN MINUTES--USE H FOR ONE-HALF MINUTE) DEFAULT IENDTM=100000 (SEE DOCUMENTATION).
17-24	NT	0	LE 0 -DO NOT COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE GT 0 -COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE
25-32	NVFCAL	0	GT 0 -VF CALCULATED USING FINITE DIFFERENCE METHOD LT 0 -VF CALCULATED USING CONTOUR INTEGRATION METHOD EQ 0 -PROGRAM SELECTS METHOD TO BE USED (SEE DOCUMENTATION)
33-40	NFE	0	LE 0 -MESH SIZE FOR EACH ELEMENT SPECIFIED ON INPUT DATA GT 0 -MESH SIZE SET TO 1 BY 1 FOR EACH ELEMENT. (OVERRIDES INPUT DATA)
41-48	NFS	0	GE 0 -SHADING CONSIDERED USING DATA FROM INPUT LT 0 -NEGLECT ALL SHADING
49-56	RMAX	0.0	MAXIMUM AREA/DISTANCE RATIO. DEFAULT RMAX = 0.1 (SEE DOCUMENTATION).

A-5

NASTRAN FORMATTED INPUT
SEMI-SORTED BULK DATA ECHO

1	2	3	4	5	6	7	8	9	10
*****	*****	*****	*****	*****	*****	*****	*****	*****	*****
CHBDY	10		AREA4	10	11	12	13	50	
CHBDY	20		REV	20	30			55	
CHBDY	30		AREA4	100	103	102	101	60	
CHBDY	40	10	POINT	50				65	+C1
+C1					0.0	0.0	1.		
CHBDY	50	10	POINT	40				65	+C2
+C2					0.0	0.0	-1.0		
GRID	10		2.5	-5.0	0.0				
GRID	11		2.5	5.0	0.0				
GRID	12		-2.5	5.0	0.0				
GRID	13		-2.5	-5.0	0.0				
GRID	100		2.5	-2.5	10.0				
GRID	101		2.5	2.5	10.0				
GRID	102		-2.5	2.5	10.0				
GRID	103		-2.5	-2.5	10.0				
GRID	20		.5						
GRID	30		.5		10.0				
GRID	40								
GRID	50				10.0				
PHBDY	10		.786						
\$VIEW	50	0	1	25	15				
\$VIEW	55	1	0	10	20				
\$VIEW	60	0	1	15	15				
\$VIEW	65	1	1	5	5				

CONTINUED

NASTRAN FORMATTED INPUT
SEMI-SORTED BULK DATA ECHO

INPUT SUMMARY

NUMBERS OF COMPLETE LOGICAL CARD TYPES DETECTED

CHBDY = 5 CORD1 = 0 CORD2 = 0 GRDSET = 0 GRID = 12 PHBDY = 1 VIEW = 4

VIEW FACTORS

ELEMENT	AREA	ELEM TO ELEM = VIEW FACTOR	ELEM TO ELEM = VIEW FACTOR	VF SUM FROM ELEMENT
10	0.50000E 02	*****		*****
		10 - 20 =0.673263E-01	20 - 10 =0.107153E 00	
		10 - 30 =0.503031E-01	30 - 10 =0.100606E 00	
				0.11763E 00
20	0.31416E 02	*****		*****
		20 - 30 =0.728946E-01	30 - 20 =0.916021E-01	
				0.18005E 00
30	0.25000E 02	*****		*****
				0.19221E 00
40	0.78600E 00	*****		*****
				0.0
50	0.78600E 00	*****		*****
				0.0

PROGRAM SUMMARY FOR CASE 1

PROGRAM TERMINATED NORMALLY

CPU TIME (SECONDS)

PROCESSING OF INPUT DATA= 0
VIEW FACTOR COMPUTATIONS= 19

NUMBER OF VIEW FACTOR COMPUTATIONS

FINITE DIFFERENCE= 115221
CONTOUR INTEGRAL= 540

REQUESTED CORE SPACE ALLOWS

300 ELEMENTS
660 SUB-ELEMENTS
1360 SUB-ELEMENTS
AFTER COMPRESSION
900 GRID POINTS

THE INPUT DATA GENERATED

5 ELEMENTS
850 SUB-ELEMENTS
12 GRID POINTS

21K BYTES OF CORE WERE NOT USED IN THIS CASE.

(62K ON ELEMENTS, 21K ON SUB-ELEMENTS AND 62K ON GRID POINTS.)

* END CASE 1 *

APPENDIX B

SAMPLE PROBLEM USING NASTRAN AND RAVFAC-TYPE DATA

The physical structure to be modeled in this problem is shown in Figure B-1.

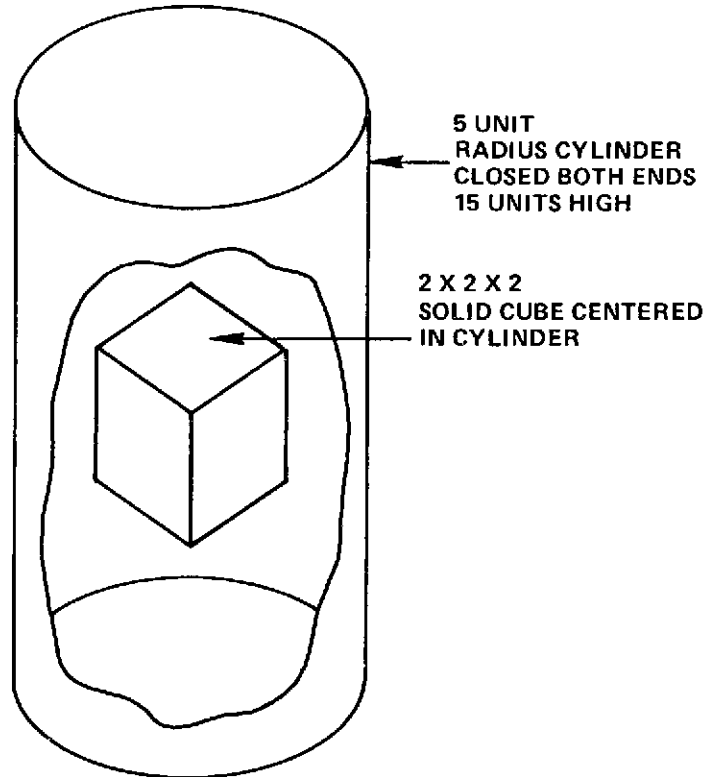


Figure B-1. Structure To Be Modeled

Figure B-1 might be an approximation of a spacecraft in a thermal vacuum chamber. Suppose it is desired to determine the view factors between the exterior of the spacecraft and the interior of the chamber.

The model should be made using NASTRAN-type data for the spacecraft and RAVFAC-type data for the chamber. Locating the origin of the basic coordinate system at the center of the spacecraft, the element model in the basic coordinate system can be drawn as seen in Figure B-2.

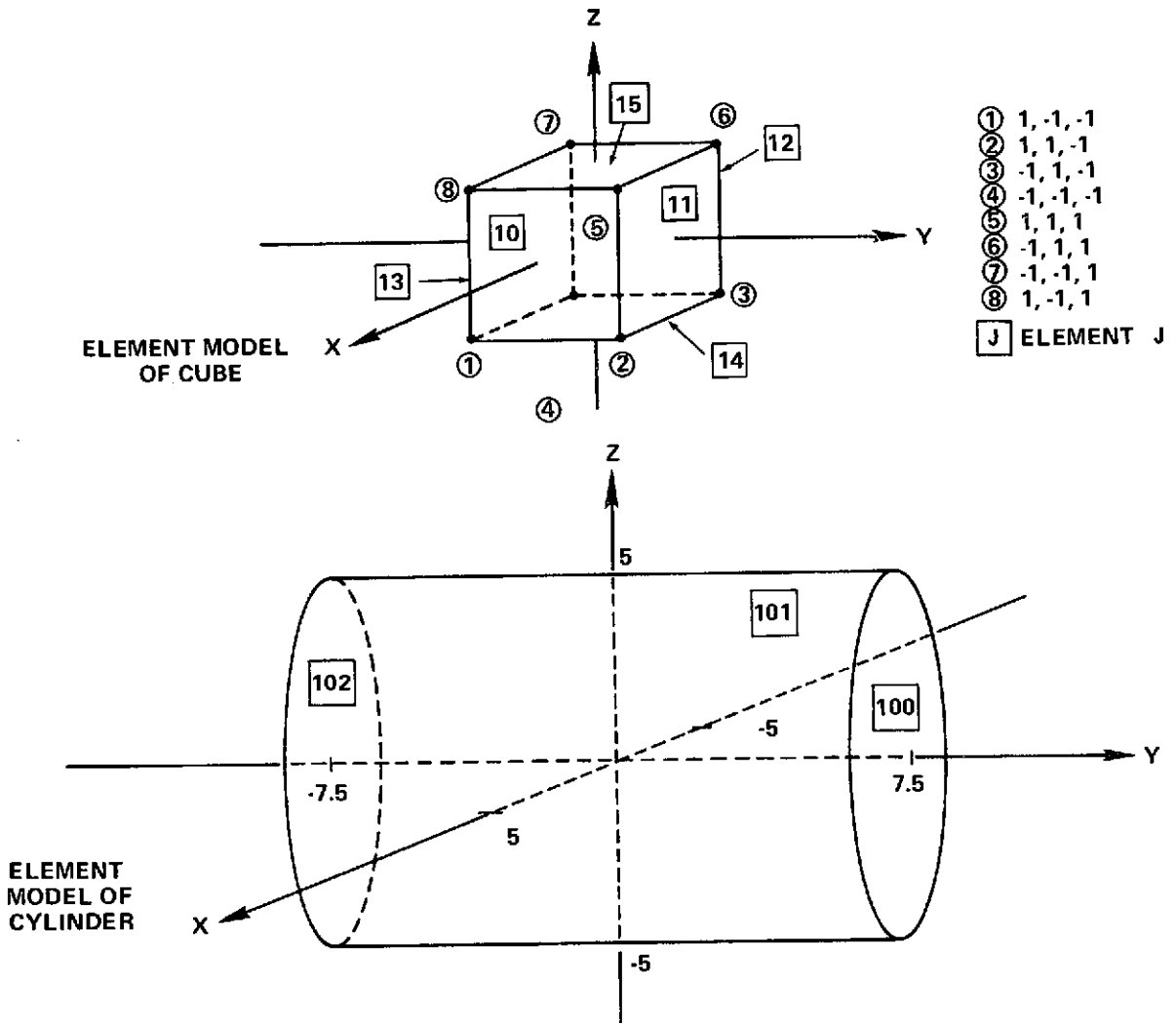


Figure B-2. Element Model in the Basic Coordinate System

Table B-1 lists the optional shading flags.

Table B-1
Shading Flags

Element	Can Shade	Can Be Shaded
10	YES	NO
11	YES	NO
12	YES	NO
13	YES	NO
14	YES	NO
15	YES	NO
100	NO	YES
101	NO	YES
102	NO	YES

Figure B-3 is a listing of the input used to run this problem. It is followed by a copy of the program output.

```
//B9EFPT02 JOB (■■■■■1831C,T,B00081,009002),YYY,MSGLEVEL=(2,0)
// EXEC VIEW,REGION=200K
TITLE    /// SAMPLE PROBLEM NO. 2 ///
CASE=    9
BEGIN BULK
GRID      1          1.0  -1.0  -1.0
GRID      2          1.0   1.0  -1.0
GRID      3         -1.0   1.0  -1.0
GRID      4         -1.0  -1.0  -1.0
GRID      5          1.0   1.0   1.0
GRID      6         -1.0   1.0   1.0
GRID      7         -1.0  -1.0   1.0
GRID      8          1.0  -1.0   1.0
CHBDY    10        AREA4      1      2      5      8      25
CHBDY    11        AREA4      2      3      6      5      25
CHBDY    12        AREA4      3      4      7      6      25
CHBDY    13        AREA4      8      7      4      1      25
CHBDY    14        AREA4      4      3      2      1      25
CHBDY    15        AREA4      8      5      6      7      25
$VIEW    25          1      0      5      5
ENDDATA
RAVFAC DATA
100 1          END OF CHAMBER
-2  1  1  5  5  7.5  0.0  5.0  0.0  360.
                                -90.
102 1          OTHER END OF CHAMBER
-2  1  1  5  5  7.5  0.0  5.0  0.0  360.
                                90.
101 1          CHAMBER SIDES
-4  1  1  40  20  5.0  -7.5  7.5  0.0  360.
                                90.
-1
ENDCASE
```

Figure B-3. Program Input for Sample Problem 2

G O D D A R D S P A C E F L I G H T C E N T E R N A S A
* * * * * * * * * * * * * * * * * * * * * * * * *

THE

V V
V V
V V
V V
V V
V V
V V
V V
V V
V V

I
I
I
I
I
I
I
I
I
I

EEEEEEE
E
E
E
EEEE
EEEE
E
E
E
EEEEEEE

W W
W W
W W
W W
W W
W W
W W
W W
W W
W W

PROGRAM

ED PUCCINELLI
REG MITCHELL
CLIFF JACKSON

VERSION ONE----OCTOBER 31, 1972

DATE 7/27/73
TIME 9.27.29

* CASE 1 *

PROBLEM TITLE /// SAMPLE PROBLEM NO. 2 ///

CASE CONTROL CARD

COLUMNS	VARIABLE	VALUE	DESCRIPTION
9-16	IENDTM	6	ENTER CPU TIME USED ON JOB CARD (IN MINUTES--USE H FOR ONE-HALF MINUTE) DEFAULT IENDTM=100000 (SEE DOCUMENTATION).
17-24	NT	0	LE 0 -DO NOT COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE GT 0 -COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE
25-32	NVFCAL	0	GT 0 -VF CALCULATED USING FINITE DIFFERENCE METHOD LT 0 -VF CALCULATED USING CONTOUR INTEGRATION METHOD EQ 0 -PROGRAM SELECTS METHOD TO BE USED (SEE DOCUMENTATION)
33-40	NFE	0	LE 0 -MESH SIZE FOR EACH ELEMENT SPECIFIED ON INPUT DATA GT 0 -MESH SIZE SET TO 1 BY 1 FOR EACH ELEMENT. (OVERRIDES INPUT DATA).
41-48	NFS	0	GE 0 -SHADING CONSIDERED USING DATA FROM INPUT LT 0 -NEGLECT ALL SHADING
49-56	RMAX	0.0	MAXIMUM AREA/DISTANCE RATIO. DEFAULT RMAX = 0.1 (SEE DOCUMENTATION).

B-5

NASIRAN FORMATTED INPUT
SEMI-SORTED BULK DATA ECHO

	1	2	3	4	5	6	7	8	9	10
CHBDY	10		AREA4	1	2	5	8	25		
CHBDY	11		AREA4	2	3	6	5	25		
CHBDY	12		AREA4	3	4	7	6	25		
CHBDY	13		AREA4	8	7	4	1	25		
CHBDY	14		AREA4	4	3	2	1	25		
CHBDY	15		AREA4	8	5	6	7	25		
GRID	1		1.0	-1.0	-1.0					
GRID	2		1.0	1.0	-1.0					
GRID	3		-1.0	1.0	-1.0					
GRID	4		-1.0	-1.0	-1.0					
GRID	5		1.0	1.0	1.0					
GRID	6		-1.0	1.0	1.0					
GRID	7		-1.0	-1.0	1.0					
GRID	8		1.0	-1.0	1.0					
\$VIEW	25	1	0	5	5					

INPUT SUMMARY

NUMBERS OF COMPLETE LOGICAL CARD TYPES DETECTED

CHBDY = 6 CORD1 = 0 CORD2 = 0 GRDSET = 0 GRID = 8 PHBDY = 0 VIEW = 1

RAVFAC DATA

SURFACE INPUT DATA

SURFACE NUMBER 100 CAN SHADE - NO END OF CHAMBER
 CAN BE SHADED- YES

SURFACE TYPE = -2 ALPHA = 0.75000E 01
 NODES, B-DIR.= 1 ELEM/NODE, B-DIR.= 5 BETA(MIN) = 0.0 BETA(MAX) = 0.50000E 01
 NODES, G-DIR.=, 1 ELEM/NODE, G-DIR.= 5 GAMMA(MIN)= 0.0 GAMMA(MAX)= 0.36000E 03

R(X)= 0.0 R(Y)= 0.0 R(Z)= 0.0
 PHI = 0.0 PSI = 0.0 OMEGA=-0.90000E 02 REF. TO INTERMEDIATE COORDINATE SYSTEM 0

SURFACE NUMBER 200 CAN SHADE - NO OTHER END OF CHAMBER
 CAN BE SHADED- YES

SURFACE TYPE = -2 ALPHA = 0.75000E 01
 NODES, B-DIR.= 1 ELEM/NODE, B-DIR.= 5 BETA(MIN) = 0.0 BETA(MAX) = 0.50000E 01
 NODES, G-DIR.=, 1 ELEM/NODE, G-DIR.= 5 GAMMA(MIN)= 0.0 GAMMA(MAX)= 0.36000E 03

R(X)= 0.0 R(Y)= 0.0 R(Z)= 0.0
 PHI = 0.0 PSI = 0.0 OMEGA= 0.90000E 02 REF. TO INTERMEDIATE COORDINATE SYSTEM 0

SURFACE NUMBER 300 CAN SHADE - NO CHAMBER SIDES
 CAN BE SHADED- YES

SURFACE TYPE = -4 ALPHA = 0.50000E 01
 NODES, B-DIR.= 1 ELEM/NODE, B-DIR.= 40 BETA(MIN) = -0.75000E 01 BETA(MAX) = 0.75000E 01
 NODES, G-DIR.=, 1 ELEM/NODE, G-DIR.= 40 GAMMA(MIN)= 0.0 GAMMA(MAX)= 0.36000E 03

R(X)= 0.0 R(Y)= 0.0 R(Z)= 0.0
 PHI = 0.0 PSI = 0.0 OMEGA= 0.90000E 02 REF. TO INTERMEDIATE COORDINATE SYSTEM 0

B-7

VIEW FACTORS

ELEMENT	AREA	ELEM TO ELEM = VIEW FACTOR	ELEM TO ELEM = VIEW FACTOR	VF SUM FROM ELEMENT
10	0.40000E 01	10 - 100 =0.272450E-01 10 - 200 =0.272449E-01 10 - 300 =0.944122E 00	100 - 10 =0.138757E-02 200 - 10 =0.138757E-02 300 - 10 =0.801395E-02	0.99861E 00
11	0.40000E 01	11 - 100 =0.368380E 00 11 - 300 =0.632604E 00	100 - 11 =0.187614E-01 300 - 11 =0.536971E-02	0.10010E 01
12	0.40000E 01	12 - 100 =0.272449E-01 12 - 200 =0.272449E-01 12 - 300 =0.944122E 00	100 - 12 =0.138757E-02 200 - 12 =0.138757E-02 300 - 12 =0.801395E-02	0.99861E 00
13	0.40000E 01	13 - 200 =0.368380E 00 13 - 300 =0.632604E 00	200 - 13 =0.187614E-01 300 - 13 =0.536971E-02	0.10010E 01
14	0.40000E 01	14 - 100 =0.291990E-01 14 - 200 =0.233474E-01 14 - 300 =0.944122E 00	100 - 14 =0.148709E-02 200 - 14 =0.118907E-02 300 - 14 =0.801395E-02	0.99667E 00
15	0.40000E 01	15 - 100 =0.233475E-01 15 - 200 =0.291991E-01 15 - 300 =0.944114E 00	100 - 15 =0.118908E-02 200 - 15 =0.148710E-02 300 - 15 =0.801388E-02	0.99666E 00
100	0.78540E 02	100 - 200 =0.750117E-01 100 - 300 =0.892820E 00	200 - 100 =0.750116E-01 300 - 100 =0.148803E 00	0.99204E 00
200	0.78540E 02	200 - 300 =0.892821E 00	300 - 200 =0.148804E 00	0.99205E 00
300	0.47124E 03	300 - 300 =0.655388E 00	300 - 300 =0.655387E 00	0.99579E 00

PROGRAM SUMMARY FOR CASE 1

PROGRAM TERMINATED NORMALLY

CPU TIME(SECONDS)

PROCESSING OF INPUT DATA= 0
VIEW FACTOR COMPUTATIONS= 166

NUMBER OF VIEW FACTOR COMPUTATIONS

FINITE DIFFERENCE= 651256
CONTOUR INTEGRAL= 13902

REQUESTED CORE SPACE ALLOWS

416 ELEMENTS
894 SUB-ELEMENTS
1862 SUB-ELEMENTS
AFTER COMPRESSION
1248 GRID POINTS

THE INPUT DATA GENERATED

9 ELEMENTS
1000 SUB-ELEMENTS

8 GRID POINTS

39K BYTES OF CORE WERE NOT USED IN THIS CASE.

(86K ON ELEMENTS, 39K ON SUB-ELEMENTS AND 8/K ON GRID POINTS.)

* END CASE 1 *

APPENDIX C

SAMPLE PROBLEM DEMONSTRATING THE USE OF LOCAL COORDINATE SYSTEMS

The physical structure to be modeled in this problem is shown in Figure C-1.

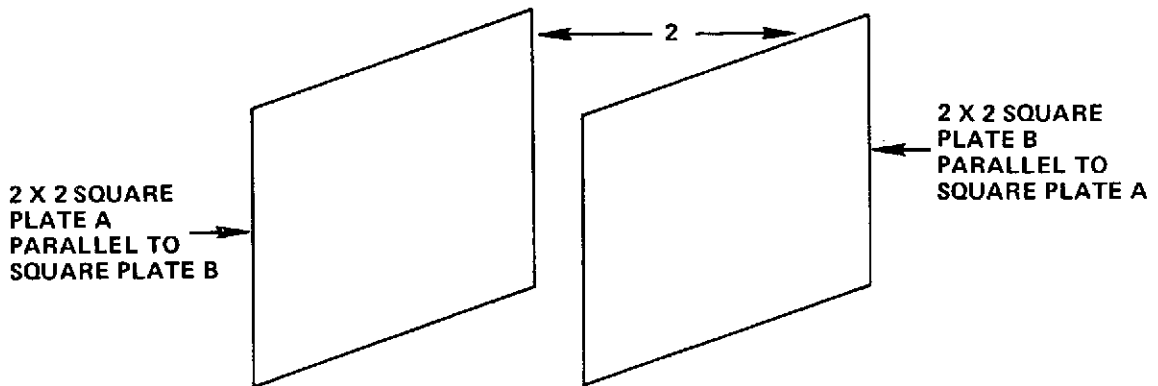


Figure C-1. Structure to Be Modeled

The structure to be modeled consists of the two opposite sides of a cube "looking" at each other. Plate A will be modeled using NASTRAN-type data and using a local coordinate system while plate B will be modeled using RAVFAC-type data and another local coordinate system.

The basic coordinate system will be chosen to lie halfway between the two plates with the X-Z plane parallel to the plates. The local coordinate systems will be simple translations and rotations so that the X-Y planes contain their respective plates. This is shown in Figure C-2.

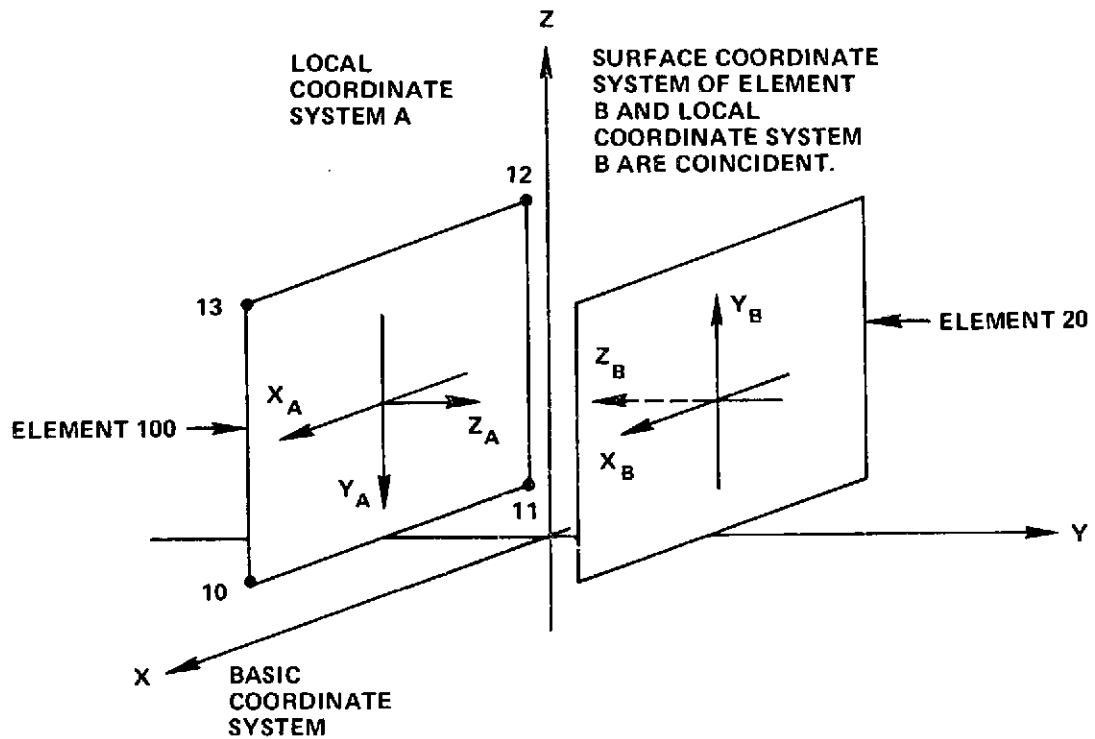


Figure C-2. Element Model and Its Coordinate Systems

The coordinates of grid points 10 through 13 with respect to local coordinate system A are respectively $(1, 1, 0)$, $(-1, 1, 0)$, $(-1, -1, 0)$, $(1, -1, 0)$.

Figure C-3 is a listing of the input used to run this problem. It is followed by a copy of the program output.

DATE 9/21/73
TIME 11.23.23

* CASE 1 *

PROBLEM TITLE *** SAMPLE PROBLEM NO. 3 ***

CASE CONTROL CARD

COLUMNS	VARIABLE	VALUE	DESCRIPTION
9-16	IENDTM	H	ENTER CPU TIME USED ON JOB CARD (IN MINUTES--USE H FOR ONE-HALF MINUTE) DEFAULT IENDTM=100000 (SEE DOCUMENTATION).
17-24	NT	0	LE 0 -DO NOT COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE GT 0 -COPY INPUT DATA ONTO TAPE (UNIT 2) FOR RESTART USE
25-32	NVFCAL	0	GT 0 -VF CALCULATED USING FINITE DIFFERENCE METHOD LT 0 -VF CALCULATED USING CONTOUR INTEGRATION METHOD EQ 0 -PROGRAM SELECTS METHOD TO BE USED (SEE DOCUMENTATION)
33-40	NFE	0	LE 0 -MESH SIZE FOR EACH ELEMENT SPECIFIED ON INPUT DATA GT 0 -MESH SIZE SET TO 1 BY 1 FOR EACH ELEMENT. (OVERRIDES INPUT DATA)
41-48	NFS	0	GE 0 -SHADING CONSIDERED USING DATA FROM INPUT LT 0 -NEGLECT ALL SHADING
49-56	RMAX	0.0	MAXIMUM AREA/DISTANCE RATIO. DEFAULT RMAX = 0.1 (SEE DOCUMENTATION).

NASIRAN FORMATTED INPUT
SEMI-SORTED BULK DATA ECHO

	1	2	3	4	5	6	7	8	9	10
CHBDY	100			AREA4	10	11	12	13	50	
CORD2R	5			0.0	-1.0	1.0	0.0	-0.5	1.0	+C1
+C1	.5		-0.5	1.0						
GRID	10	5		1.0	1.0	0.0				
GRID	11	5		-1.0	1.0	0.0				
GRID	12	5		-1.0	-1.0	0.0				
GRID	13	5		1.0	-1.0	0.0				
\$VIEW	50		0	0	2	2				

INPUT SUMMARY

NUMBERS OF COMPLETE LOGICAL CARD TYPES DETECTED

CHBDY = 1 CORD1 = 0 CORD2 = 1 GRDSET = 0 GRID = 4 PHBDY = 0 VIEW = 1

RAVFAC DATA

SURFACE INPUT DATA

SURFACE NUMBER 20 CAN SHADE - NO PLATE B
CAN BE SHADED- NO

SURFACE TYPE = 1 ALPHA = 0.0
NODES, B-DIR.= 1 ELEM/NODE, B-DIR.= 2 BETA(MIN) = -0.10000E 01 BETA(MAX) = 0.10000E 01
NODES G-DIR. =, 1 ELEM/NODE, G-DIR.= 2 GAMMA(MIN) = -0.10000E 01 GAMMA(MAX) = 0.10000E 01

R(X) = 0.0 R(Y) = 0.0 R(Z) = 0.0
PHI = 0.0 PSI = 0.0 OMEGA = 0.0 REF. TO INTERMEDIATE COORDINATE SYSTEM 15

INTERMEDIATE COORDINATE SYSTEMS

ICS NUMBER 15 RX=0.0 RY=0.10000E 01 RZ=0.10000E 01 PHI=0.0 PSI=0.0 OMEGA=0.90000E 02

VIEW FACTORS

ELEMENT	AREA	ELEM TO ELEM = VIEW FACTOR	ELEM TO ELEM = VIEW FACTOR	VF SUM FROM ELEMENT
100	0.40000E 01	***** 100 - 20 = 0.207850E 00	***** 20 - 100 = 0.207849E 00	***** 0.20785E 00
20	0.40000E 01	*****	*****	***** 0.20785E 00

PROGRAM SUMMARY FOR CASE 1

PROGRAM TERMINATED NORMALLY

CPU TIME(SECONDS)

PROCESSING OF INPUT DATA= 0
VIEW FACTOR COMPUTATIONS= 0

NUMBER OF VIEW FACTOR COMPUTATIONS

FINITE DIFFERENCE= 0
CONTOUR INTEGRAL= 16

REQUESTED CORE SPACE ALLOWS

180 ELEMENTS
393 SUB-ELEMENTS
815 SUB-ELEMENTS
AFTER COMPRESSION
540 GRID POINTS

THE INPUT DATA GENERATED

2 ELEMENTS
8 SUB-ELEMENTS

4 GRID POINTS

36K BYTES OF CORE WERE NOT USED IN THIS CASE.

(36K ON ELEMENTS, 36K ON SUB-ELEMENTS AND 36K ON GRID POINTS.)

* END CASE 1 *
