1987

Display of molecular models with interactive computer graphics

Steve Wuter Yang

Rochester Institute of Technology

School of Computer Science and Technology

Display of Molecular Models

with

Interactive Computer Graphics

by

Steve Wuter Yang

A thesis, submitted to
The Faculty of the School of Computer Science and Technology,
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

Approved by:

Guy Johnson
Chairman, Professor Guy Johnson

Donald Kreher June 15, 1987
Professor Donald Kreher

Andrew Kitchen
Professor Andrew Kitchen

June 8, 1987

I prefer to be contacted whenever reproductions of this document are requested. I can be contacted at the following address.

Steve Wuter Yang

Steve Wuter Yang, June 8, 1987.

3-3 Fu-Yuan Street, Lane 211

Taipei, Taiwan

Republic of China

Dedication

To my wife, Li-Yun,

for her encouragement, and love

to sustain me.

Acknowledgement

I would like to thank my adviser, professor Guy Johnson, for his guidance and encouragement during my thesis work. I also thank professor Donald Kreher and professor Andrew Kitchen for their suggestions and criticisms.

Also, I am very grateful to my parents, brothers, sisters, relatives, and friends for their moral support and financial support.

Lastly, I thank my lovely daughter, Kailing, for her joy to gear me up.

Table of Contents

Abstract

Computer graphics devices offer a way to present images of three-dimensional objects with enough detail to aid human understanding of the structure of molecules. In this thesis, the Computer Aided Molecular Design system (CAMD system) will illustrate this idea. The CAMD system is an educational package designed for students in high school. The user of the CAMD system can create different molecules and store these molecules into a relational data base for future display. A Relational Data Base Management system was implicitly implemented for the CAMD system.

1. Introduction.

1.1. Display of Molecular Model.

Chemists have used mechanical models to study and understand the structure of molecules, particularly large, organic ones. These models represent the locations of atoms in the molecule relative to one another. In this project, users , especially the student in high school. can create and retrieve different molecules to display on the screen.

The CAMD system produces colored, and ball-and-stick renditions of molecular models with highlighting (illumination, shining, or shading). The purpose is to allow humans to visualize molecules interactively in a manner similar to that offered by plastic ball-and-stick models.

Some basic geometric structures are supplied as tools for users to create new molecular models. These geometric structures are line, triangle, trigonal plane, square plane, tetrahedron, trigonal pyramid, trigonal bi-pyramid, octahedron, and square pyramid. For example, both methane (CH4) and carbon tetrachloride (CCl4) are tetrahedral models. If these two molecular models have to be created, then these two tetrahedral models will be different in the radius and color of some atoms. The atom

of carbon will be the same of color and size at any time whenever it appears in any geometric model in the CAMD system.

During the last fifteen years a number of graphic systems have been developed for researchers to model and study molecular structures. The ATOMS system has been developed at Bell Laboratories. ATOMS is a set of FORTRAN subroutines for depicting, in color, configurations of space-filling atoms and/or ball-and-stick models. Pictures are rendered as in traditional cartoon animation: solid monochrome areas separated by thin black lines [25]. The user supplies x, y, z, radius, and color information for atoms; atom pairs, thickness and color for bonds, also parameters defining rules for viewing and picture drawing-including position and orientation of a hypothetical camera in use-defined space. The system is ideally suited for use with a filming unit that contains or is driven by a minicomputer. The host computer can then perform visibility calculations, yielding parameterized descriptions of visible patches of various colors; a routine resident in the mini generates the appropriate set of parallel vectors to fill solidly each patch area. The system is intended for both educational stills and movies, and for research purposes that benefit from good visual display of hypothetical morphologies and kinetics. The goal of the system is efficient-and thus

affordable-computation yielding approximate perspective views of user-defined situations. This is achieved by using simplified picture elements and simple, approximate calculations while maintaining visual validity for the configurations that make sense for chemistry. Picture elements treated are in fact bounded only by circular arcs and straight lines; efficiency is further enhanced by doing most of the computation on the picture plane-a circumstance commonly called "2 1/2 D" which deals with flat objects that hide pieces of one another according to an ordering in depth. The intended pictorial output consists of filled, solid-color areas separated from one another by thin black lines as in standard cartoon cell animation. This is done normally either by exposing color film directly in a filming device containing a color wheel, or by making the appropriate sequence of color separation negatives, to be combined and colored later by optical printing. This system is particularly well suited for use with a film exposing device containing or driven by a minicomputer: the main (host) computer's job is then to perform visibility calculations, delivering to the actual picture-drawer a list of parameterized descriptions of patches of various colors; the mini contains a program for calculating the many parallel vectors that effectively fill the patch. This is the system with the following properties: it treats ball-and-stick models, interpenetrating atoms, and intermediate stages; it yields

perspective pictures in color with hidden surfaces removed; it is computationally fast.

Chemically Oriented Graphics System (COGS) is a molecular modelling system, for small and macro-molecules, which incorporate a wide range of functionality has been developed [26]. The system is 'user friendly' and is controlled almost exclusively by a puck (mouse), in a manner akin to the Apple Macintosh. The system is written in FORTRAN 77 and the graphics adheres to the CORE standard, so that a reasonable degree of portability is assured. COGS is a molecular modelling program designed to be useful in four major areas: small molecule stereochemical design, model building, and comparison; investigation of the interaction between organic or inorganic crystal lattices with small and medium sized organic molecules; enzyme-substrate or enzyme-inhibitor docking studies; and protein engineering. The system is designed to be as comprehensive as possible, while remaining easy to use. COGS was developed by White and Pearson in 1985, UK. Most of the facilities in COGS perform their operations immediately, with the obvious exception of molecular mechanics and molecular dynamics calculations, and conformational search procedures. COGS incorporates extensive error checking so that it is very difficult, but not yet impossible, to crash the program by doing something nonsensical. When an error is trapped the

user is always given an opportunity to correct it (interactively) without losing place in a sequence of commands. There are a number of main menu entries in COGS, each of which is the head of a tree of a number of submenus. The "draw molecules" option allows models to be constructed from a 2D structural chemical formula of the desired molecule drawn with the puck. This occurs in exactly the same way as it would be drawn by a chemist using pencil and paper. This 2D diagram, of connected atoms of different types, is converted automatically by COGS into a full 3D molecular representation which is then displayed on the screen. Models constructed with draw molecules option will overwrite any other molecule(s) currently in the workspace. COGS consists of a family of subroutines (around 150), each with a single well defined and logically distinct function, which communicate with each other via FORTRAN common blocks (arguments to subroutines are avoided where sensibly possible). Extensive use is made of structured programming techniques and long variable names (with words separated by underscores) so that the intent of each subroutine is clear and may almost be read as English text. In addition each subroutine has a multiline text header describing its function, entry points, references to the algorithm used, etc.

The MAGIC system (Modeling And Graphics In Chemistry) has been developed for interactively handling and manipulating in the computer molecules usually dealt with in organic or inorganic chemistry [27]. The emphasis has been put on fast input, display, and editing of structures. MAGIC overcomes disturbing tracking subpictures for the light pen and the clumsy operation by simultaneously manipulating light pen and keyboard. The system is designed for easy use, is self explanatory, and does not require any complicated command language. All functions of the system are invoked via menus using a light pen. The system has powerful modeling and display functions allowing for directly modeling molecular structures with a light pen. The MAGIC system is designed by Bauer and Schubert at Organisch-Chemisches Institute, West Germany, in 1982. In order facilitate direct editing of structures stick diagrams are used to display the structures on the screen. More elaborate pictures with atoms represented as spheres and hidden lines eliminated can be generated on a plotter. The functions of MAGIC may be conceptually divided into modeling functions and those for computing structural parameters and saving or recalling models. Modeling a structure comprises entering, editing, and display the structure. Stick diagrams appear on the display in order to keep the number of subpictures small and to avoid flickering of the screen. Atoms and bonds in front of the screen can be

highlighted to give a better impression of the spatial arrangement of the atoms. More elaborated representations like ball and stick diagrams or diagrams of space filling models are not chosen to appear on the screen. These representations usually contain too many lines to allow specific light pen hits for editing of the picture. MAGIC fully utilizes the storage capabilities of a computer. Complete models or those being developed, substructures or building blocks may be stored and retrieved by MAGIC. This way the user is able to add, modify or delete items in the collection of building blocks maintained by the system. Therefore, MAGIC may be used to build a data bank of structures and their models. Instead of storing generated structures, they can either be printed or punched in interpreted form and then be used for other purposes. The interpreted form contains all pertinent data for reconstructing the three dimensional model so the data may be transferred to other programs or installations. Many structural properties of molecules are readily described in terms of distances and angles between atoms, bonds, lines, and planes within the molecule. MAGIC therefore allows for calculation of all these parameters. The operations for calculating geometric parameters are loaded from a menu and the atoms, bonds, lines, or planes are specified as attributes through light pen hits. The value of the geometric parameter calculated is immediately displayed.

1.2. Relational Database Model.

A "database" is a computerized storehouse of data. The data is organized so that it can be retrieved and processed, for one and only one purpose : to answer questions. A "database management system" is a computer program that works as a data librarian. The database management system (DBMS) organizes the electronic equivalent of bookshelves. It places data on shelves, retrieves data from the shelves, and combines and manipulates data according to instructions provided by the user. It's like having your own personal assistant who remembers data you provide. keeps track of where it is and assembles the data in various ways to provide answers to questions you ask.

A data base can be analyzed from two viewpoints -- the physical storage of the data and the logical, or conceptual, view of data. Files are used to physically store data in a data base. Most data bases use either direct files or indexed files, or a combination of the two, to physically store data on disk.

The logical, or conceptual, view of a data base is concerned with how data is logically organized and how the data can be retrieved for information purposes. The three conceptual views of a data base are a hierarchical data

base, a network data base, and a relational data base. A hierarchical data base consists of elements which act in a parent-child relationship. This is the oldest of conceptual data base, such as IMS from IBM. A network data base acts in much the same manner as a hierachical data base except that an element can have more than one parent. In network database nomenclature, a parent is called an owner and the child is called a member.

A relational data base is the newest conceptual view of data and offers many advantages. The most important advantage is that the relationships between data can be determined dynamically at the time the user requests information from the data base. To begin, a relation is defined as a conceptual file where each conceptual record is unique and each conceptual record has the same number and type of fields. A conceptual file containing records is also called a table and each record within the file is sometimes called a tuple. The relational data base model was selected for the CAMD system.

Binary tree algorithms and index sequential algorithms can be used to create and search molecular models in the relational data base. For example, a user may wish to store or display a molecule of water (H2O). A user will go to a relational table to check whether this molecule exists or not. If it is not in the CAMD system,

then the user stores some relevant data for this molecule. The name of the molecule should be a primary key in the relational table to avoid duplicate molecules. If user wants to display this molecule, the CAMD system will go to one relational table called GEOMETRY to get the central value of each atom, and go to another relational table called ATOM TABLE to get radius and color for each atom. The name of the basic geometric structure also will be a key in a table.

The following relations exist in the CAMD system.

relation 1:

MOLECULE(MOLE.NAME, MODEL.TYPE).

relation 2:

ATOM TABLE(ATOM.NAME, RADIUS, COLOR).

relation 3:

LINEAR(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3).

relation 4:

TRIANGLE(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3).

relation 5:

TRIGONAL PLANE(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3, ATOM.4).

relation 6:

SQUARE PLANE(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3, ATOM.4, ATOM.5).

relation 7:

TRIGONAL PYRAMID(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3, ATOM.4).

relation 8:

TETRAHEDRAL(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3, ATOM.4, ATOM.5).

relation 9:

TRIGONAL BI-PYRAMID(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3,
ATOM.4, ATOM.5, ATOM.6).

relation 10:

SQUARE PYRAMID(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3,
ATOM.4, ATOM.5, ATOM.6).

relation 11:

OCTAHEDRAL(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3, ATOM.4,
ATOM.5, ATOM.6, ATOM.7).

relation 12:

GEOMETRY(GEOM.NAME, X.1, Y.1, Z.1, RADIUS.1, COLOR.1,
X.2, Y.2, Z.2, RADIUS.2, COLOR.2,
X.3, Y.3, Z.3, RADIUS.3, COLOR.3,
X.4, Y.4, Z.4, RADIUS.4, COLOR.4,
X.5, Y.5, Z.5, RADIUS.5, COLOR.5,
X.6, Y.6, Z.6, RADIUS.6, COLOR.6,
X.7, Y.7, Z.7, RADIUS.7, COLOR.7).

1.3. Computer Graphics Overview.

Computer graphics is often used to assist business executives in analyzing data and to assist individuals in preparing data for analysis and presentation to others [7]. Computer graphics allows information to be displayed in the form of charts, graphs, or pictures so that the information can easily and quickly be understood. A picture is the fundamental cohesive concept in computer graphics. This thesis considers how molecules are represented in computer graphics, how molecules are prepared for presentation, and how interaction with the molecule is accomplished. Many computer graphics applications involve the display of three-dimensional objects and scenes [9]. For example, computer-aided design systems allow their users to manipulate models of machined components, automobile bodies and aircraft parts. These applications differ from two-dimensional applications not only in the added dimension, but they also require concern for realism in the display of objects. In this thesis, the Computer Aided Molecular Design system (the CAMD system) is a 3-D computer graphics application.

1.4. Vectrix Graphic Processor.

In the CAMD system, the Vectrix VX384, a graphic processor, and Vectrix VXM monitor is used to display molecules. The VX384 can be used with any computer having a serial or parallel interface. This processor can be operated with any operating system or language. The CAMD system was implemented in the C programming language running under the Unix operating system.

The VX384 can display 512 colors simultaneously from a palette of 16 million colors and has a resolution of 672 x 480 pixels. The screen is laid out as a rectangle in 2-D mode. The pixels are numbered 0 - 671 along the horizontal axis, and 0 - 479 on the vertical axis. The coordinate (0,0) is in the lower left hand corner of the screen; The coordinate (671, 479) is in the upper right hand corner. The Vectrix graphic processor is designed to relieve the host computer of graphic calculations and to relieve the programmer of subroutine and procedure calls.

The Vectrix VXM monitor is a high quality, high resolution monitor well matched to the capabilities of the frame buffer. There are almost 76 available commands that can be used in interactive mode to demonstrate use of the Vectrix. The Vectrix graphics processor offers a wide variety of geometric forms that can be combined on the display to create a virtually infinite number of images.

These basic forms are called graphics primitives. The Vectrix command set includes primitives to draw dots, lines, polygons, arcs, patterns, and several types of color fills and floods. Most of these commands work in both two dimensional and three dimensional modes: D (Dot), M (Move), L (Line), and P (Polygon), and F (Filled Polygon) commands work in either 2-D or 3-D space. These commands will be described more detail in chapter 3.5 of this thesis.

The color manipulation commands allow precise color control for graphics and text primitives and the display background. Colors can be mixed in several ways to create very subtle shading effects. Colors can also be used to hide images in the background until a new color definition puts them in the foreground, making possible animation and hidden drawing effects.

The graphics memory is organized as nine bitplanes, and each pixel's color is determined by varying combinations of the nine bits. The VX384 uses the selected color number as a nine bit index to a color lookup table where the color values are stored. This index is grouped into three bits of red, three bits of green, and three bits of blue. Since there are 2^9 combinations possible, 512 colors can be stored in the color table at any given time. when a color table entry

is referenced, three stored eight bit values are output to the RGB monitor. These three values are amounts of red, green, and blue an entry will display. They control the intensities of each of the three color guns in the monitor. Using eight bits for each color allows each color value to be from 0 to 255. Over 16 million (255^3) colors can be created by combining the red, green, and blue components at different levels.

Changing the contents of a color table entry changes the color that will be displayed when a color command references the table. The display is driven by the table contents at time it is scanned. For example, an entry can be made to display a grey by setting each of the three stored values, red, green and blue, to the same value. A 256 entry grey table can be built by giving entries 0 through 255 equal incremental values for red, green, and blue.

The default color table's 512 entries are arranged so that the lower three bits of color number index represent the red components, the middle three bits represent the greens, the high bits represent the blues. As the value of each group of bits increases, the intensity of its respective color increases.

The algorithm used to determine the default color table values is this: The last entry, #511, is set to R=255, G=255, B=255. Red, green and blue values are decreased by 32 in nested loops until all three equal 31. Red is decremented first, then green, then blue. The first entry, #0, is set to R=0, G=0, B=0. Some other Vectrix descriptions can be found in [6].

2. Relational Database Management in the CAMD system.

2.1. Relational Model Approach.

A relational data base is made up of any number relations. Each relation is given a name. For example, "ATOM TABLE" is one of the relation in Computer Aided Molecular Design system (CAMD system). The relation can be viewed as a table that is made up of a number of rows and columns. Each column is called an attribute and given a name such as ATOM.NAME, RADIUS, and COLOR in "ATOM TABLE" relation. The rows of the relation are called tuples or record and contain the data. such as, (H, 0.1250, GREEN) is a tuple of data. The values in each column of each row come from a domain of values. One such domain is associated with each attribute and defines the range of allowed values for that attribute. The instance of a relation is the content of the relation at a particular instant of time.

There is a distinction between domain and attribute. A domain is a set of values, as such it may appear in more than one relation or sometimes more than once in the same relation. It is common to choose domain names to signify value sets; for example, INTEGER, ALPHA, NUMERIC are domain names. Attribute names, on the other hand, are chosen to be meaningful within the context of the

enterprise. COLOR is the third attribute name in "ATOM TABLE" relation. It means that atom H will be drawn in green.

Each relation possess several rules as follows: (1) There is one column in the relation for each attribute of the relation. Each such column is given a name that is unique in the relation. (2) The entries in the column come from the same domain. (3) The order of the columns or attributes in the relation has no significance. (4) The order of the rows is not significant. (5) There are no duplicate rows. The most important term of all is the relation key. This key is the attribute or set of attributes that uniquely identifies tuples in a relation. A relation key is formally defined as a set of one or more relation as attributes concatenated so that the following three properties hold for all time and for any instance of the relation [4]: (1) Uniqueness : The set of attributes takes on a unique value in the relation for each tuple. (2) Nonredundancy : If an attribute is removed from the rest of attributes. the remaining attributes do not possess the uniqueness property. (3) Validity : No attribute value in the key may be null. Null value is a special value that is used to represent "value unknow" or "value inapplicable". It is not the same as blank or zero.

It is possible for relations to have more than one relation key. Each key is made up of a different set of attributes, and is often called the candidate key. If a key is the only key of relation, it is generally referred to as the primary key. There is only one primary key implemented in CAMD system. A primary key also can be a primary key for another relation. such as, MOLE.NAME, which is a attribute name, can be a primary key of both "MOLECULE" table and "TRIANGLE" table.

When an attribute in one relation is a key of another relation, the attribute is called a foreign key. The term means that the attribute is a key, but in a foreign relation. Thus, in the CAMD system, attribute "MODEL.TYPE" of "MOLECULE" relation is a foreign key. To construct the relationship, the system will match "MODEL.TYPE" with values of "GEOM.NAME" in "GEOMETRY" relation. Foreign keys are important when defining constraints across relations. For example, the database designer may want to specify that no "GEOMETRY" tuple can be deleted if its "GEOM.NAME" is a value of foreign key.

The relational model was first proposed by Dr. E. F. Codd in a seminal paper in 1970 [5]. since then, there have been many implementations of the relational approach. System R is a well known relational data base system. It was designed and developed over the period 1974 to 1979 at

the IBM San Jose Research Laboratory. All access to this database is via a data sublanguage called SQL. The original version of SQL was based on an earlier language called SQUARE. The two languages are essentially the same. However, SQUARE uses a rather mathematical syntax whereas SQL is much more English-like. SQL is an acronym for "Structured Query Language". It provides not only retrieval functions but also a full range of update operations. It includes both a data definition language (DDL) and a data manipulation language (DML). DDL provides for the definition or description of database objects, such as DEFINE operation, DESCRIBE operation. DML supports the manipulation or processing of database objects, such as SELECT operation, DELETE operation. The more detail description can be found in [17].

The CAMD system provides a DDL and DML based on some ideas from SQL. SQL is a self-contained command language classified by mode. The CAMD system is a self-contained menu selection language.

In traditional relational data base system, there are several kinds of operations, such as DEFINE, DESCRIBE, INSERT, DISPLAY, SELECT, PROJECT, DELETE, and JOIN. The DEFINE operation is used to create a relation. The DESCRIBE operation is used to describe the properties of attribute characteristics from a relation. The INSERT

operation is used to add one new tuple of data in a relation. The SELECT operation is used to extract certain tuples (rows) from a relation. The PROJECT operation is used to extract attribute(s) (columns) from a relation. The DELETE operation is used to remove tuples from a relation. The DISPLAY operation is used to show the whole relation. The JOIN operation is used to combine relations.

The CAMD system does not provide a SQL-like language to do interactive queries. However, some of these commands are implicitly performed in the CAMD system. It is simple to use the SQL language. SELECT-FROM-WHERE is the fundamental structure of SQL language commands. The SELECT expression specifies the name of all attributes of relation. FROM specifies the relation to be used, and new expression, WHERE, provides the conditions for the selection.

There are several SQL-like examples below. Note that no such interface is currently implemented as part of the CAMD system.

(1). To create a relation using DEFINE operator.

```
DEFINE    MOLECULE
FIELD     MOLE.NAME     TYPE    CHARACTER(10)    PRIMARY KEY
FIELD     MODEL.TYPE    TYPE    CHARACTER(20);
```

(2). To add new data using INSERT operator.

```
INSERT
INTO      ATOM TABLE( ATOM.NAME, RADIUS, COLOR)
VALUES    ('H', 1.250, 'GREEN');
```

(3). To project one column using PROJECT operator.

```
PROJECT MOLE.NAME
FROM    MOLECULE;
```

(4). To select one tuple using SELECT operator.

```
SELECT   MOLE.NAME,  MODEL.TYPE
FROM     MOLECULE
WHERE    MOLE.NAME = 'SO2';
```

(5). To display whole table using DISPLAY operator.

```
DISPLAY ATOM TABLE;
```

(6). To select one tuple with specify attributes using SELECT operator.

```
SELECT  ATOM.1, ATOM.2, ATOM.3
FROM    TRIANGLE
WHERE   MOLE.NAME = 'SO2';
```

(7). Subquery with comparison operator to get one tuple.

```
SELECT  *
FROM    ATOM TABLE
WHERE   ATOM.NAME =
        (SELECT ATOM.1
         FROM   TRIANGLE
         WHERE  MOLE.NAME = 'SO2');
```

(8). To remove one tuple data using DELETE operator.

```
DELETE  GEOMETRY
WHERE   GEOM.NAME = 'TRIANGLE';
```

(9). To combine two relation using JOIN operator.

```
JOIN     MOLECULE, TRIANGLE
WHERE    MOLECULE.MOLE.NAME = TRIANGLE.MOLE.NAME;
```

(10). To describe properties of a relation using DESCRIBE.

```
DESCRIBE          ATOM TABLE
```

Actually, the CAMD system does not allow the user to employ this kind of command language, but the CAMD system does implement all of these operations in its internal processes. The system has been made more friendly to the user by using an interactive environment. It may ask user questions. It is not necessary to realize SQL-like language. CAMD does not support all relational operations. For example, in the PROJECT operation, the CAMD system only projects the first attribute from any one relation. It does not support projection of any other attributes, because it is not very important to project any other attributes to the user of the system. To project the primary key from any table, it is good enough to know how many molecules existed in the system, how many kinds of geometric model are offered in the system, or what kinds of atom existed in data base.

The other reason that system does not use the traditional query language is to avoid the user inserting data in one relation without inserting data to another relation. Suppose the user adds one tuple of new molecule data (SO2, TRIANGLE) to relation MOLECULE(MOLE.NAME, MODEL.TYPE). Later on, the system can not find the SO2 tuple in relation TRIANGLE(MOLE.NAME, ATOM.1, ATOM.2, ATOM.3). So the system automatically requests user input data for relation TRIANGLE. The CAMD system does not provide a parser system to analyze the query language according to the syntax. From the user's point of view, it is more convenient, since a user follows the menu to do data retrieval. Here is an example that employs a SQL-like language to illustrate the underlying operations. Assuming that a user wanted to display molecule SO2 (Sulfur dioxide) on the Vectrix graphic screen. Also, assuming that the CAMD system supports SQL-like query commands, then it will be performed as follows:

step 1. To find what is the geometric model type of molecule SO2 (Sulfur Dioxide):

```
SELECT   MODEL.TYPE
FROM     MOLECULE
WHERE    MOLE.NAME = 'SO2';
```

The result:

MODEL.TYPE
TRIANGLE

step 2. To find which atom locate on which position.

```
JOIN     MOLECULE AND TRIANGLE
WHERE    MOLECULE.MOLE.NAME = TRIANGLE.MOLE.NAME
GIVING   R1;
```

The result: relation R1

MOLE.NAME	MODEL.TYPE	ATOM.1	ATOM.2	ATOM.3
SO2	TRIANGLE	S	O	O

step 3. To get radius and color for first atom:

```
JOIN     R1 AND ATOM TABLE
WHERE    R1.ATOM.1 = ATOM TABLE.ATOM.NAME
GIVING   R2;
```

The result: relation R2

MOLE.NAME	MODEL.TYPE	ATOM.1	ATOM.2	ATOM.3	RADIUS	COLOR
SO2	TRIANGLE	S	O	O	1.850	BLUE

step 4. To get rid of unrelated atom:

```
SELECT  MOLE.NAME, MODEL.TYPE, ATOM.1, RADIUS, COLOR
FROM    R2
WHERE   MOLE.NAME = 'SO2'
GIVING  R3;
```

The result: relation R3

MOLE.NAME	MODEL.TYPE	ATOM.1	RADIUS	COLOR
SO2	TRIANGLE	S	1.850	BLUE

step 5. To find first set of (x, y, z) coordinate value:

```
SELECT  GEOM.NAME, X.1, Y.1, Z.1
FROM    GEOMETRY
WHERE   GEOM.NAME = 'TRIANGLE'
GIVING  R4;
```

The result: relation R4

GEOM.NAME	X.1	Y.1	Z.1
TRIANGLE	0.000	0.000	0.000

step 6. To get completed message for first atom.

```
JOIN     R3 AND R4
WHERE    R3.MODEL.TYPE = R4.GEOM.NAME
GIVING   R5;
```

The result: relation R5

MOLE.NAME	MODEL.TYPE	ATOM.1	RADIUS	COLOR	X.1	Y.1	Z.1
SO2	TRIANGLE	S	1.850	BLUE	0.0	0.0	0.0

After getting relation R5, the data of the first atom of molecule SO2 will be send to graphic system to draw first sphere on Vectrix. then the CAMD system repeats the same retrieval from step (3) to step (6) to get the second atom data and sends to graphic system to draw the second atom on Vectrix. Again, to repeat the rest of atoms until to finish drawing this molecule. Actually the CAMD system does not support these query language for user to code these commands. But the CAMD system still implicitly performs this processing.

The CAMD system maintains two support files, which are relation.tbl and attribute.tbl. "relation.tbl" is a flat file treated as a two-dimensional array of elements. It contains the name of all relations, the name of data file, a descriptor of data file, the name of index file, a descriptor of index file, the number of tuple in data file, the size of a tuple in the data file, the starting position of a tuple in the attribute.tbl file of the relation, the number of attribute in the relation, the end of file position in the index file, and the end of file position in the data file. The length of a relation.tbl tuple is 136 bytes in the system.

The layout of this data structure was designed in C language as follows:

```
struct  rel_buf
     {
        char    rel_name[MAXCHAR];
        char    data_name[MAXCHAR];
        int     data_fd;
        char    index_name[MAXCHAR];
        int     index_fd;
        int     d_tuple;
        int     rec_size;
        int     att_1st_rec_pos;
        int     att_numb;
        int     ieof_pos;
        int     deof_pos;       };
```

The other system support file, attribute.tbl, also can be treated like a relation. It contains the name of all the relations in the system, the name of all the attributes in the system, the data type of the attribute, the length of the attribute, and a flag indicating whether the attribute is primary key. The length of a attribute.tbl tuple is 80 bytes.

The data structure of attribute.tbl was designed as follows:

```
struct  att_buf
     {

        char    rel_name[MAXCHAR];

        char    att_name[MAXCHAR];

        int     att_type;

        int     att_count;

        int     is_key;
     };
```

2.2. To Display a Molecule in Data Pool.

The CAMD system defines twelve relations which are MOLECULE, GEOMETRY, ATOM TABLE, LINEAR, TRIANGLE, TRIGONAL PLANE, SQUARE PLANE, TRIGONAL BI-PYRAMID, TETRAHEDRAL, TRIGONAL PYRAMID, SQUARE PYRAMID, and OCTAHEDRAL. There are no duplicate relational names. Also, the system automatically defines attribute data types, attribute names, and attribute numbers. For example, in ATOM TABLE, there are three attribute fields for one tuple. The first attribute is a primary key to store atom name. It is string character data type. The second attribute is decimal data type to store radius of atom. The third attribute is string character data type to describe atom color. Each relation would be described in more detail later. Basically, this section describes how to create a molecule, how to display a relation data, how to draw a molecule, and how to delete a molecule. The following are implemented in a similar fashion: defining a relation, selecting a tuple, deleting a tuple, adding a tuple, displaying a whole table data, and describing a relation properties in relational data base management system.

To create a relational table, the system writes some properties of the relation to "relation.tbl" file and "attribute.tbl" file which are the system support files. There are three primitive data structures in the CAMD

system. These are integer type, real type, and character type. If the attribute data type is a character string, it also needs to be assigned a character number for that field. Each relation is associated with one data file and one index file. The "relation.tbl" file is composed of relation names and there relative properties: name of data file and index file, descriptor of data file and index file, total tuples in data file, size of a data record, and field of attribute number, end of file position in data file and index file. The "attribute.tbl" file is composed of relation name, all of attribute names, and data structure of each attribute.

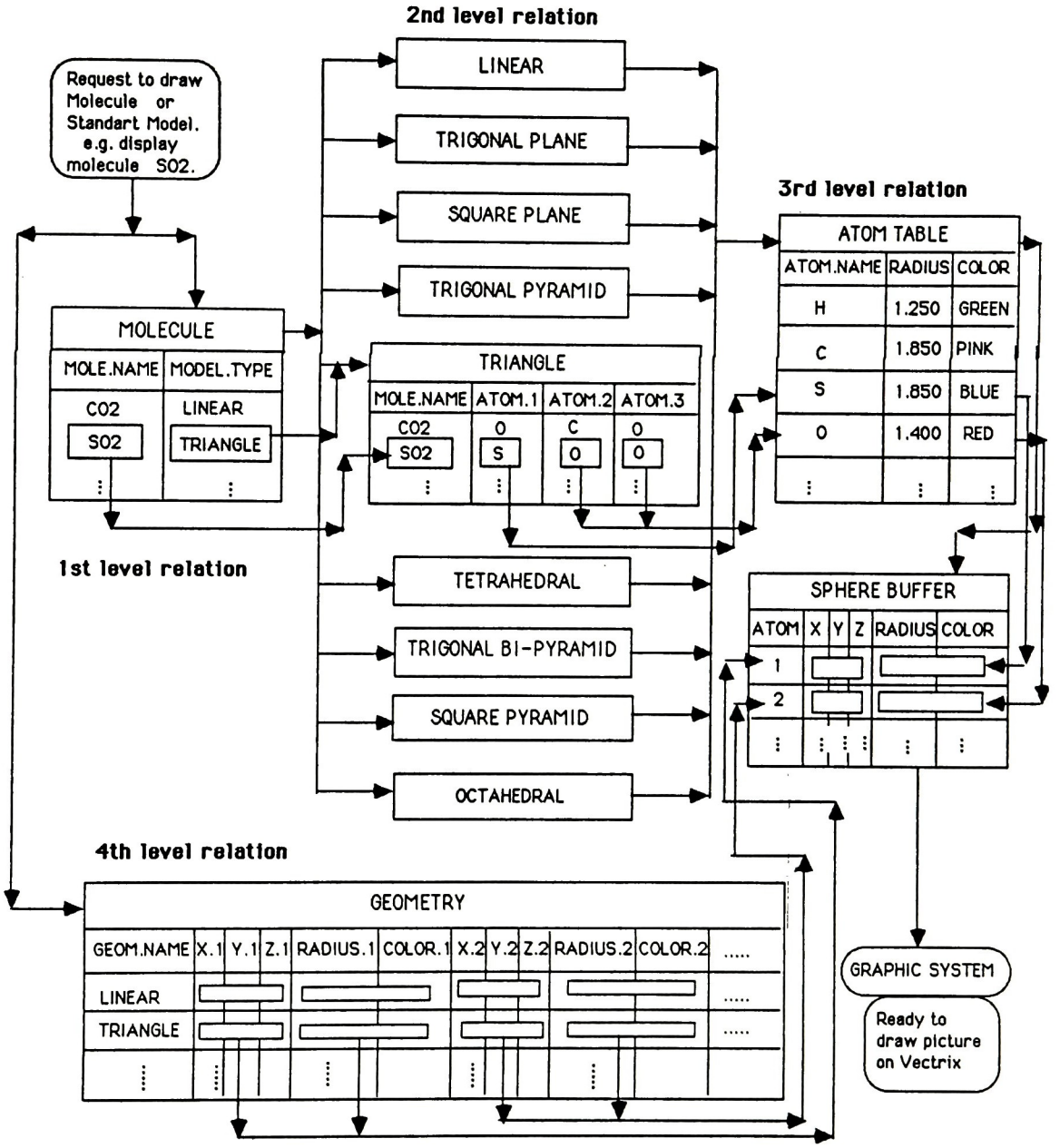
To add a tuple of data to one of the relations, the system will check whether this relation already exists in the CAMD system. Also, it will check if there is a duplicate key in a relation. If there is no duplicate, then it stores a tuple of data at the end of this data file. If the user of the CAMD system wants to create a new molecule, for example Sulfur dioxide (SO2), the system will go to the "MOLECULE" relation to check its primary key. If there is no duplicate SO2, it will store SO2 as its new key. And then the system asks the user to input another item of data for the next attribute field which in this case is "TRIANGLE". Also, the system will add to the relation "TRIANGLE" one tuple of its relative data as follows: SO2 for the first attribute which is the

molecular name, S for the second field which is the first atom in its geometric structure, O for the third field which means the second atom is O, and another O for the fourth field for the third atom.

At the beginning, the user has to store some basic data in the "ATOM TABLE" relation and the "GEOMETRY" relation for the future use. Each tuple in "ATOM TABLE" is composed of atom name, radius of atom, and color of atom. In the "GEOMETRY" relation, it is stored, geometric name, center in (x, y, z)-coordinate value of each sphere (atom), radius, and color. If some data needs to be changed in one tuple, then that tuple is deleted and the modified data is stored. If the system wishes to display the image of Sulfur dioxide (SO2) on the Vectrix graphic screen, it must perform the following operations. First, the system checks the first level relation "MOLECULE" to see whether SO2 exists or not. If it exists, then the system selects the corresponding tuple of data from "MOLECULE". The system can now retrieve more data from this tuple. The value of the tuples attribute "MODEL.TYPE" is in this case TRIANGLE. Then the system goes to find a relation called "TRIANGLE" in the relational data base. After this second level relation is found, the system will select a tuple of data from it whose primary key is SO2. If it exists, it then retrieves the rest of the data. In the second attribute of this data, the system retrieves S

which is the first atom in the molecular structure. The "ATOM TABLE", a third level relation is then searched to find the radius and color of S. The system keeps this data in a temporary buffer (sphere buffer), and goes back to the second level relation "TRIANGLE" to find the third attribute which is for our example O. Again, the system searches the third level relation "ATOM TABLE" to get the data for atom O and this data is stored into a temporary buffer again. This process is continued until the last attribute is reached in the second level relation "TRIANGLE". Finally, the system reaches the fourth level relation "GEOMETRY" to get the (x, y, z) coordinate value for each atom. The graphic system then draws the ball and stick representation of this molecule on the Vectrix screen. This implementation is shown in figure 2.1..

Figure 2.1. **To Display a Molecule in Relational Data Base.**



The data structure of sphere buffer is designed as follows:

```
struct  sphere_type
     {
        char    atom[10];

        double  x,
                y,
                z;

        double  radius;

        char    color[20];
      };
```

The string "TRIANGLE" is appeared in the MOLECULE relation, the TRIANGLE relation, and the GEOMETRY relation. It can be found that many relationships exist in the whole data base to control data integration. Because the "TRIANGLE" is a data of the second attribute in the first level relation (MOLECULE) and also is one of the relational name in the second level relation. Also, the "TRIANGLE" is one of the primary keys. in the "GEOMETRY" relation

In addition to displaying any molecule, the CAMD system also can draw all of the nine basic geometric models if requested. The system selects one tuple from the "GEOMETRY" relation and sends it to graphic system to draw this model. It is convenient for the user to realize what it looks like in order to decide which atom should be

located in which location during the creation of a new molecule. If the system needs to delete one tuple of data in relation "MOLECULE", it also has to delete another tuple in the second level relation, under the same primary key.

2.3. Primary Key Handling with Index Tree.

Each relation has its related index file and data file. The index file is used to store all primary keys from the relation. The data file is used to store all data objects. The system employs an index file to avoid duplicate keys in the data base system and to find the starting tuple position with a specified key in the data file. Each index file was created in as a tree structure. Each tuple looks contains five fields: lesser-pointer (left pointer), lesser-indicator (left indicator), index key, greater-pointer (right pointer), and greater-indicator (right indicator).

The index key is the primary key of the relation and treated as a root in a tree called index tree. This root has two left nodes and two right nodes. If an inserting key is equal or lesser than index key (a root of tree or subtree), the data of the inserting key will be put on lesser-pointer and lesser-indicator. If an inserting key is greater than index key, the data of inserting key will be put on greater-pointer and greater-indicator.

The purpose of the left pointer and right pointer is to store the value of the starting position of the tuple from data file or index file. The purpose of the left indicator and right indicator is to indicate that the left pointer or the right pointer points to the starting

position of a tuple in the data file or index file.

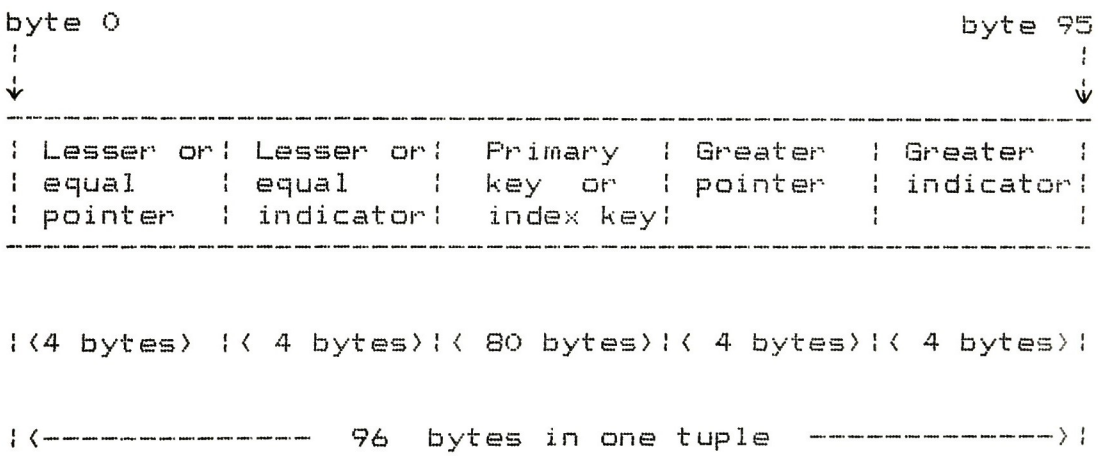
The layout of index file record was designed in the system coded with the C programming language as follows:

```
struct  index_record
     {
        int      leptr;
        int      letype;
        int      key[MAXSTR].
        int      gtptr;
        int      gttype;
     };
```

Suppose, a relation was created in the system called MOLECULE. It owns two attributes such as MOLE.NAME, MODEL.TYPE. In its data structure, MOLE.NAME was set to character type with 20 bytes of memory. Attribute MODEL.TYPE was set to character type with 20 bytes. Also, put a data_in_use control flag for each tuple at the beginning of each data tuple to see whether the data tuple is deleted or not. This control flag is an integer data structure and occupies 4 bytes. So the total number bytes of one tuple is 44 in its data file. The first tuple was stored from byte offset 0. The next tuple was stored from byte offset 44, and so on.

To store one tuple of the index file, the system needs 96 bytes of disk memory. Because one tuple of index file includes 80 bytes for primary key, 4 bytes for lesser-pointer, 4 bytes for storing lesser-indicator which is either index-indicator or data-indicator, 4 bytes for greater-pointer, and 4 bytes for storing greater-indicator which is either index-indicator or data-indicator.

One tuple of index file, Byte offset from 0



In the C programming language, The data structure of lesser-pointer, lesser-indicator, greater-pointer, greater-indicator are all set as an integer type. An integer data type occupies four bytes of space. The data structure of primary key is set as 80 characters type. The length of one tuple of data file has 44 bytes of disk memory. If the system wants to store six tuples in the

data file of the MOLECULE relation, then the starting position of each primary key in the data file of this relation is described for example as follows:

initial position (in bytes)	primary key
0	H2O
44	NH3
88	BeCl2
132	CO2
176	CH4
220	SO2
264	for next key

There are several steps to add new keys in the index file. Each time a new inserting key will compared with index key to decide how to store this data. This index file is stored as a tree structure called index tree.

The final conclusion of inserting six tuple will be:

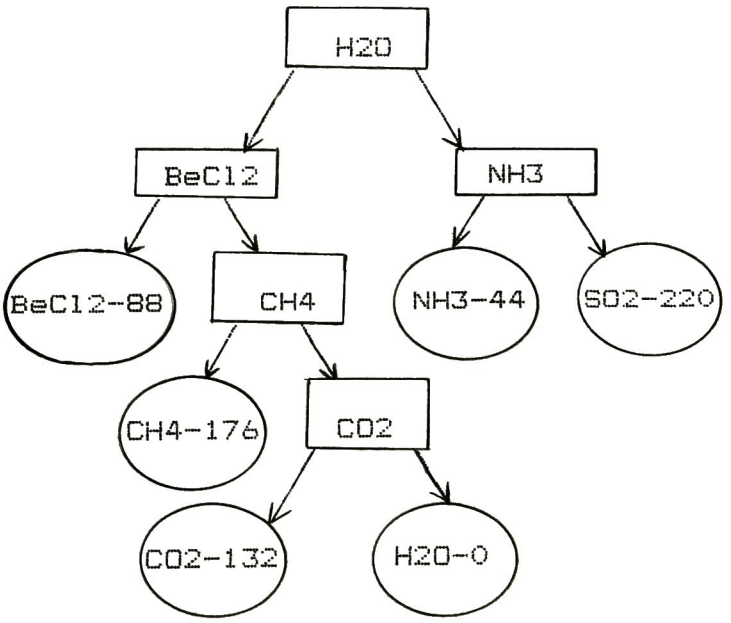
In index file:

byte offset in index file

↓

0	96	index type	H2O	384	index type
96	88	data type	BeCl2	288	index type
192	132	data type	CO2	0	data type
288	176	data type	CH4	192	index type
384	44	data type	NH3	220	data type

In index tree:



Mark:
Square sign represents index key or index node as a pointer.

Circle sign represents data node with byte value

for its allocation. (It will be leaf only in index tree).

2.4. The CAMD System Program Coding Chart.

The overall CAMD system design is illustrated in figure C-1. The system allows the user to reach the relational data base management system (R.D.B.M. system) to retrieve data from relational date base. Then the system will display data on terminal or enter graphic system to draw pictures on the vectrix screen. There are several program coding charts to explain how the CAMD system was designed from figure C-2 to C-14. These program charts are related to the following process.

(1). how to use this package.
(2). how to describe the properties of each relation.
(3). how to display the entire relation table.
(4). how to create a relation table.
(5). how to add data objects to data file.
(6). how to add primary keys to index file.
(7). how to design index tree to check duplicate key.
(8). how to list molecular name or list primary keys from any relation.
(9). how to delete a molecule or any one tuple in the relation.
(10). how to get molecular data and send to graphic system to draw picture on Vectrix graphics terminal.
(11). how to get a tuple of data from one relation to another relation.

(12).how to implement color look-up table,
and Vectrix commands.

(13).how to draw a sphere on graphic system.

(14).how to set surface shading (light intensity
on sphere surface).

The C.A.M.D. System Design.

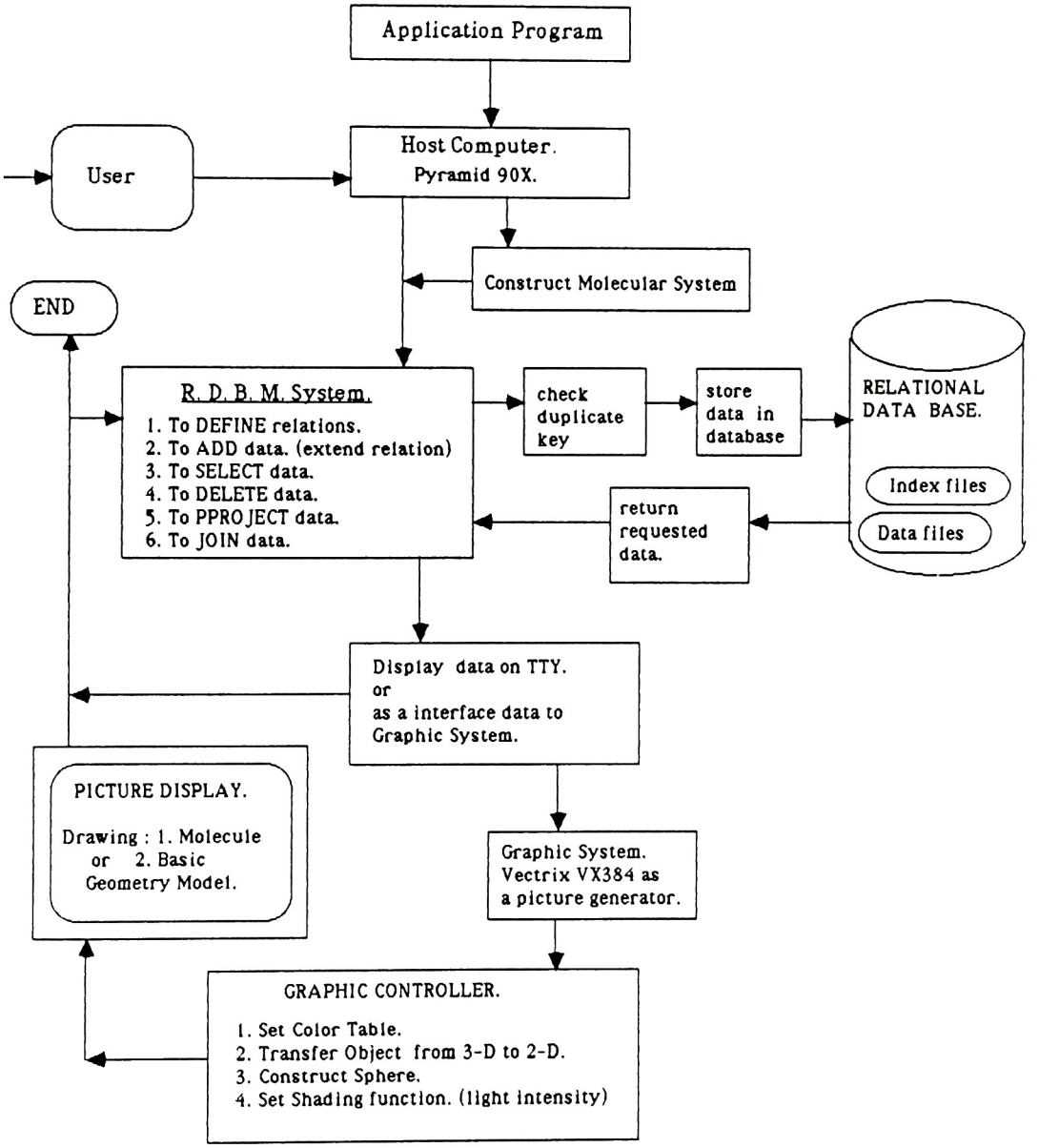


Figure. C - 1

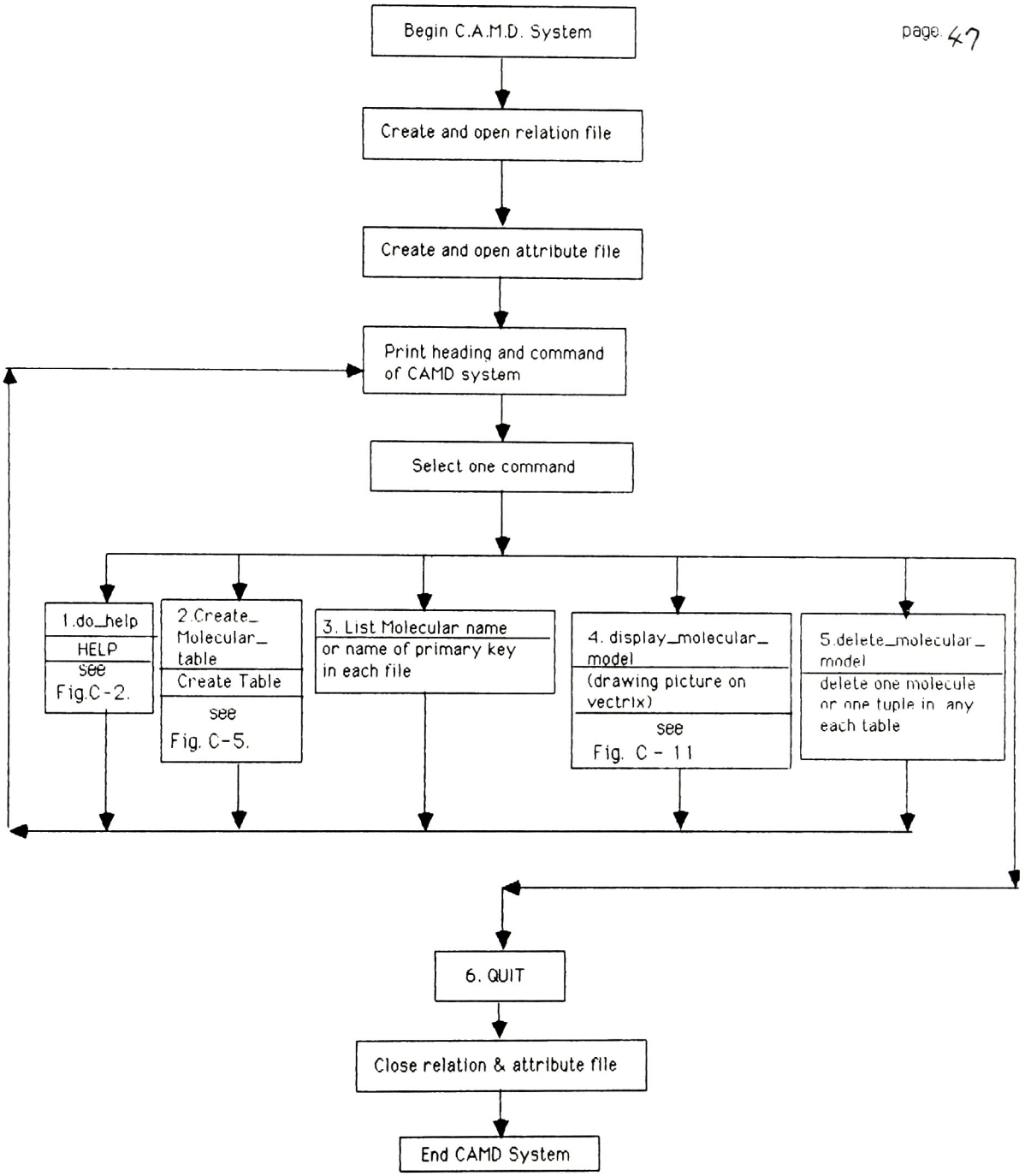


Figure C-2.

Main Program for Overall System.

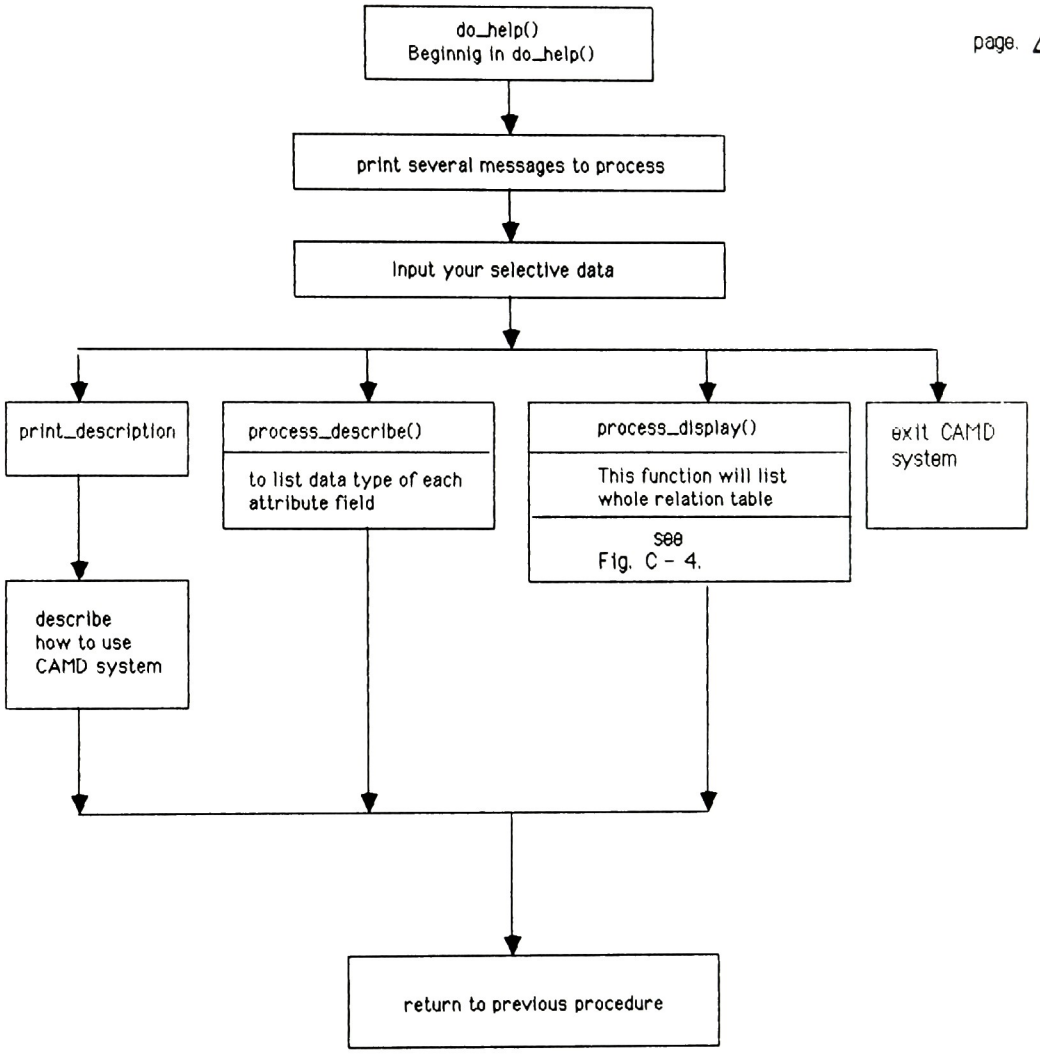


Figure. C-3.
To get Menu Description, to Describe and Display a Relation.

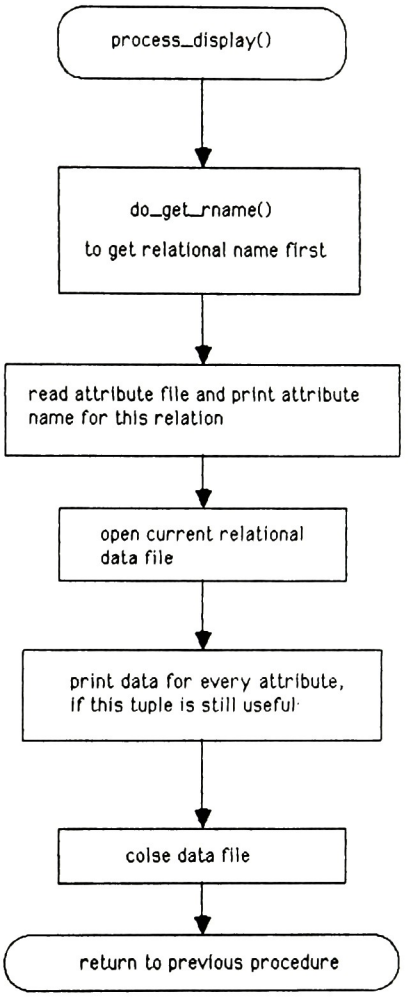


Figure C – 4.
To display whole table.

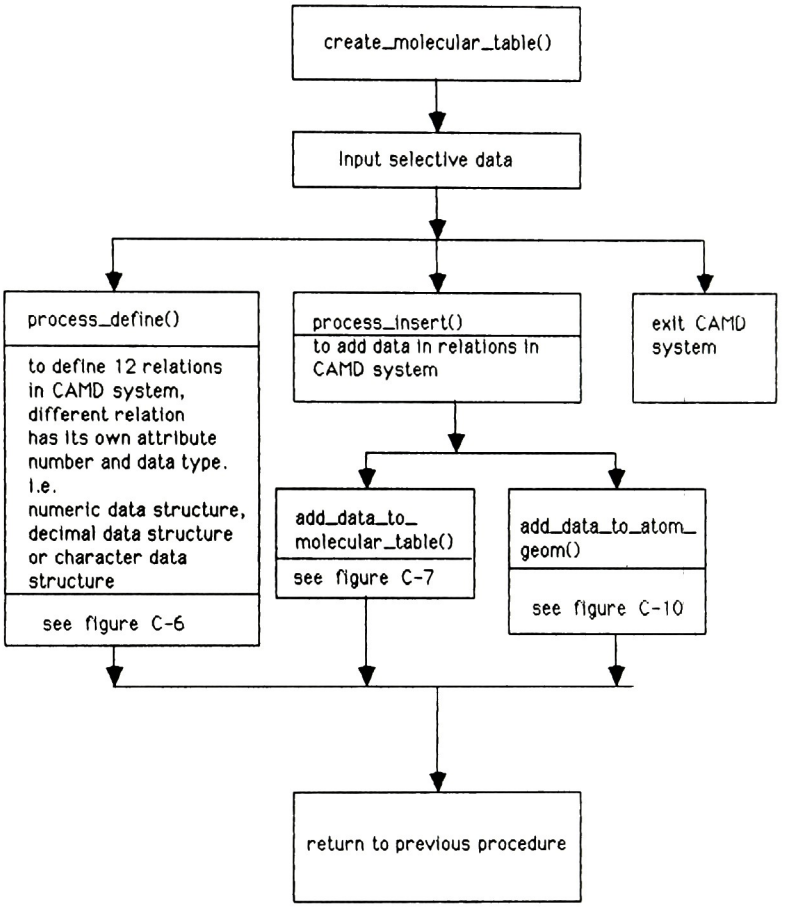


Figure C-5.
To Create Molecular Table:

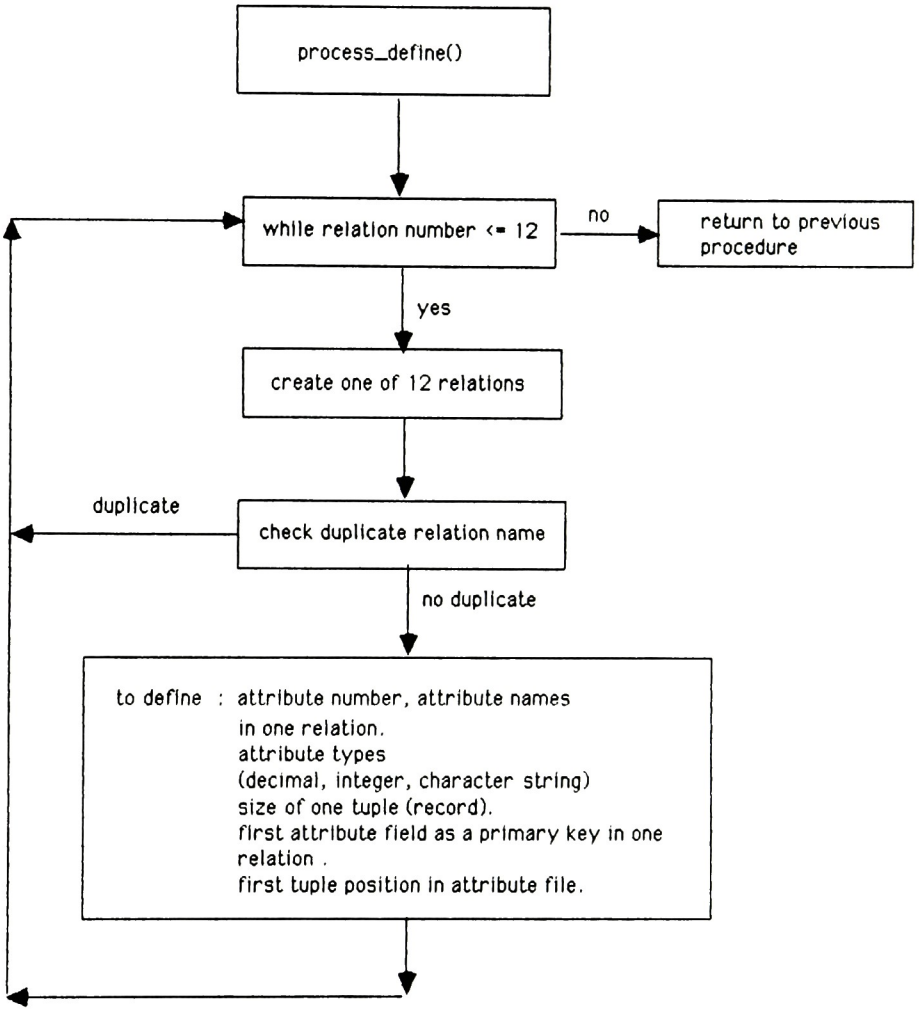


Figure C-6.

To Define a Relation :

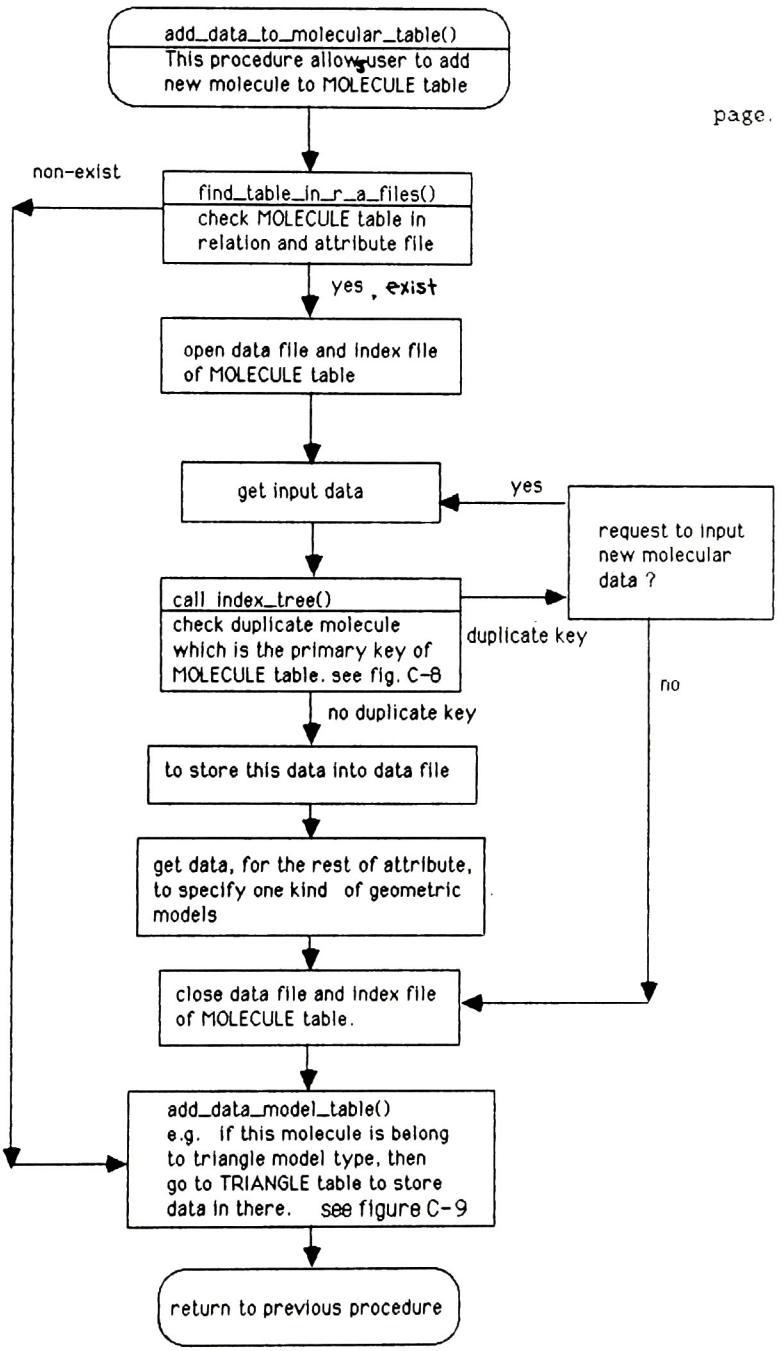


Figure C-7 . Add Data to MOLECULE Relation:

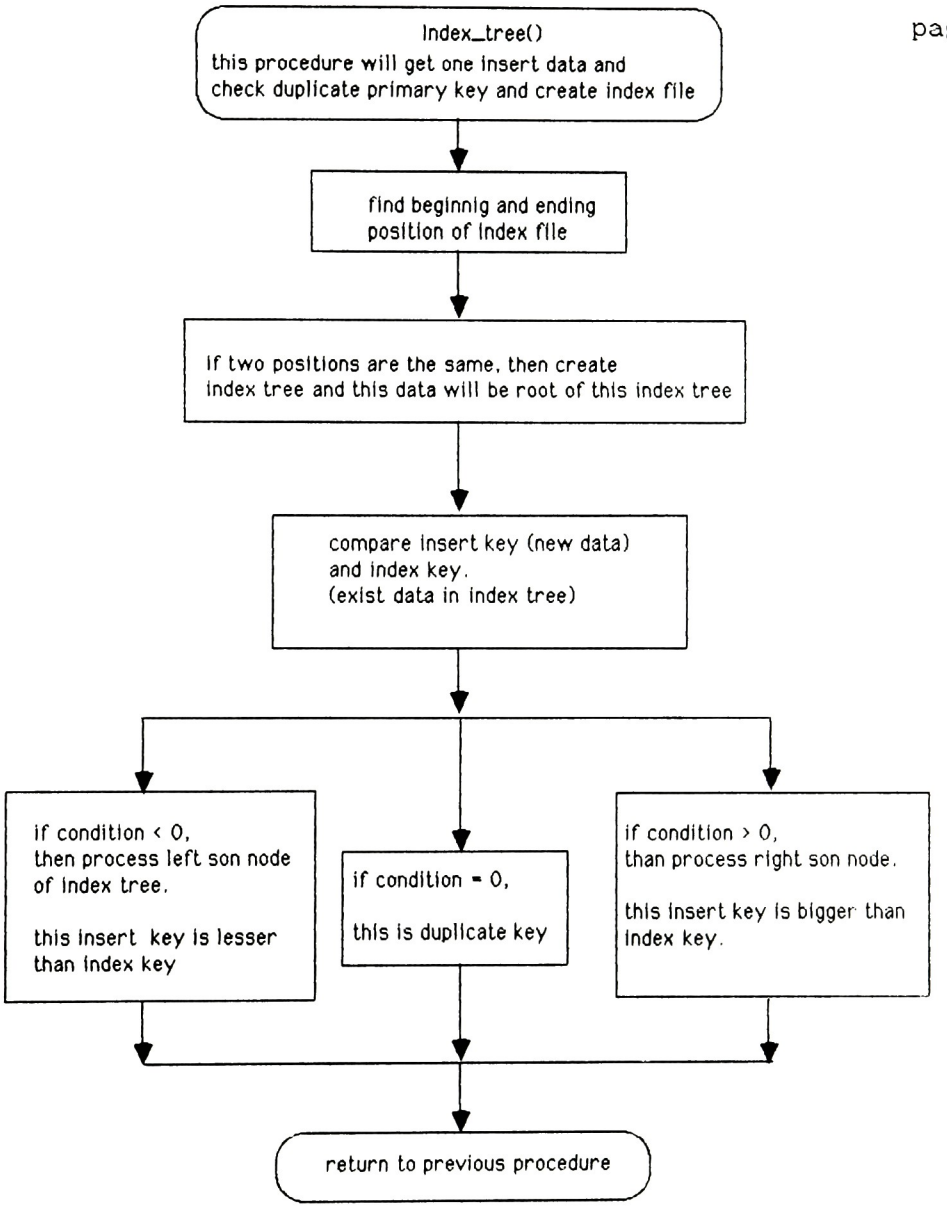


Figure C-8.

To Create Index Tree in Index File.

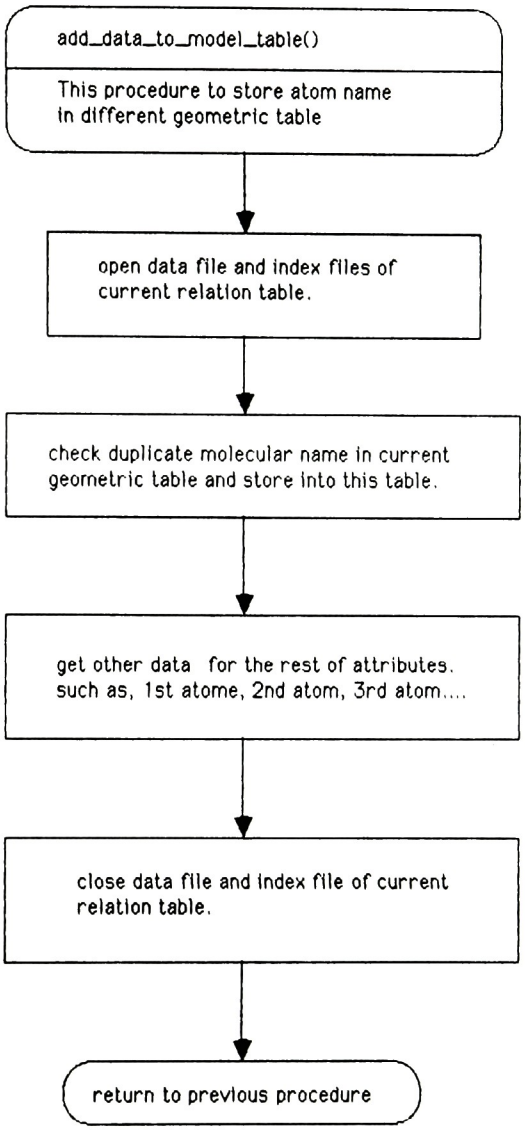


Figure C-9. Add data to specify model table:

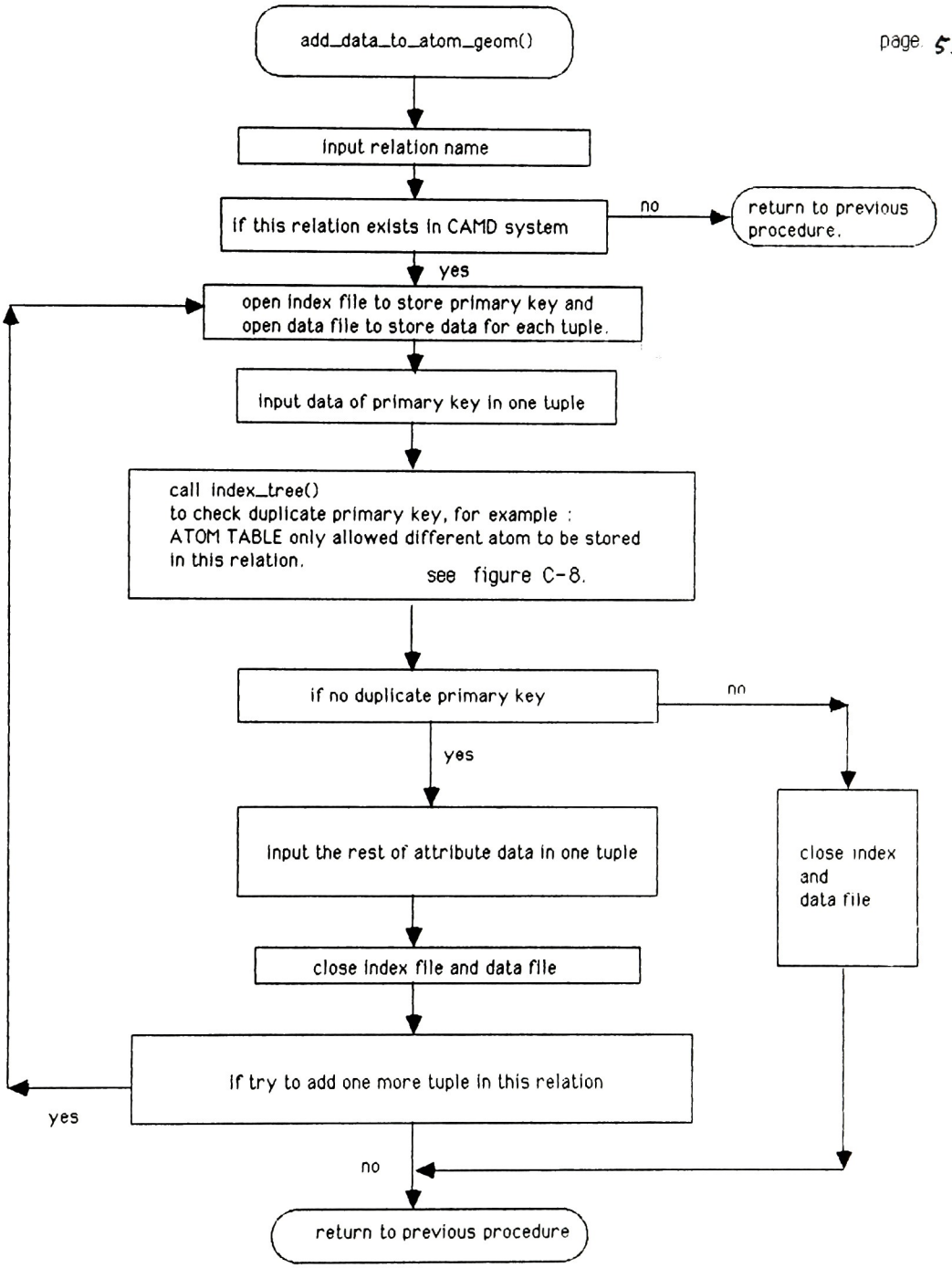


Figure C-10. Add data to ATOM TABLE and GEOMETRY relations.

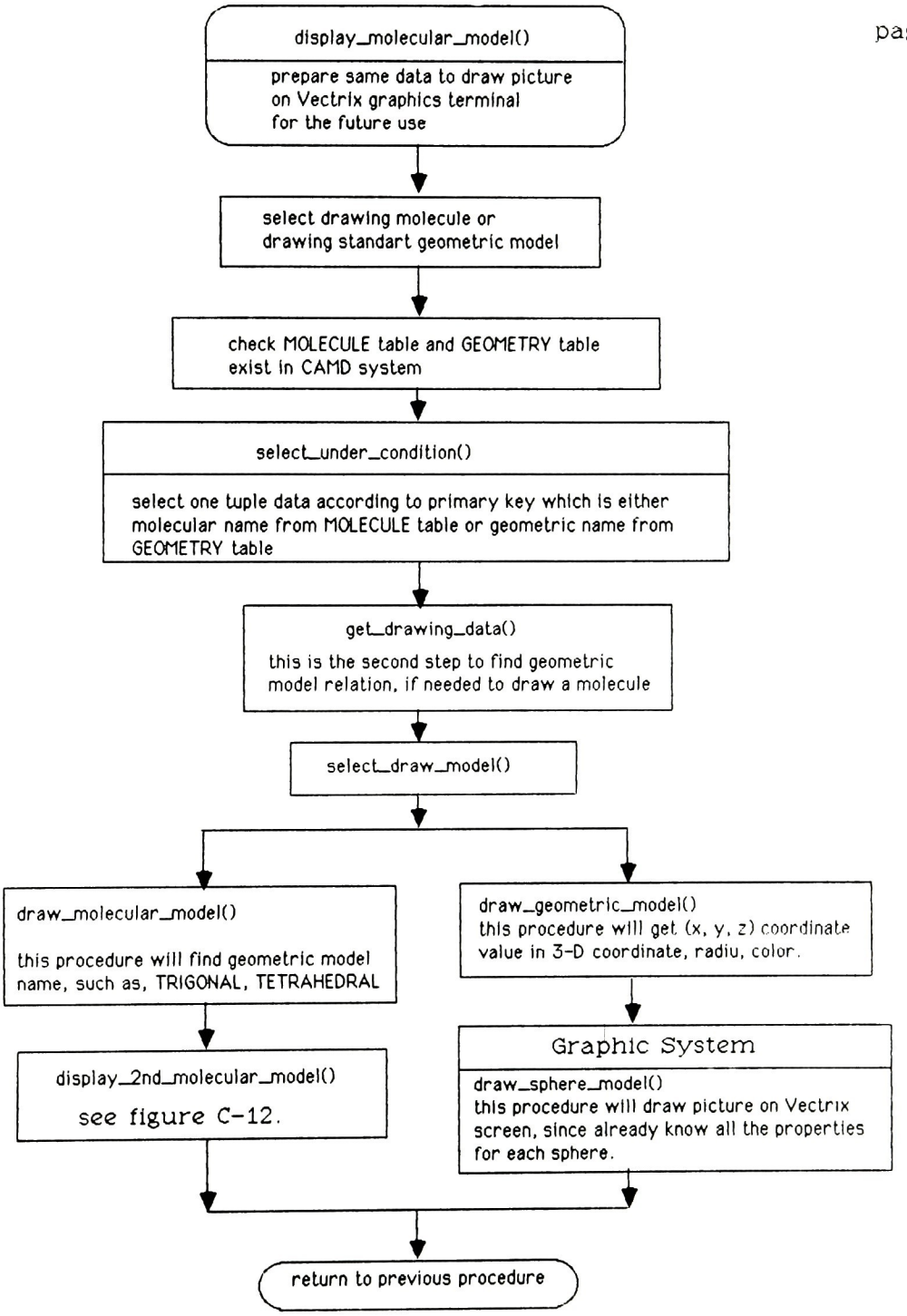


Figure C-11. To Display a Molecule or a Basic Geometric Model.

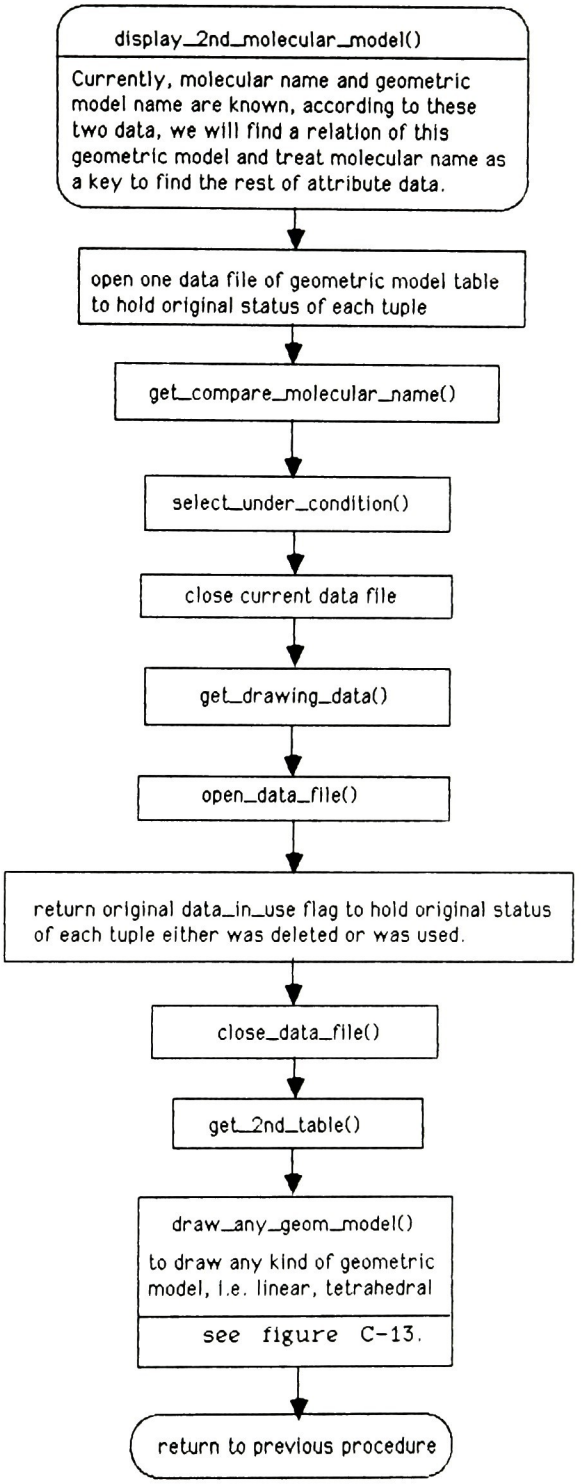


Figure C-12. **To Get Data from 2nd Level Relation.**

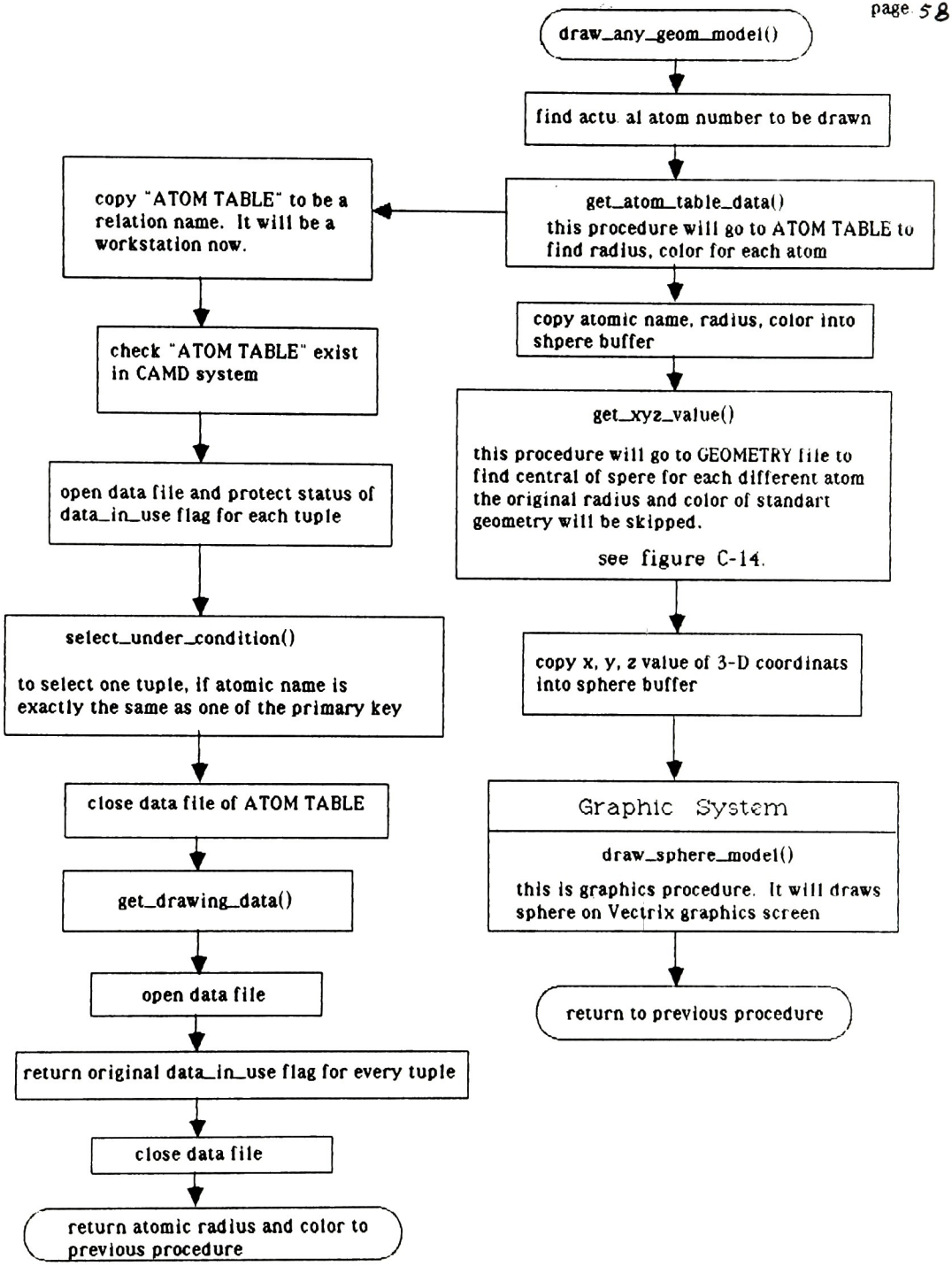


Figure C-13. To get data from 3rd level relation.

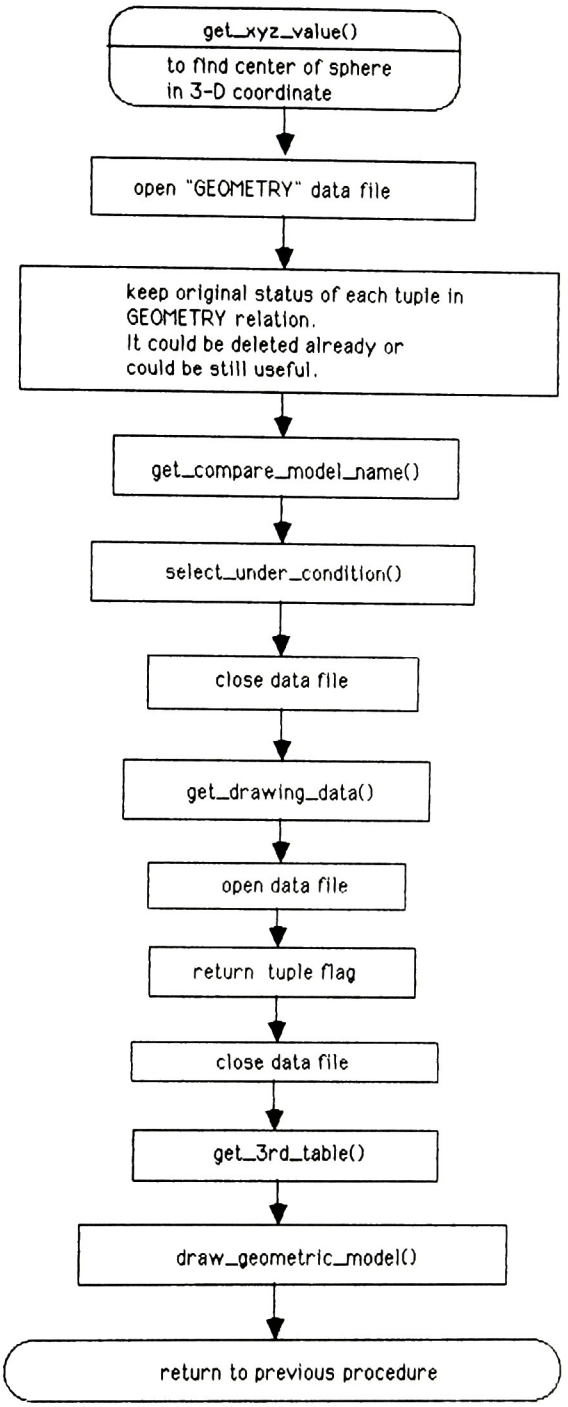


Figure C-14. **To Find Sphere Location in 3-D.**

3. Computer Graphics in the CAMD System.

3.1. Transformation and Projection.

The CAMD system handles models of three dimensional objects. The viewing surface is only two dimensional. The CAMD system considers ways of projecting its object onto this flat surface to form the image. The simplest object is, of course, the point. In two dimensions, the system can specify a point by establishing a coordinate system and listing the coordinates of the point as in figure 3-1. In three dimensions, the system will need an additional coordinate axis for the third dimension (three axes in all, one for height, one for width, and a third for depth as in figure 3-2.

The two dimensional image corresponds to a particular view of the three dimensional object. The process of finding which points on the flat screen correspond to the lines and surfaces of the object involves a viewing transformation.

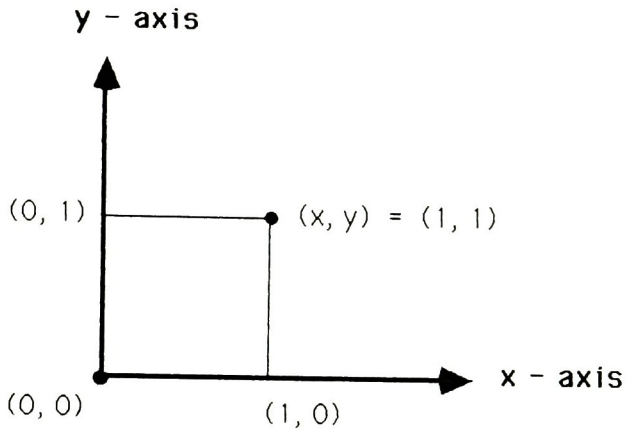


Figure 3-1 : position of the point (x, y) = (1, 1)
in 2-dimensional coordinate system.

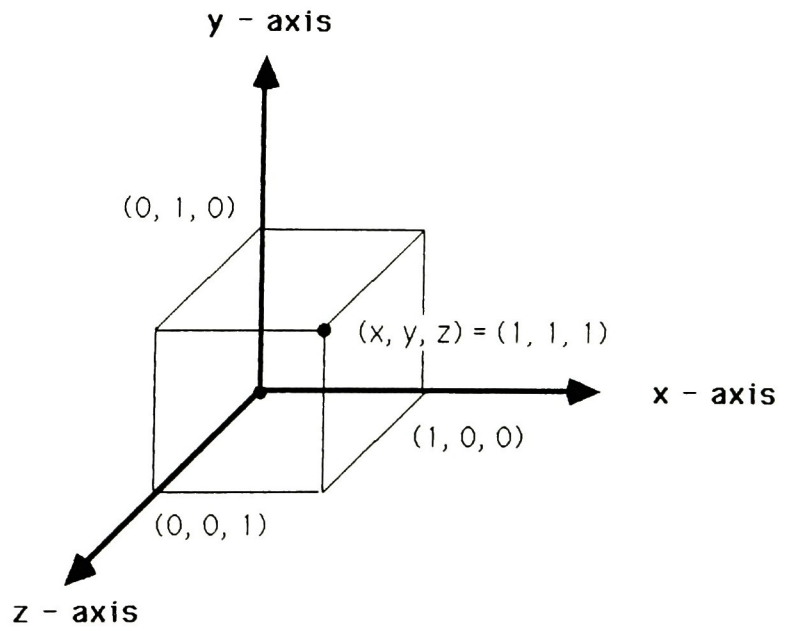
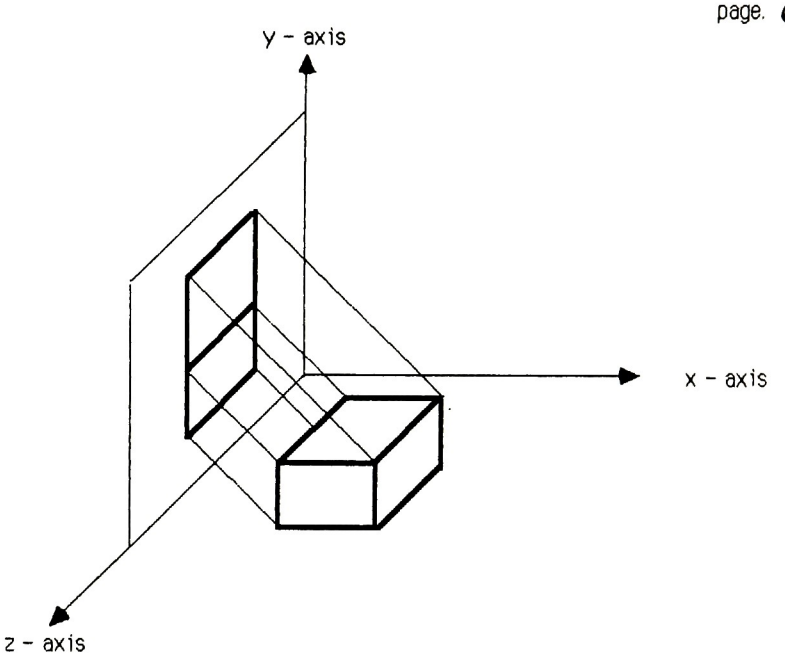


Figure 3-2 : position of the point (x, y, z) = (1, 1, 1)
in 3-dimensional coordinate system.

There are several ways of projecting a three dimensional object onto the two dimensional screen [8]. A parallel projection is formed by extending parallel line from each vertex on the object until they interest the plane of the screen. The point of intersection is the projection of the vertex. It connects the projected vertex by line segments which correspond to connections on the original object as in figure 3-3.

One special case of discarding the Z coordinate is the case where the screen, or viewing surface, is parallel to the xy plane, and the lines of projection are parallel to the z-axis. When one moves along these lines of projection, only the value of z coordinate changes; the values of x and y remain constant. So the point of intersection with the viewing surface has the same x and y coordinates as does the vertex on the object. The projected image is formed from the x and y coordinates, and the z value is discarded.

In a general parallel projection, any direction may be selected for the lines of projection. Suppose that the direction of projection is given by the vector [xp, yp, zp] and that the image is to be projected onto the xy plane. If there is a point on the object at (x, y, z), one wishes to determine where the projected point (x2, y2) will lie.



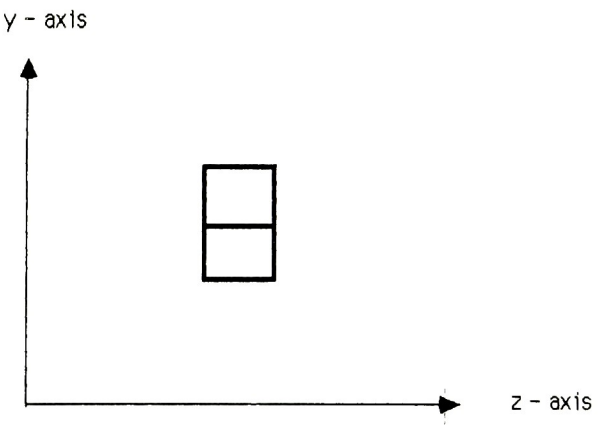


Figure 3-3 : A Parallel Projection.

The parallel projected point (x2,y2) is derived from the following two equations:

$$x2 = x1 - z1 \ (xp \ / \ zp)$$

$$y2 = y1 - z1 \ (yp \ / \ zp)$$

Alternatively one may use a perspective projection. In a perspective projection, the further away a object is from the viewer, the smaller it appears as is the case within the human eye. The lines of projection are not parallel. They all converge at a single point called the center of projection. This would be the paths of rays of light coming from the object to the viewer's eye. It is the intersections of these converging lines with the plane of the screen that determines the projected image.

Parallel and perspective projection can be used to form a two dimensional image from a three dimensional object as seen from the front. But sometimes, the viewer wish to view the object from the sides, or the top, or even from behind. We now discuss how can this be done.

All that the system need to do is to apply some rotation transformations before projecting. The system can think of the view plane (that is, the plane of our display surface) as fixed and the object as rotated or the system can picture the object as fixed and the view plane as repositioned. The system uses the film in a camera as

the view plane. and we can move the camera anywhere. So the system can view the object from any angle. In the CAMD system, the camera is fixed in one direction. The picture taken by this synthetic camera is what is shown on the display surface. as if the film is developed and stuck upon the screen.

In the CAMD system, (xp/zp) and (yp/zp) are set to be the constant (0.1), then all points (x1, y1, z1) in the 3-dimension coordinate system will be changed to (x2, y2) in the 2-dimension coordinate system from the following modified equations:

$$x2 = x1 - (0.1 * z1)$$

$$y2 = y1 - (0.1 * z1)$$

Under this condition, the viewing surface will be the direction of about 45 degree rotated y-axis on vectrix graphics screen. After then, the CAMD system transfers this original point from the left bottom of corner of screen to the central of screen and adjusts the size of Vectrix screen by 480 unit pixels according to the following formulas:

$$x = (480 * x2) + 336$$

$$y = (480 * y2) + 240$$

The viewer will see this origin point(0, 0, 0) on the Vectrix screen after above transformations as in figure 3-4. The camera is treated as a view plane. Also, this view plane will be the same as on the Vectrix screen, see in figure 3-5.

3.2. Changing Color Table Entries.

In some computer graphic applications, one may need precise manipulation of the color table to achieve special effects to display depth cues, surfacing, illumination shading and color balancing, for visual preference. Typical methods applying for these techniques include solid modelling, animation, paint programs, and graphics arts [9].

The CAMD system employs Vectrix commands to change color table entries on the Vectrix graphic system. The "Q" command allows the user to change any entry in the table to any value. The format is "Q i n r1 g1 b1 r2 g2 b2...". The "i" parameter indicates the first table entry to change. "n" indicates the total number of table entries to be changed, with "n" ranging from 1 to 512. The remaining values are the actual values to be inserted in each consecutive table entry for "n" entries. To change entries #56 (pure green in default table) to pure red, "Q 56 1 255 0 0" would be entered. Entering "Q 1 3 255 0 0 0 255 0 0 0 255" will change entries 1, 2, and 3 to pure red, green, and blue respectively.

The CAMD system sets up a color table in the Vectrix. The first three bits choose the color, The other 6 bits for the shade. There are eight kinds of pure color in the

system, these are black, red, green, yellow, blue, pink, cyan, and white. Since the Vectrix can display 512 colors simultaneously on the screen, the CAMD system uses 64 different color intensities for each pure color. For example, Greys are produced by mixing the three colors red, green and blue in even amounts (i.e. 30, 30, 30, or 62, 62, 62). The higher the numbers, the brighter the shade of grey. The value of black is (0, 0, 0). So to set up black we use "Q 1 64 0 0 0 1 1 1 2 2 2 ... 63 63 63". To set up the red entries in table locations from 65 to 128, The Vectrix command "Q" with "Q 65 64 4 0 0 8 0 0 12 0 0 ... 256 256 256" is used, and represents 64 different intensity of red. The color table entry 65 is light red. on the other hand, the entry 128 is dark red.

The entire color look-up table will be created by the CAMD system using the following vectrix commands forms:

Black will be:

```
Q  1  64  0 0 0  1 1 1  2 2 2  3 3 3 ... 63 63 63
```

Red will be:

```
Q  65  64  4 0 0  8 0 0  12 0 0  16 0 0 ... 256 0 0
```

Green will be:

```
Q  129  64  0 4 0  0 8 0  0 12 0  0 16 0 ... 0 256 0
```

Yellow will be:

```
Q  193  64  4 4 0  8 8 0  12 12 0  16 16 0 ... 256 256 0
```

Blue will be:

```
Q  257  64  0 0 4  0 0 8  0 0 12  0 0 16 ... 0 0 256
```

Pink will be:

```
Q  321  64  4 0 4  8 0 8  12 0 12  16 0 16 ... 256 0 256
```

Cyan will be:

```
Q  385  64  0 4 4  0 8 8  0 12 12  0 16 16 ... 0 256 256
```

White will be:

```
Q  449  64  4 4 4  8 8 8  12 12 12  16 16 16...256 256 256
```

3.3. Algorithm to draw a sphere.

This section describes how a sphere was drawn on the graphics system. The CAMD system puts a sphere in any location in the 3-dimension Cartesian coordinate system.

In figures, the Z-axis is perpendicular to the XY-plane or on the paper. The center of sphere is denoted by (cx, cy, cz). There are many planes passing through the sphere that includes its center. Lat A be the plane that parallel to the XY-plane, and includes (cx, cy, cz). Let B be the circle formed by the intersection of Plane A and the sphere, see figure S-1.

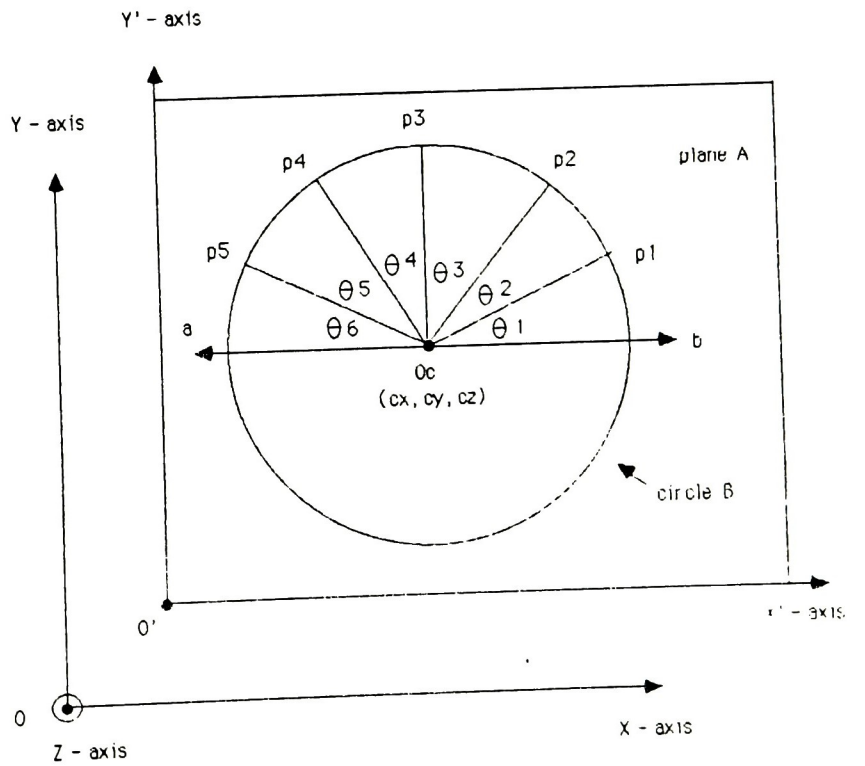


Figure S - 1 To divided upper semi-circle from a sphere in 3-D.

Let ab be the line through the center of circle B and parallel to the X-axis. The line ab divides circle B into two semi-circles, the upper semi-circle and the lower semi-circle. This upper semi-circle can be cut into many equal sections. In the CAMD system, it is cut into twenty-five equal partitions. But now, for the convenience of description, it is cut into six partitions. In other words, there are six equal angles θ1, θ2, θ3, θ4, θ5 and θ6 for these six partitions, and they describes five points p1, p2, p3, p4 and p5 on the upper semi-circle, see figure S-1. These five points define five planes that cut the sphere into four rings (or ring belts) and two shells, see figure S-2. These five planes must contain one of the five points and are parallel with the YZ-plane.

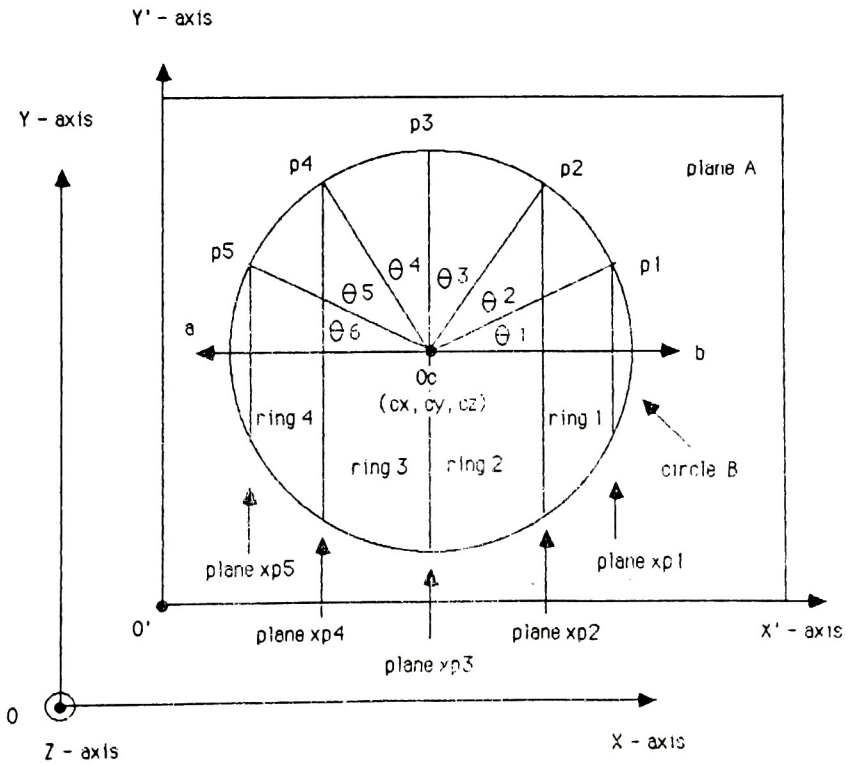


Figure S - 2.

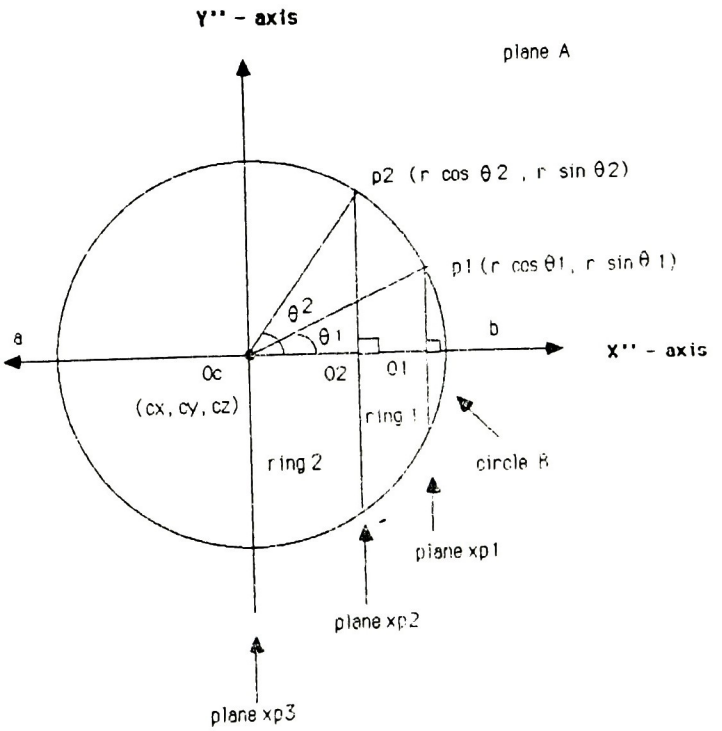
To produce four rings and shells from a sphere in 3-D.

The distance of point p1 from the center of the sphere is the radius of sphere. This distance is the same as from point p2 to the center of sphere. The values of p1(x, y) and p2(x, y) can be calculated using a Trigonal coordinate system. Here are the basic formula used in the CAMD system.

$$\cos(\theta) = \frac{x}{\sqrt{x^2 + y^2}} \qquad \text{and}$$

$$\sin(\theta) = \frac{y}{\sqrt{x^2 + y^2}}$$

Thus p1(x, y) = (r * cos (θ1), r * sin (θ1)) and p2(x, y) = (r * cos (θ2), r * sin (θ2)), where r is the radius of circle B, see figure S-3.



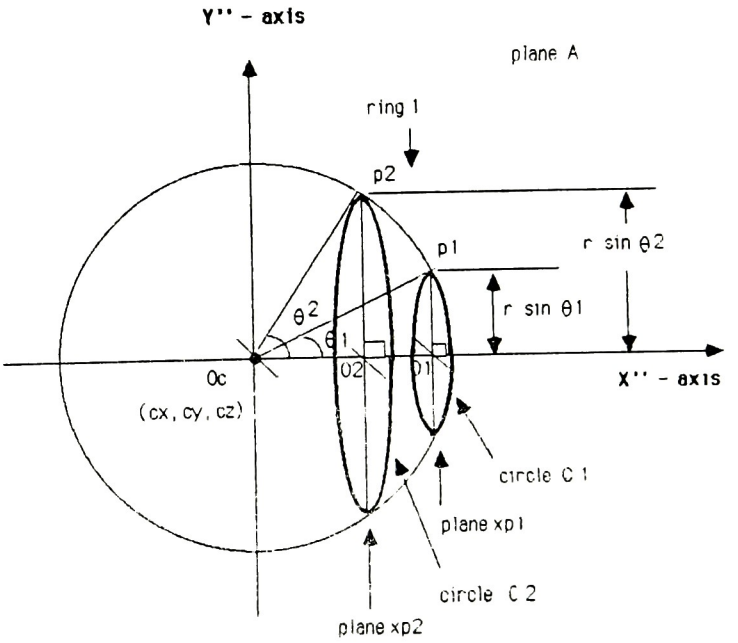
The distance of plane xp1 to plane yz is (r cos([illegible]) + cx)

The distance of plane xp2 to plane yz is (r cos([illegible]) + cx).

Figure S - 3.

To find p1 and p2 coordinate values.

Each ring bounded by two circles. For ring 1 let c1 and c2, be these two circles. The center of circle c1 is the point o1(r * cos (θ1) + cx, cy, cz). and the radius of circle c1 is r * sin (θ1). This value is exactly the same as the y value of point p1. The center of the circle c2 is the point O2 (r * cos (θ2) + cx, cy, cz). The radius of circle c2 is the height of point p2 which is r * sin (θ2). In 3-dimension coordinate system, the x values for each of the points on circle c1 are all the same value. It is (r * cos (θ1) + cx). This value is exactly the same as the x value of the center of this circle, because this circle c1 is parallel to the yz-plane, see figure S-3 and figure S-4.



The radius of circle C1 is r sin θ1 = r sin (π / 6).

The radius of circle C2 is r sin θ 2 = r sin (2 π / 6)

Figure S - 4.

To find two radius of circle C1 and circle C2.

If the xz-plane is rotated 90 degree in a clockwise direction. the X-axis will be perpendicular to the paper. This view is drawn in figure S-5.

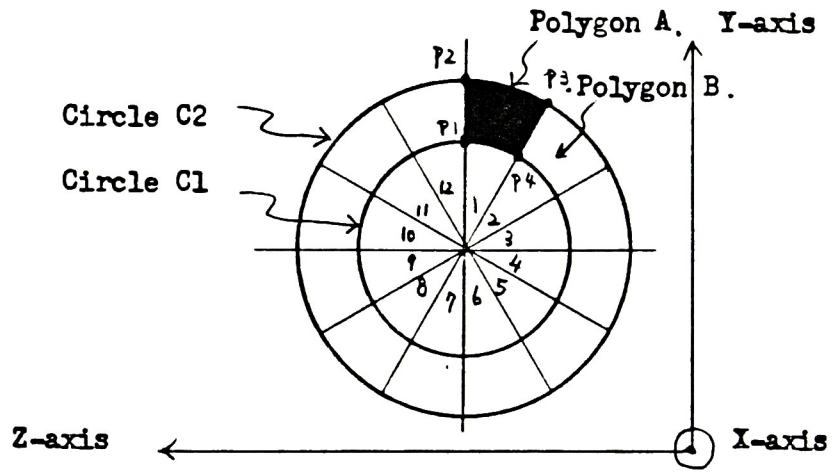


Figure S-5. A ring is divided into 12 equal facets.

Now it can be seen that each ring is made up of two circles. One is deeper and the other is closer to the viewer. The rings can be considered as being divided into a number of equal sections. In the CAMD system, each ring was cut into fifty pieces to make the surface of the sphere appear smooth. To make the description of the algorithm used by CAMD easier we will consider each ring to be divided into 12 sections. Thus, each piece makes an angle of (360 / 12) = 30 degrees to the x-axis, see figure S-5. Then we can find (x, y, z) value for point p3 and p4. The system connects point p1, p2, p3, and p4 to

become a four sides of polygon as shown in figure S-5. This polygon will be the rear part of the ring. This just like to draw deeper one first in z-value. Also, the system can calculate next eleven polygons step by step. Then according to different color intensive to draw each polygon. After the CAMD system has finished drawing one ring, the system will draw the next ring until the last shell of the sphere is drawn.

In the shell of sphere, each polygon only exists three sides. In other words, each polygon only employed three points to draw last part of sphere. The fourth point in new polygon is gone after the last cutting plane xp5 happened. The system will treat this shell as a ring which consists of 12 triangles as shown in figure S-6. Each triangle is a degenerated polygon. For example, polygon A has only three vertex, say p1, p2, and p3.

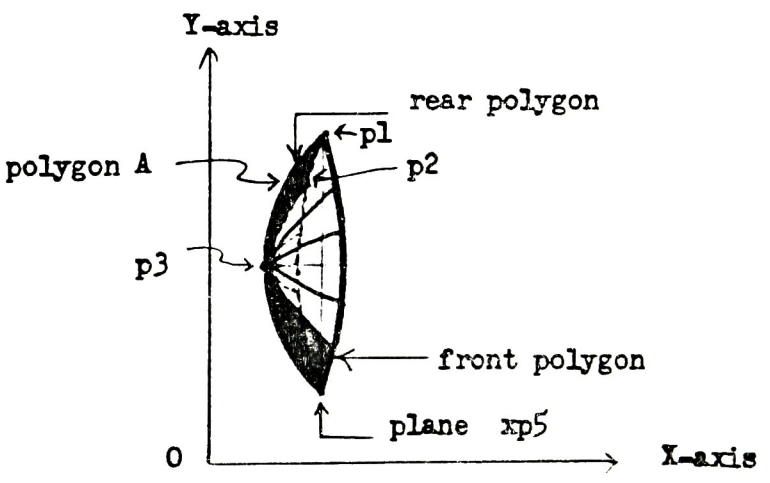


Figure S - 6. A Shell is divided into 12 equal triangles.

In any four sides of polygon, there are four points to produce this convex polygon A named p1, p2, p3, and p4 as shown in figure S-7 and figure S-8. To find the next four points for the connective convex polygon B, Point 2 and point 3 in polygon A are the first point and the fourth point in polygon B respectively.

Figure S - 7.

A structure of a ring.

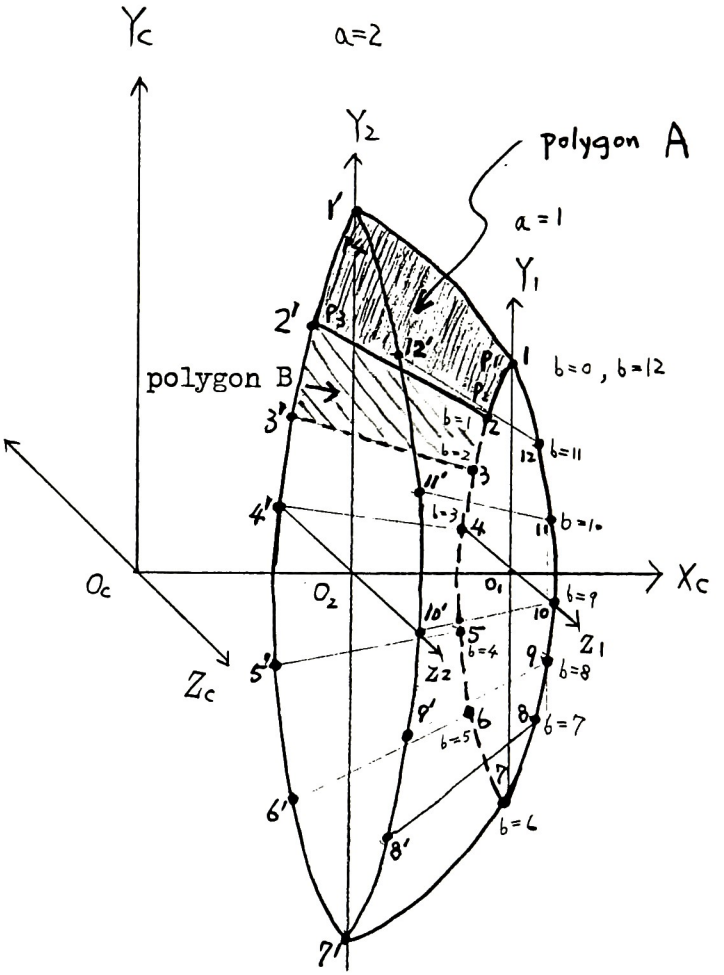
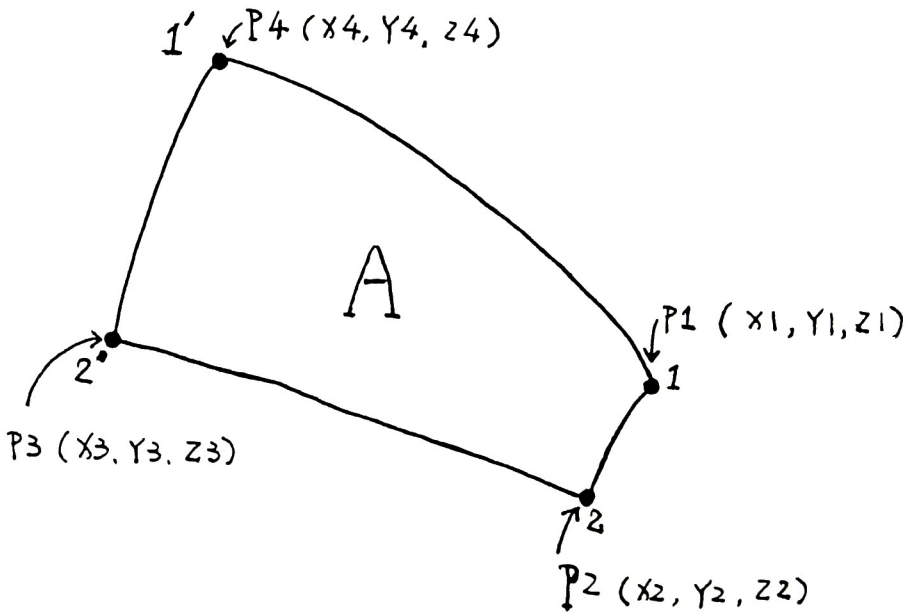


Figure S - 8.

Polygon A is the first polygon from the first ring of a sphere.



There are some coordinate values of four sets of four points in polygon A (p1p2p3p4) of first ring from figure S-7 and figure S-8. Assuming that this sphere is divided into six rings and each ring is divided in 12 facets. Let the coordinate of vertex be Xi, Yi, and Zi. So,

point 1:

x1 = xplane[1]

= r * cos ((3.14 * 1) / 6) + cx

= 0.1732

y1 = ring[1][0][0]

= (r * sin ((3.14 * 1) / 6)) * cos ((2 pi * 0) / 12) + cy

= 0.1000

z1 = ring[1][0][1]

= (r * sin ((3.14 * 1) / 6)) * sin ((2 pi * 0) / 12) + cz

= 0.0000

point 2:

x2 = xplane[1]

= r * cos ((3.14 * 1) / 6) + cx

= 0.1732

y2 = ring[1][1][0]

= (r * sin ((3.14 * 1) / 6)) * cos ((2 pi * 1) / 12) + cy

= 0.0866

z2 = ring[1][1][1]

= (r * sin ((3.14 * 1) / 6)) * sin ((2 pi * 1) / 12) + cz

= 0.0500

point 3:

x3 = xplane[0]

= r * cos ((3.14 * 2) / 6) + cx

= 0.1000

y3 = ring[0][1][0]

= (r * sin ((3.14 * 2) / 6)) * cos ((2 pi * 1) / 12) + cy

= 0.1500

z3 = ring[0][1][1]

= (r * sin ((3.14 * 2) / 6)) * sin ((2 pi * 1) / 12) + cz

= 0.0886

point 4:

x4 = xplane[0]

= r * cos ((3.14 * 2) / 6) + cx

= 0.1000

y4 = ring[0][0][0]

= (r * sin ((3.14 * 2) / 6)) * cos ((2 pi * 0) / 12) + cy

= 0.1732

z4 = ring[0][0][1]

= (r * sin ((3.14 * 2) / 6)) * sin ((2 pi * 0) / 12) + cz

= 0.0000

In figure S-3, The distance from Original to xp1-plane is ((r * cos ((1 * 3.14) / 6) + cx). The distance from Original to xp2-plane is ((r * cos ((2 * 3.14) / 6) + cx).

The radius of circle c1 is (r * sin 01) = (r * sin 3.14 / 6). and the radius of circle c2 is (r * sin 02) = (r * sin (2 * 3.14) / 6). A detail sphere diagram can be found in figure S-9 and two color plates A and B. Color plate A is in drawing status.

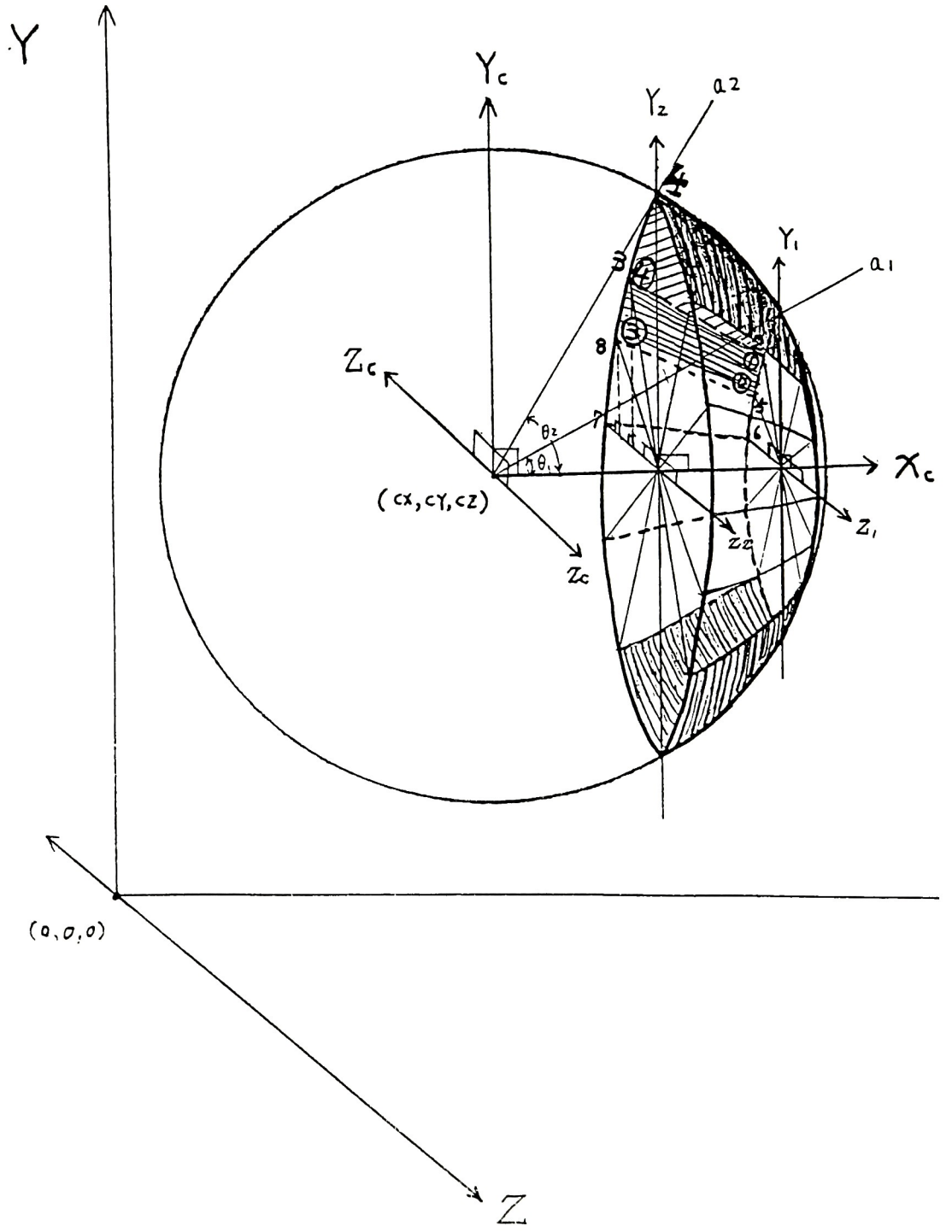


Figure S - 9. A Structure of Sphere.

3.4. Shading a Polygonal Facet in Sphere.

The next step toward achieving realism is the shading of the visible surface. The appearance of a sphere surface depends on the normal vector on each facet. The CAMD system draws polygons one by one to create a sphere with the appropriate color. The graphic system has to know how much the color intensity value should be for each polygon. The system assumes that the polygon is a planar object of uniform color and uses the first three points to determine the plane. These three points can produce two vectors in the same plane and the cross product produces a normal vector. The system uses this normal vector to determine the color intensity value for this facet of the sphere.

There are several properties of a vector that are used in this project. A vector is a directed line segment, characterized by its length and its direction only. If the initial point of $\vec{b}$ is the terminal point of $\vec{a}$. Then the sum of $\vec{a}$ and $\vec{b}$ is defined as the vector $\vec{c}$ drawn from the initial point of $\vec{a}$ to the terminal point of $\vec{b}$. It could be written

$$\vec{c} = \vec{a} + \vec{b}$$

The length of a vector $\vec{a}$, denoted by $|\vec{a}|$, is the distance between its initial and its terminal points. The notation

$-\vec{a}$ is used for the vector that has length $|\vec{a}|$ and whose direction is opposite to $\vec{a}$.

The direction of the three unit vectors $\vec{i}$, $\vec{j}$, $\vec{k}$ are the positive directions of coordinate axes. They are orthogonal unit vectors. The coordinate system is right-handed, which means that if a rotation of $\vec{i}$ into the direction of $\vec{j}$ through 90 degree corresponds to turning a right-handed screw. The vector $\vec{k}$ has the direction in which the screw advances. Usually, the Origin O of the coordinate system is the initial point of all vectors. Any vector $\vec{v} = [x, y, z]$ can be written as a linear combination of the unit vectors $\vec{i}$, $\vec{j}$, $\vec{k}$ as follows:

$$\vec{v} = x\,\vec{i} + y\,\vec{j} + z\,\vec{k}$$

The number x, y, z are sometimes called the elements of vector $\vec{v}$. The cross product of two vectors $\vec{v1}$, $\vec{v2}$ is another vector $\vec{v1} \times \vec{v2}$ that is perpendicular to each of the original two vectors. The direction for this vector $\vec{v1} \times \vec{v2}$ may be determined by imagining that the perpendicular line is grasped by the right hand with the fingers curling from $\vec{v1}$ to $\vec{v2}$. The right thumb then points in the correct direction for $\vec{v1} \times \vec{v2}$.

Now, suppose that A(x1, y1, z1), B(x2, y2, z2), C(x3, y3, z3), and D(x4, y4, z4) are the four points of a polygonal facet. The normal $(\vec{N})$ to the facet ABCD can be

obtained by finding the cross-product of adjacent vectors ($\overrightarrow{AB}$ and $\overrightarrow{AC}$) at vertice A. The normal to this plane at A is a straight line through A and perpendicular to the facet ABCD. So

$$\overrightarrow{AB} = [(x2 - x1) \quad (y2 - y1) \quad (z2 - z1)]$$
$$\overrightarrow{AC} = [(x3 - x1) \quad (y3 - y1) \quad (z3 - z1)]$$

The cross product of the two vector $\overrightarrow{AB}$ and $\overrightarrow{AC}$ is written

$$\overrightarrow{AB} \times \overrightarrow{AC} = ((x2-x1)\,\vec{i} + (y2-y1)\,\vec{j} + (z2-z1)\,\vec{k}) \times ((x3-x1)\,\vec{i} + (y3-y1)\,\vec{j} + (z3-z1)\,\vec{k})$$

which can be written

$$\overrightarrow{AB} \times \overrightarrow{AC} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ x2 - x1 & y2 - y1 & z2 - z1 \\ x3 - x1 & y3 - y1 & z3 - z1 \end{vmatrix}$$

where $\vec{i}$, $\vec{j}$, $\vec{k}$ are the unit vectors in the x, y, z directions, respective. This is a mnemonic aid rather than a true determinant, since the element of the first row are vectors instead of numbers.

Let, $\overrightarrow{AB} \times \overrightarrow{AC} = \overrightarrow{N} = [nx, ny, nz]$,

$\overrightarrow{N}$ is the so called surface normal vector of A.

The length or magnitude of surface normal vector $\vec{N}$ is

$$|\vec{N}| = \sqrt{(nx)^2 + (ny)^2 + (nz)^2}$$

and the surface unit normal vector $\vec{UN}$ is

$$\vec{UN} = \frac{\vec{N}}{|\vec{N}|} = \frac{\vec{AB} \times \vec{AC}}{|\vec{AB} \times \vec{AC}|}$$

$$= \frac{nx}{\sqrt{nx^2 + ny^2 + nz^2}}\vec{i} + \frac{ny}{\sqrt{nx^2 + ny^2 + nz^2}}\vec{j} + \frac{nz}{\sqrt{nx^2 + ny^2 + nz^2}}\vec{k}$$

Then, the system selects one vector $\vec{m}$ = [mx, my, mz]. The unit normal vector of m is $\vec{MV} = \vec{m} / |\vec{m}|$ = [MX, MY, MZ]. Then the system uses each surface unit normal vector $\vec{UN}$ to subtract the other unit normal vector $\vec{MV}$ to produce a new vector $\vec{NV}$.

so, the new vector is

$\vec{NV}$ =

$$\left[\left(\frac{nx}{\sqrt{nx^2 + ny^2 + nz^2}} - MX\right)\left(\frac{ny}{\sqrt{nx^2 + ny^2 + nz^2}} - MY\right)\left(\frac{nz}{\sqrt{nx^2 + ny^2 + nz^2}} - MZ\right)\right]$$

The vector length of $\vec{NV}$ is

$$|\overrightarrow{NV}| =$$

$$\sqrt{\left(\frac{nx}{\sqrt{nx^2 + ny^2 + nz^2}} - MX\right)^2 + \left(\frac{ny}{\sqrt{nx^2 + ny^2 + nz^2}} - MY\right)^2 + \left(\frac{nz}{\sqrt{nx^2 + ny^2 + nz^2}} - MZ\right)^2}$$

since, $|\overrightarrow{MV}| = \sqrt{MX^2 + MY^2 + MZ^2} = 1$

It is easy to see that

$$0 <= |\overrightarrow{NV}| <= 2$$

Let, $VecLen = |\overrightarrow{NV}| / 2$

Thus, VecLen has value between 0 and 1 and is used for percentage of full color intensity for the polygonal facet. The full color intensity was separated in 64 levels. If the value of VecLen is very small, the drawing color will be very bright. The following expressions are used to determine color intensity.

shade = (full color intensity) x (VecLen)

drawing color = (basic color) - shade

The CAMD system uses this technique to obtain the final drawing color for each facet of each sphere. The Vectrix graphic "F" command is used to fill the polygonal facet with this drawing color. The CAMD system displayed the spheres with color and illumination as shown in Color Plates 1-18.

3.5. Vectrix Tools in the CAMD System.

In the CAMD system, the Vectrix is assumed to be one of the TTY terminals treated as a Unix file. So the CAMD system has to open a file and allow messages to be sent to this file. The source code to draw a line from (x1, y1) to (x2, y2) in white on the Vectrix screen written in the C programming language is as follows.

```
#include <stdio.h>
#include <sys/file.h>
#define color_white 511
int x1, x2, y1, y2;
FILE  *fopen(),  *vtx;
main()
{
  vtx = fopen("/dev/ttyi26", "w");
  fprintf(vtx, "G");
  fprintf(vtx, "RE");
  fprintf(vtx, "KF");
  scanf("%d%d%d%d", &x1, &y1, &x2, &y2);
  fprintf(vtx, "C %d \n", color_white);
  fprintf(vtx, "M %d %d \n", x1, y1);
  fprintf(vtx, "L %d %d \n", x2, y2);
}
```

There are several vectrix commands used by the CAMD system. These tools are described as follows:

The C command (set color) is used for selecting a color for the next drawing operation. The C command sets the current drawing color. All drawing will use this color until the next C command is used. If a color value larger than 511 is specified, a modulus operation will be performed to determine the color. For example, "C512" will actually select color table index entry #0.

The E command (erase screen) clears the screen to a particular color. There are several functions for this command:

1. To erase the screen, Any image is overlaid.
2. To set the entire screen's color to the color number following the E.
3. To set the current drawing point to x = 0, y = 0 in 2-D mode or x= 0, y = 0, z = 0 in 3-D mode.

The F command (fill polygon) draws an "n" sided polygon and fills it with the specified fill color. The F command draws a polygon with "count" number of vertices at the specified x, y coordinates, or if in 3-D mode, at the x, y, z coordinates. Vertices may be specified either in clockwise or counterclockwise order. The Vectrix will

automatically fill the interior with the color specified by "fill color". The border of the polygon is drawn first, using the "fill color", The current drawing color is not changed. The final side of the polygon need not be specified because the Vectrix assumes the first vertex is also the last. After this command executes, the current drawing point becomes the first vertex. This applies in both absolute and relative mode. The polygon fill algorithm used is a "Y scan line convex polygon fill" and does not fill polygon with concave (squeezed in) areas on the top or bottom of the polygon.

The G command (go warmstart) resets most operating parameters. This command provides a way to reset all modes and many other conditions to a default state. This process is called a warmstart initialization. The conditions resets are Absolute coordinates, 2-D coordinates, Blank video mode, ASCII decimal transmission, Current drawing point to (0, 0), Color drawing mode, Default character set, Character magnification to 1, Character angle to 0, Character spacing to 7 (horizontal) and 0 (vertical), and 12 (line), Rectangular Fill Pattern to all ones, Viewport to full screen, 3-D transformation matrix to identity, Image pan to zero, Bit plane Mask Register to all ones, Pattern Register to all ones (a solid line)

The KF command (select flash video mode) sets flash mode for greater transmission speed. In flash mode, the display can be updated continually rather than only during the vertical retrace period - hence update is faster. However, the screen may flash short horizontal streaks as the image is changed.

The L command (line) draws a line from the current drawing points to the coordinates specified. In 2-D mode, L draws a line from the current drawing point to the x, y position on the screen. In 3-D mode, this command draws a line to x, y, z coordinate values in object space. The color of the line will be the current drawing color. After execution of this command, the current drawing point will be the endpoint specified in the command. This command is affected by the pattern register. To draw a solid line, the register must contain all ones.

The M command (move) moves the current drawing point to a new location. Similar to lifting a pen from the paper surface and moving it to another point, this command moves the starting point for the next command. Move is useful for specifying beginning points of lines, the seed points of complex fills, and the starting points of text. The current drawing point moves to the new (x, y) position on the screen without drawing any lines on the display.

The Q command (define color lookup table value) used to redefine the color lookup table create custom colors or to reorganize the table.

The RE command (REplace mode) used to replace an area with a new color, rather than mixing the new and overlaid color. The replace mode permits user to write over an existing image with a new one. It replaces an old image with a new one, without mixing the current and previous colors. This mode is slower than OR mode, because all enabled bitplanes must be written in order to replace the color.

4. Properties of Molecular Structure.

4.1. Prototype of Atom.

In the CAMD system there is a relation for storing all data about an atom in its relational data base. The name of the atom is the primary key of this table. The other two attributes store the radius and color of this atom. The radius of an atom (assuming a spherical shape) may be defined as the distance of closest approach to another atom and is the distance at which the mutual repulsion of the electron clouds and the mutual attraction of the nuclear charge of each for the electrons of the other are in equilibrium under specified circumstances. If the two atoms are the same, the radius of each is one half the internuclear distance; if they are unlike, the internuclear distance is the sum of the individual radii.

In general, the radii decrease from the beginning to end of any period, the rate of decrease being less the higher the number of the period. In the CAMD system, the value of atom was measured in angstroms, such as, the radius of hydrogen atom is 1.250 Å. In this description, angstroms (Å) are used for convenience of notation. (1 angstrom = 10^{-10} meter). Again, in general, the radii increase going down any column in the periodic table.

The following table is the structure of the atom file that exist in CAMD system.

relation name : ATOM TABLE

ATOM.NAME	RADIUS	COLOR
H	1.250	GREEN
Be	2.200	GREEN
B	2.170	GREEN
C	1.850	PINK
N	1.540	YELLOW
O	1.400	RED
F	1.350	CYAN
Al	2.530	WHITE
Si	2.240	GREEN
P	1.960	PINK
S	1.850	BLUE
Cl	1.800	PINK
Co	1.590	CYAN
Ni	1.600	CYAN
Cu	1.410	YELLOW
Ga	2.010	YELLOW
As	2.020	RED
Se	2.010	BLUE
Br	1.950	RED
Pd	1.630	GREEN
Sb	2.210	YELLOW
Te	2.220	PINK
I	2.150	BLUE
Xe	2.190	RED
Pt	1.830	YELLOW
Au	1.690	RED
Hg	1.470	CYAN

4.2. Prototype of Molecule.

This section roughly describes what a molecule looks like. A molecule is the smallest particle of a chemical substance capable of independent existence with retention of all its chemical properties. Molecules comprise one or more atoms which need not be of the same kind. Structurally, a more specific definition would be that a molecule is a local assembly of atomic nuclei and electrons in a state of dynamic stability. The cohesive forces are electrostatic, but, in addition, relatively small electromagnetic interactions may occur between the spin and orbital motions of the electrons, especially in the neighborhood of heavy nuclei.

In the CAMD system, there are nine geometric structures: linear, triangle, trigonal plane, square plane, trigonal pyramid, tetrahedral, trigonal bi-pyramid, square pyramid, and octahedral. There is a relation called MOLECULE to store all of molecules in the CAMD system. The name of molecule (MOLE.NAME) is the primary key of this table. There is another attribute which is MODEL.TYPE for specified type of molecule structure. whenever there is request to create a new molecule, to list some molecular name, to delete a molecule, or to draw a molecule. the system must go to this relation to check whether this molecule exists in MOLECULE relation or not.

Although the system allowed user to create these nine kinds of molecular structures, in fact, there are thousands of different models derived from these nine standard geometric structures and could exist in the system. For example, tetrahedron is one of these m2 Let's say, a tetrahedron is a figure having four faces, each of which is an equilateral triangle. The nucleus is at the center and the axes extend out to the corners. These four corners could be composed of many different of atoms to become different molecules depended on their radius and color of atom. such as, CH4, CHCl3, CH2ClF, and CCl4. These are all different molecules, so they are all allowed to be stored in the relational data base. The following table is one of relations in the system. The lists is the molecules that currently exist in the CAMD system. Also, some molecules can be found in color plate from 10 to 18.

relation name : MOLECULE

MOLE.NAME	MODEL.TYPE
BeCl2	LINEAR
CO2	LINEAR
HgCl2	LINEAR
HCN	LINEAR
XeF2	LINEAR
SO2	TRIANGLE
S2O	TRIANGLE
H2O	TRIANGLE
SCl2	TRIANGLE
NH3	TRIGONAL PYRAMID
NF3	TRIGONAL PYRAMID
NI3	TRIGONAL PYRAMID
PCl3	TRIGONAL PYRAMID
XeO3	TRIGONAL PYRAMID
BF3	TRIGONAL PLANE
BCl3	TRIGONAL PLANE
SO3	TRIGONAL PLANE
GaI3	TRIGONAL PLANE
OCF2	TRIGONAL PLANE
XeF4	SQUARE PLANE
[PdCl4]--	SQUARE PLANE
[PtCl4]--	SQUARE PLANE
[AuCl4]-	SQUARE PLANE
CH4	TETRAHEDRAL
XeO4	TETRAHEDRAL
CHCl3	TETRAHEDRAL
SiF4	TETRAHEDRAL
BrO4-	TETRAHEDRAL
NH4+	TETRAHEDRAL
FeCl4-	TETRAHEDRAL
CH2ClF	TETRAHEDRAL
PCl5	TRIGONAL BI-PYRAMID
SbCl5	TRIGONAL BI-PYRAMID
PF5	TRIGONAL BI-PYRAMID
NiPBrS3	TRIGONAL BI-PYRAMID
OSF4	TRIGONAL BI-PYRAMID
BrF5	SQUARE PYRAMID
TeF5-	SQUARE PYRAMID
[SbF5]--	SQUARE PYRAMID
SF6	OCTAHEDRAL
CFe6	OCTAHEDRAL
[SiF6]--	OCTAHEDRAL
AsF6-	OCTAHEDRAL

4.3. Prototype of Geometric Model.

In the GEOMETRY table, the system keeps X, Y and Z coordinate values, the standard radius, and color for each sphere in different standard geometric model. These data have already existed in the CAMD database system. The first attribute (GEOM.NAME) is the name of geometric model, the second attribute (X.1) is the value of the X coordinate value of the first sphere. The third attribute (Y.1) is the Y coordinate value of the first sphere. The fourth attribute (Z.1) is the Z coordinate value of the first sphere. The fifth attribute (RADIUS.1) is the value of radius of the same sphere. The sixth attribute (COLOR.1) is the color of the same sphere, too. The following five attributes will be the data of second sphere, and so on.

If the CAMD system wants to draw a standard tetrahedral geometric model, it uses the GEOMETRY relation to select a tuple whose primary key is tetrahedral. It then gets five attribute sets of data (X, Y, Z, RADIUS, COLOR) for the five spheres. After this, the system will draw five spheres on the Vectrix graphic screen according to different location, radius, and color. The following GEOMETRY table contains all of the data of the nine geometric models and shows us where the center location of each sphere is. Also, these nine standard geometric model can be seen in color plate from 1 to 9.

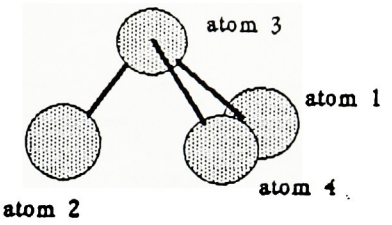
relation name : GEOMETRY

GEOM.NAME	X.1	Y.1	Z.1	RADIUS.1	COLOR.1
	X.2	Y.2	Z.2	RADIUS.2	COLOR.2
	X.3	Y.3	Z.3	RADIUS.3	COLOR.3
	X.4	Y.4	Z.4	RADIUS.4	COLOR.4
	X.5	Y.5	Z.5	RADIUS.5	COLOR.5
	X.6	Y.6	Z.6	RADIUS.6	COLOR.6
	X.7	Y.7	Z.7	RADIUS.7	COLOR.7
OCTAHEDRAL	0.300	0.000	-0.600	1.600	BLUE
	-0.300	0.000	-0.600	1.600	GREEN
	0.000	0.300	0.000	1.600	RED
	0.000	0.000	0.000	1.600	PINK
	0.000	-0.300	0.000	1.600	GREEN
	0.300	0.000	0.600	1.600	CYAN
	-0.300	0.000	0.600	1.600	YELLOW
SQUARE PYRAMID	0.300	0.000	-0.600	1.600	YELLOW
	-0.300	0.000	-0.600	1.600	BLUE
	0.000	0.300	0.000	1.600	GREEN
	0.000	0.000	0.000	1.600	RED
	0.300	0.000	0.600	1.600	CYAN
	-0.300	0.000	0.600	1.600	PINK
	-99.000	-99.000	-99.000	0.000	*
TRIGONAL BI-PYRAMID	0.150	0.000	-0.259	1.600	BLUE
	0.000	0.300	0.000	1.600	GREEN
	0.000	0.000	0.000	1.600	RED
	-0.300	0.000	0.000	1.600	PINK
	0.000	-0.300	0.000	1.600	CYAN
	0.150	0.000	0.259	1.600	YELLOW
	-99.000	-99.000	-99.000	0.000	*
TETRAHEDRAL	0.150	-0.300	-0.259	1.600	PINK
	0.000	0.300	0.000	1.600	GREEN
	0.000	0.000	0.000	1.600	CYAN
	-0.300	-0.300	0.000	1.600	YELLOW
	0.150	-0.300	0.259	1.600	RED
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*

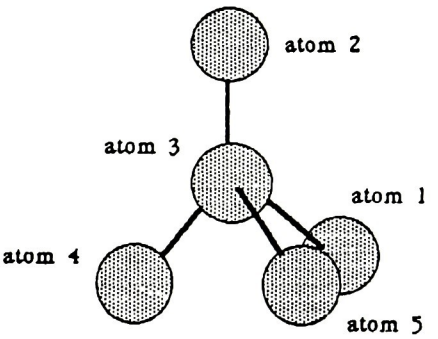
TRIGONAL PYRAMID	0.150	0.000	-0.259	1.600	CYAN
	-0.300	0.000	0.000	1.600	YELLOW
	0.000	0.300	0.000	1.600	PINK
	0.150	0.000	0.259	1.600	GREEN
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*
SQUARE PLANE	0.300	0.000	-0.600	1.600	BLUE
	-0.300	0.000	-0.600	1.600	YELLOW
	0.000	0.000	0.000	1.600	RED
	0.300	0.000	0.600	1.600	GREEN
	-0.300	0.000	0.600	1.600	PINK
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*
TRIGONAL PLANE	0.150	0.000	-0.259	1.600	YELLOW
	0.000	0.000	0.000	1.600	RED
	-0.300	0.000	0.000	1.600	GREEN
	0.150	0.000	0.259	1.600	PINK
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*
TRIANGLE	0.000	0.000	0.000	1.600	CYAN
	-0.300	0.000	0.000	1.600	GREEN
	0.150	0.000	0.259	1.600	RED
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*
LINEAR	0.300	0.000	0.000	1.600	PINK
	0.000	0.000	0.000	1.600	YELLOW
	-0.300	0.000	0.000	1.600	GREEN
	-99.000	-99.000	-99.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*
	0.000	0.000	0.000	0.000	*

The following nine tables show where atoms are located in each molecule. For instance, in the TETRAHEDRAL table (or relation), the CAMD system stores molecular names as the first attribute, and the rest of the five attributes will be stored with the atom name. When the user inputs new molecular data, they must specify which atom belongs to which location of molecule. The CAMD system has already defined the atom's order in each geometric model. These atom allocating rules are shown on the figure M-1, and figure M-2. For example, molecule Xenon Tetraoxide (XeO4), has Xe atom in the center of this tetrahedral model and O atoms in four ends. Atom 1 is located at rear of tetrahedral model. Atom 2 is located at the top of this model. Atom 3 is located at the center of this model. Atom 4 is located at left corner of this model from figure M-1. Atom 5 is located at front corner of this model. The drawing priority is from atom 1 to atom 5.

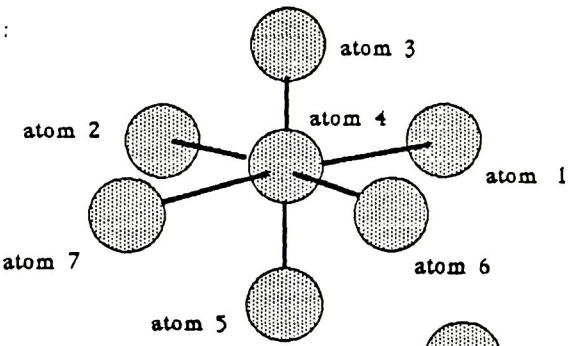
A Standard
Trigonal Pyramid Model :



A Standard
Tetrahedral Model :



A Standard
Octahedral Model :



A Standard Trigonal
Bi-Pyramid Model :

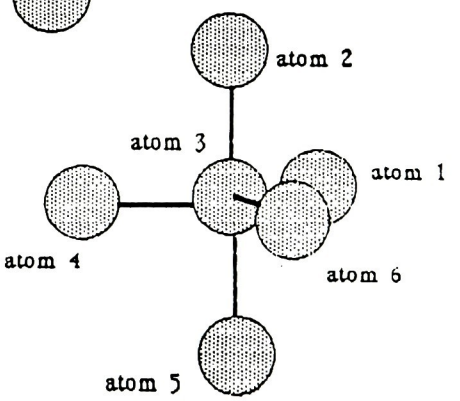
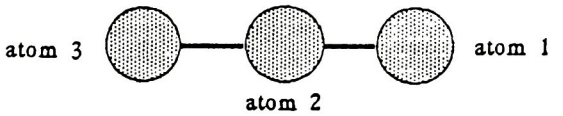
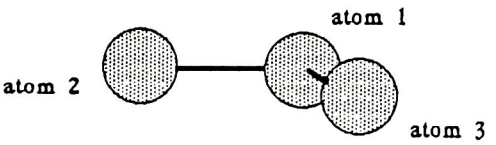


Figure M-1. **STANDARD GEOMETRIC MODELS**

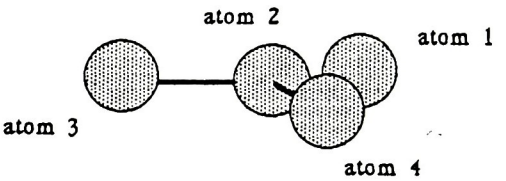
A Standard Linear Model :



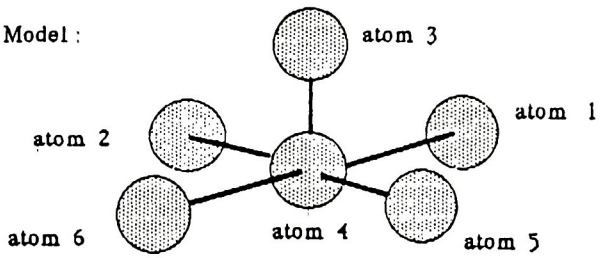
A Standard Triangle Model :



A Standard Trigonal Plane Model :



A Standard
Square Pyramid Model :



A Standard Square Plane Model :

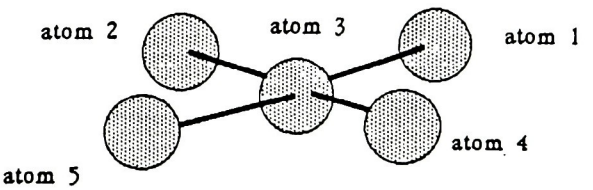


Figure M-2. STANDARD GEOMETRIC MODELS

Table 1:
relation name : LINEAR

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3
BeCl2	Cl	Be	Cl
CO2	O	C	O
HgCl2	Cl	Hg	Cl
HCN	H	C	N
XeF2	F	Xe	F

Table 2:
relation name : TRIANGLE

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3
SO2	S	O	O
S2O	O	S	S
H2O	O	H	H
SCl2	S	Cl	Cl

Table 3:
relation name : TRIGONAL PLANE

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4
BF3	F	B	F	F
BCl3	Cl	B	Cl	Cl
SO3	O	S	O	O
GaI3	Ga	I	Ga	Ga
OCF2	F	C	O	F

Table 4:
relation name : SQUARE PLANE

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4	ATOM.5
XeF4	F	F	Xe	F	F
[PdCl4]--	Cl	Cl	Pd	Cl	Cl
[PtCl4]--	Cl	Cl	Pt	Cl	Cl
[AuCl4]-	Cl	Cl	Au	Cl	Cl

Table 5:
relation name : TRIGONAL PYRAMID

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4
NH3	H	H	N	H
NF3	F	F	N	F
NI3	I	I	N	I
PCl3	Cl	Cl	P	Cl
XeO3	O	O	Xe	O

Table 6:
relation name : TETRAHEDRAL

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4	ATOM.5
CH4	H	H	C	H	H
XeO4	O	O	Xe	O	O
CHCl3	H	Cl	C	Cl	Cl
SiF4	F	F	Si	F	F
BrO4-	O	O	Br	O	O
NH4+	H	H	N	H	H
FeCl4-	Cl	Cl	Fe	Cl	Cl
CH2ClF	H	H	C	Cl	F

Table 7:
relation name : TRIGONAL BI-PYRAMID

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4	ATOM.5	ATOM.6
PCl5	Cl	Cl	P	Cl	Cl	Cl
SbCl5	Cl	Cl	Sb	Cl	Cl	Cl
PF5	F	F	P	F	F	F
NiPBrS3	S	P	Ni	S	Br	S
OSF4	F	F	S	O	F	F

Table 8:
relation name : SQUARE PYRAMID

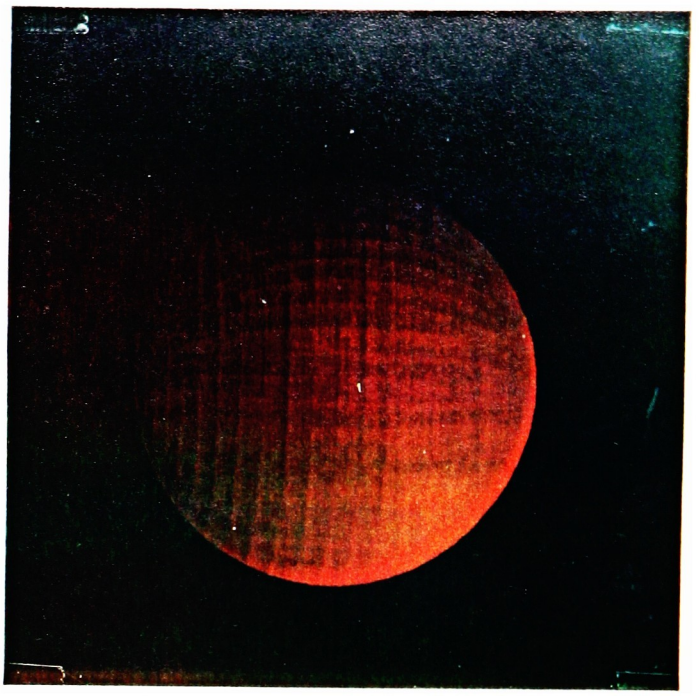
MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4	ATOM.5	ATOM.6
BrF5	F	F	F	Br	F	F
TeF5-	F	F	F	Te	F	F
[SbF5]--	F	F	F	Sb	F	F

Table 9:
relation name : OCTAHEDRAL

MOLE.NAME	ATOM.1	ATOM.2	ATOM.3	ATOM.4	ATOM.5	ATOM.6	ATOM.7
SF6	F	F	F	S	F	F	F
CFe6	Fe	Fe	Fe	C	Fe	Fe	Fe
[SiF6]--	Si	F	F	Si	F	F	F
AsF6-	F	F	F	As	F	F	F

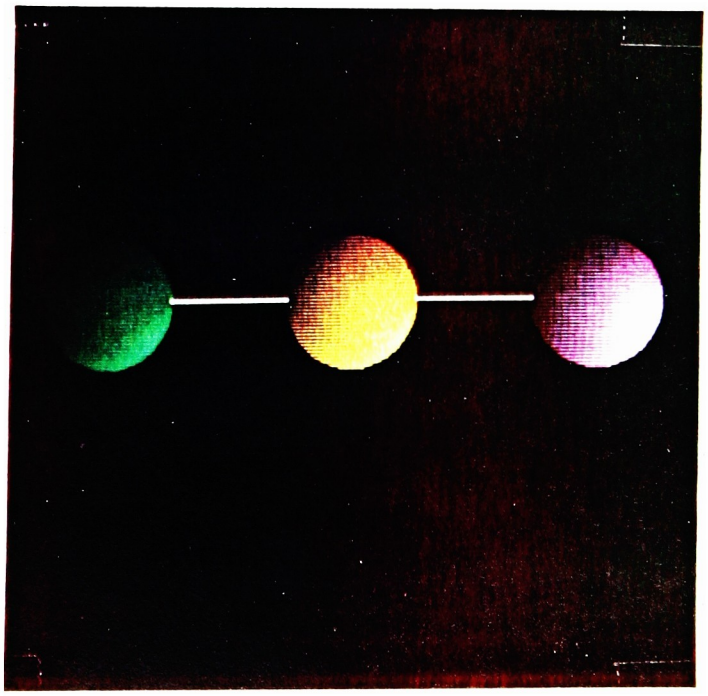
Color Plate A.

A Sphere in
Drawing Status.



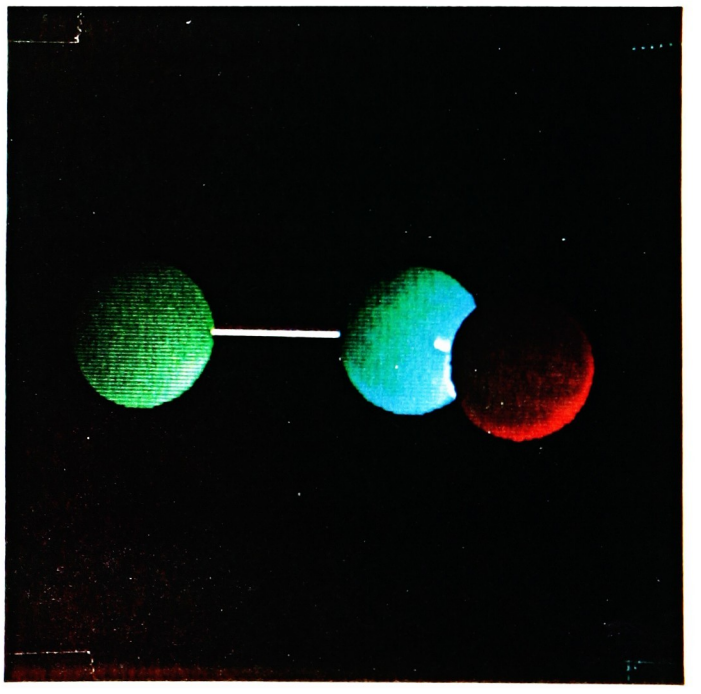
Color Plate B.

A Sphere in Red
with Shade.



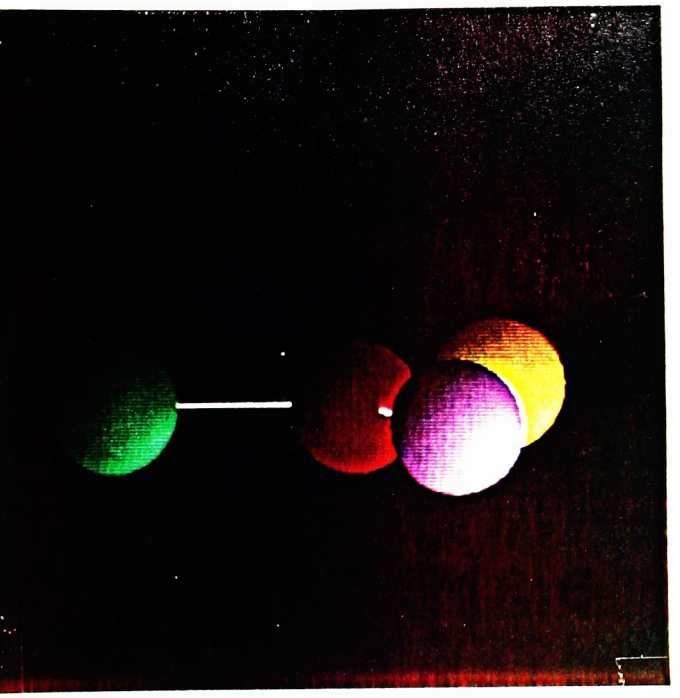
Color Plate 1.

A Standard
Linear Model.



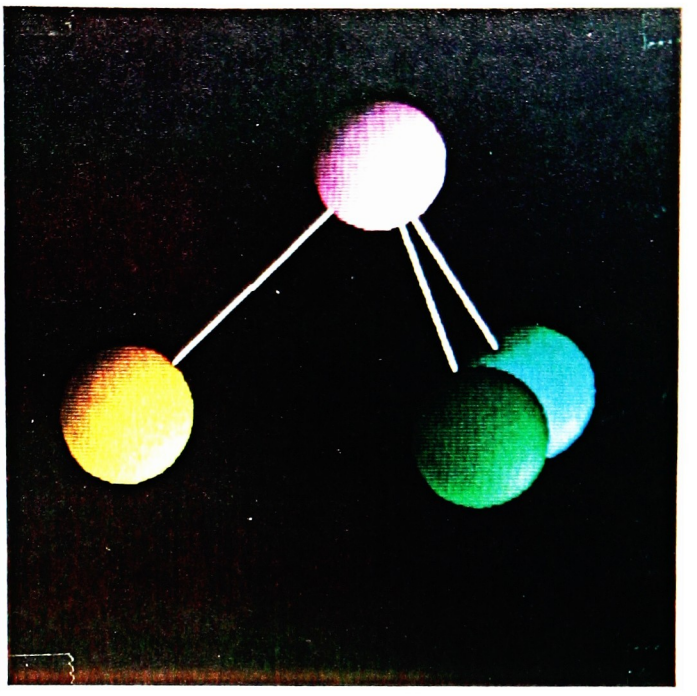
Color Plate 2.

A Standard
Triangle Model.



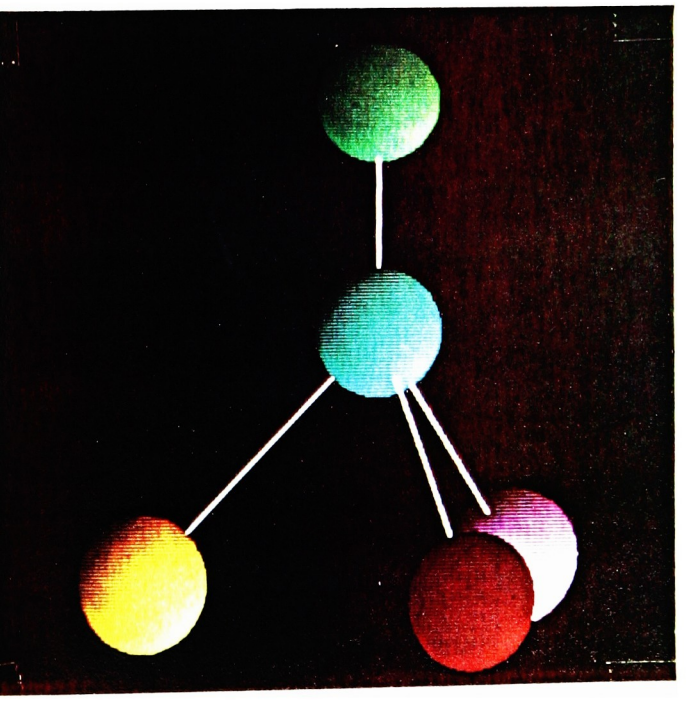
Color Plate 3.

A Standard
Trigonal Plane
Model.



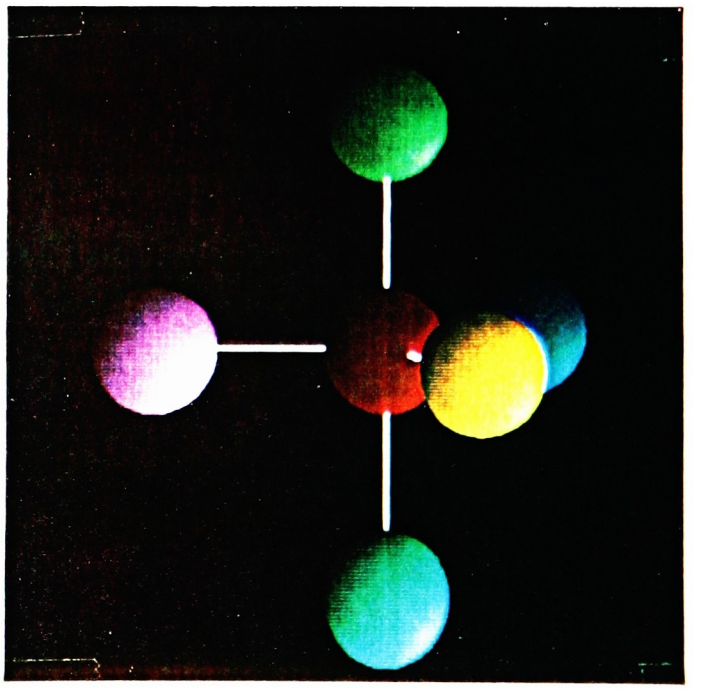
Color Plate 4.

A Standard
Trigonal Pyramid
Model.



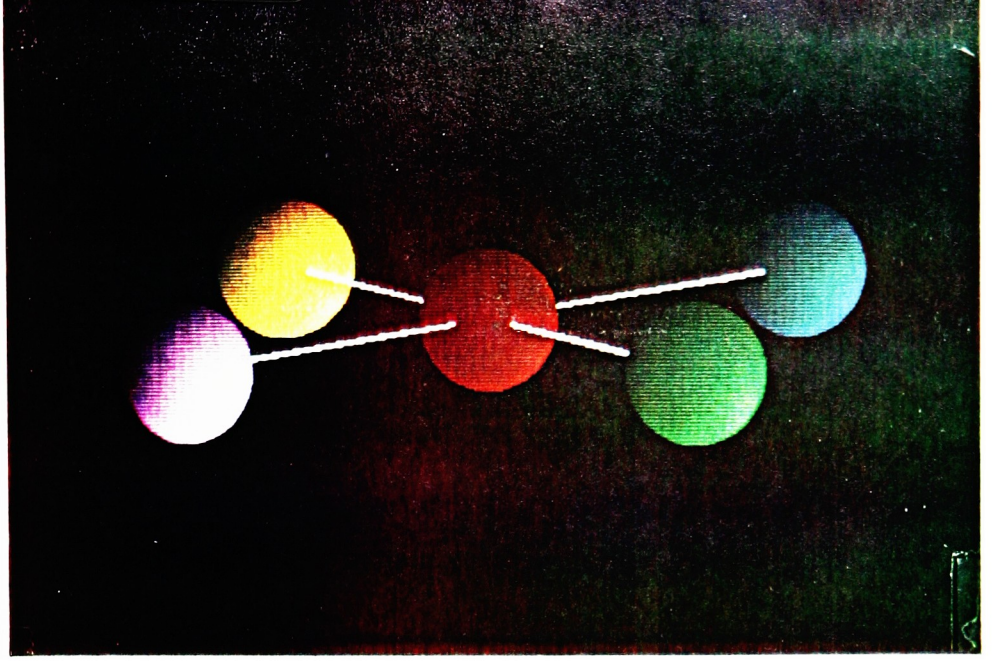
Color Plate 5.

A Standard
Tetrahedral
Model.



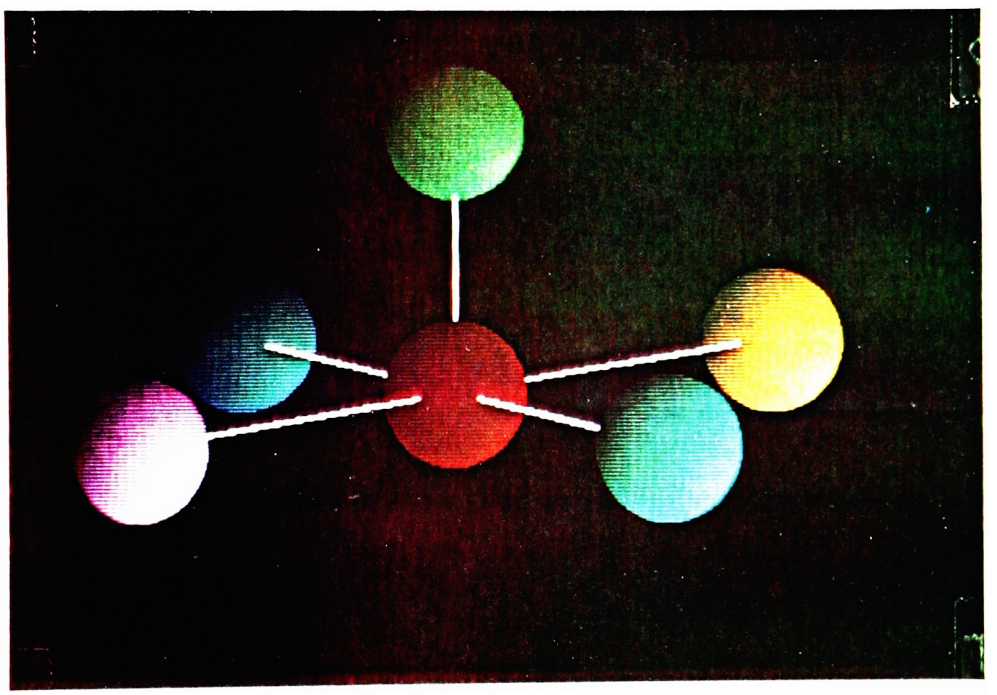
Color Plate 6.

A Standard
Trigonal
Bi-Pyramid
Model.



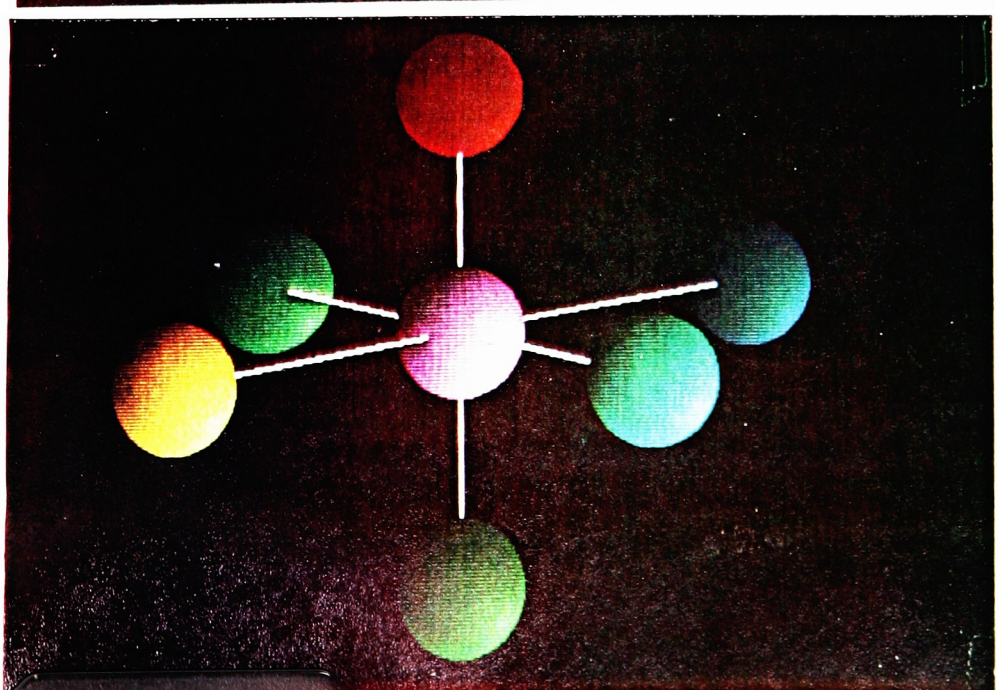
Color Plate 7.

A Standard
Square Plane
Model.



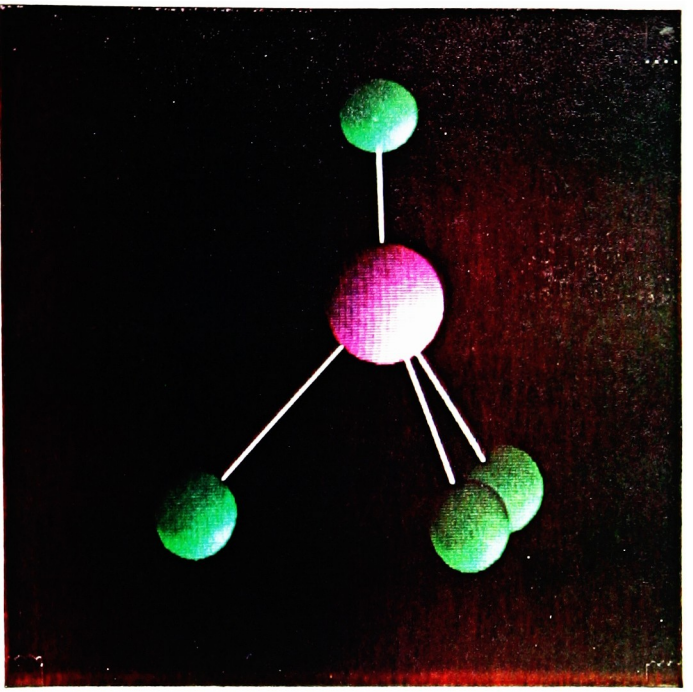
Color Plate 8.

A Standard
Square Pyramid
Model.



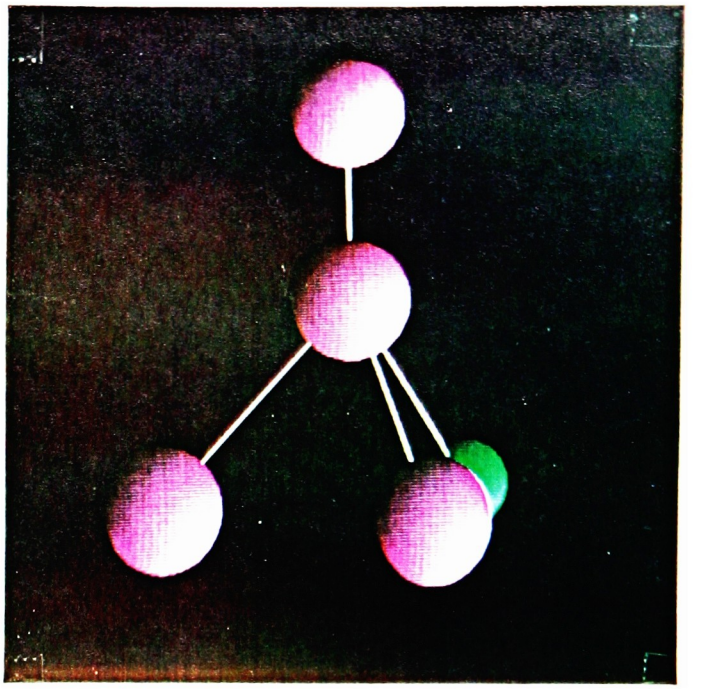
Color Plate 9.

A Standard
Octahedral
Model.



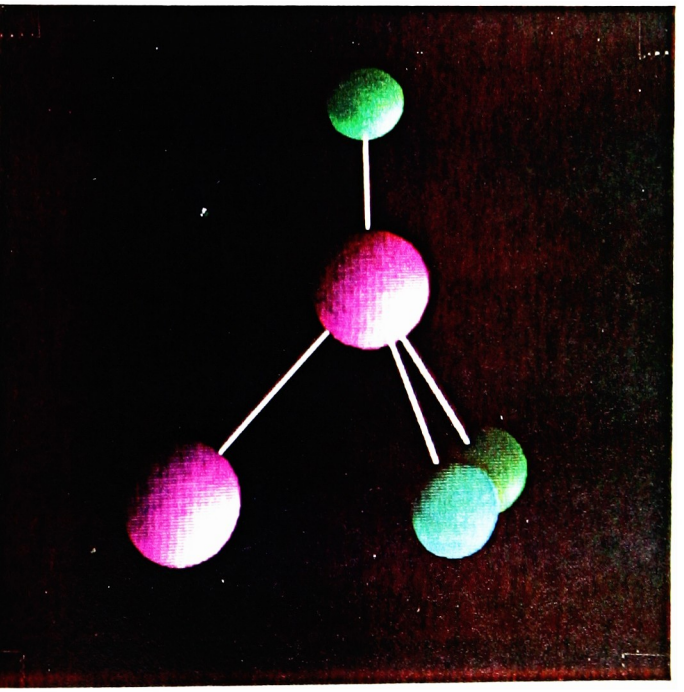
Color Plate 10.

Molecule CH4.



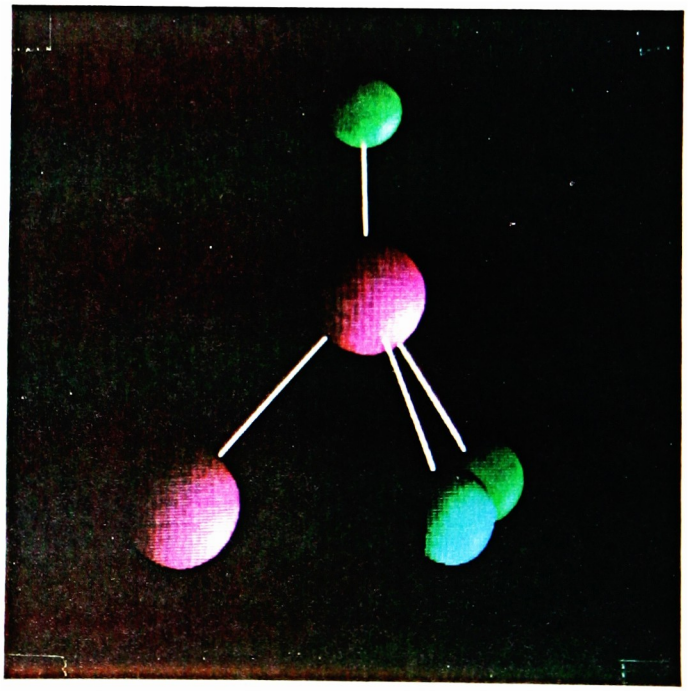
Color Plate 11.

Molecule CHCl3.



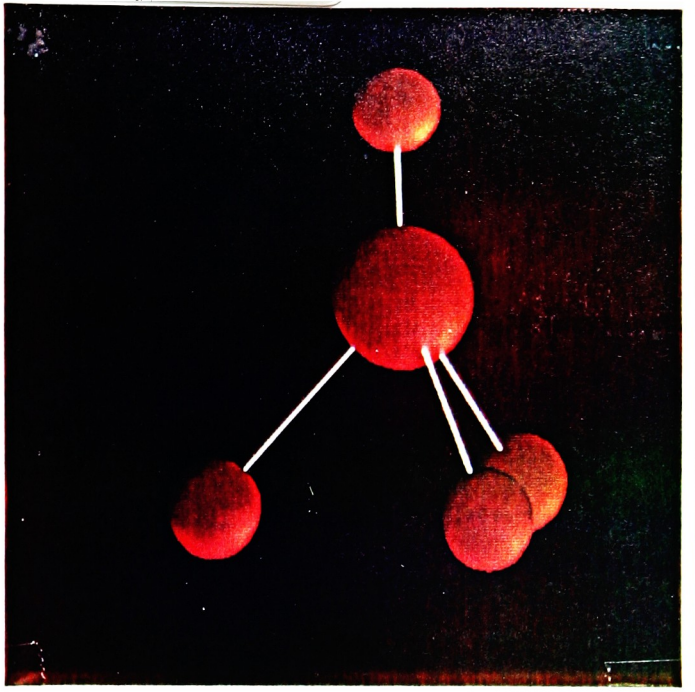
Color Plate 12.

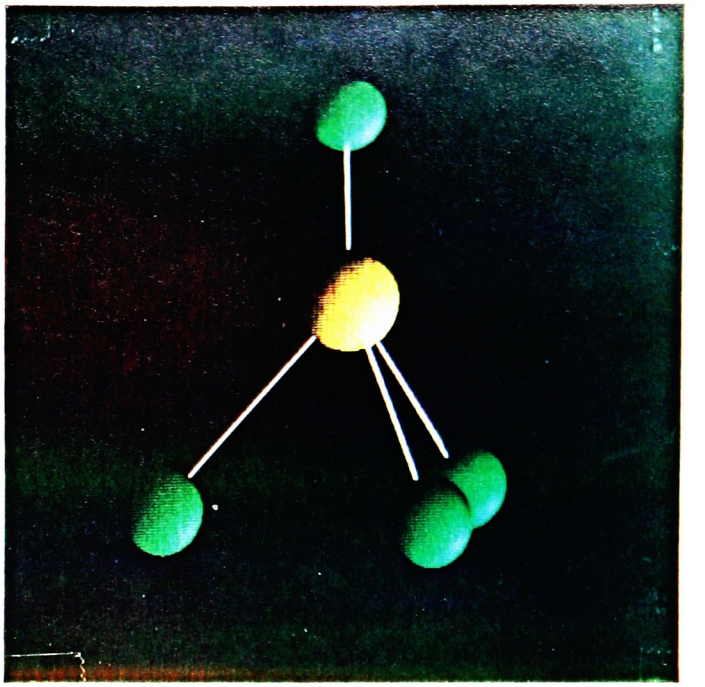
Molecule CH2ClF.



Color Plate 13.

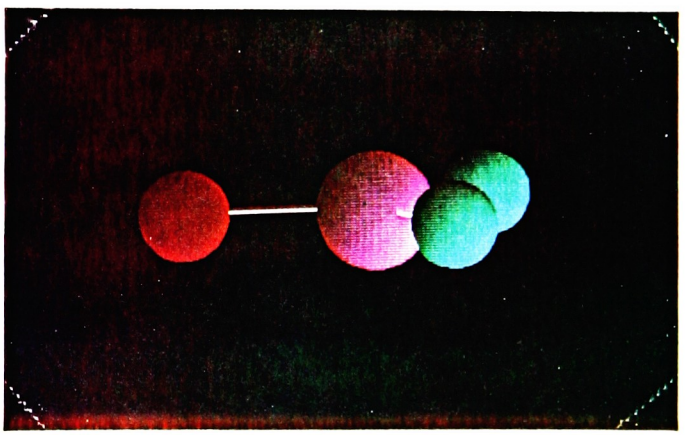
Molecule CH2ClF
in Drawing Status.





Color Plate 14.

Molecule XeO4.



Color Plate 15.

Molecule NH4+.

Color Plate 16.

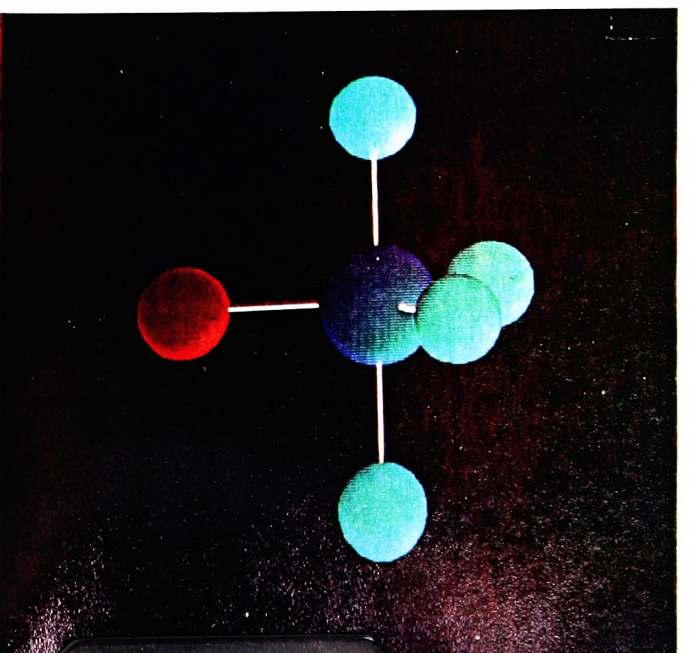
Molecule OCF2.

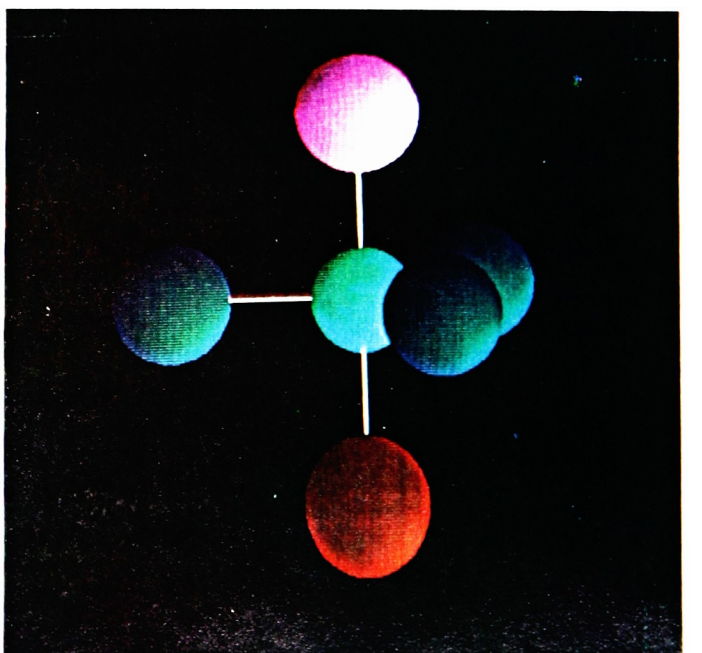
Color Plate 17.

Molecule OSF4.

Color Plate 18.

Molecule NiPBrS3.





5. Conclusions and Future Directions.

In this thesis, the Computer Aided Molecular Design System (the CAMD system) has implemented a relational data base management system to retrieve data, used index trees to prevent duplicate key from existing in one relation, and set up color lookup table in 64 intensity for each color. It also determines each surface polygon in each sphere, finds the surface normal in each facet, and determines color intensity for each polygon in order to draw a sphere with illumination.

A relational database model was chosen for the CAMD system. It employs simple tabular structures that are conceptually simple to understand. In the relational model it is easier to express data integration and extraction than with network structures. A database in the CAMD system is simply a UNIX directory. The name of the directory is the name of the database. This is a reasonable method of organization as a database can be viewed as simply a collection of data relations, with the DBMS imposing organization on the data. Correspondingly, a relation is represented by a UNIX file.

The relation.tbl and attribute.tbl files allow the CAMD system to authenticate the existence of relation and attribute names prior to opening the actual relation. For

this reason, all the CAMD system processes have to access one or both these files in some manner. Each relation has its corresponding data file and index file. Between the relations (data files and index files), the relation.tbl, and attribute.tbl file, there is sufficient information for the CAMD system to access the relations in an organized manner and answer user's requests.

In the CAMD system, the index tree performs three advantageous functions: (1) storing primary keys, (2) avoiding duplicate key, (3) finding the location of primary keys in data file. In the third function, it also makes it easier and faster to access a key in the data file than sequentially searching from the initial position of the data file.

The CAMD system does not offer multiple-record delete or all-record delete, because the system maintains twelve relations for each process to use related data in data base. For example, if the user of the CAMD system deleted all tuples in ATOM TABLE relation, the CAMD system can not find radius and color for a particular atom. Also, if the system allows the user to delete multiple-tuple, said "COLOR = GREEN", then the system can not find other atoms whose color is green. Currently, the system manipulates DELETE operation under the first attribute only to compare which tuple is to be deleted. In other words, a data

tuple was deleted for which the primary key is only equal to a specified data. But the system program originally was designed to do multiple-tuple delete, all-tuple delete under any specified attribute. If one tuple was deleted, the system will change data-in-use flag to specify that this tuple does not exist. And this tuple can not be used again. But from the disk memory point, this deleted tuple still occupied a tuple length of space in data file.

UPDATE operations can be seen in a general query language. The CAMD system does not support this operation. The UPDATE operation is used to modify attribute values in one or more existing records in a table. The user can do a deletion first, then followed by an insertion to incorrect tuple. Whenever the data needs to be modified in a tuple of table, the user has to do the DELETE operation to remove this specified tuple. Then the user adds the modified tuple at the end of data file. If these process happens many times, then the data file will become bigger and bigger. Thus the system wastes lots of storage. So, the incorporation of an "UPDATE" operation would be a future extension for the CAMD system.

The CAMD system only allows creation of nine standard geometric models. Because the program designer has already fixed nine different relations to store these nine model. When the user enters this system the first time and

requests the system to define these relations once in data base system. After that, if the user requests to define these relations, the system will show some error message, since these relations have already been defined. on the other hand, if the user really wants to create a new relation for new geometric model, then the system program has to be modified. Because the GEOMETRY relation only accommodates seven sets of atom's data. Each set of atom's data needs 5 attributes to store (x, y, z) coordinate value, radius and color. The maximum atom number in the CAMD system is seven. In each data tuple, its primary key is used to hold the geometric model type, such as TRIANGLE. Each data tuple needs 36 attributes. Therefore, to create a new geometric model which is more than seven spheres, it must extend the attribute fields. The CAMD system does not currently provide this function. Provision of this function is another future development. One can take and the CAMD system would then be more powerful in designing more complex molecules.

When inserting data into the GEOMETRY relation, the total number of sphere could be less than seven spheres in some geometric models. The triangle model only requires three spheres. It only needs sixteen attribute spaces to store its data. But there are thirty-six attributes in each tuple. So the user has to input (-99, -99, -99) for next (x, y, z) coordinate value as a terminal flag. Then

the CAMD system will discard the rest of attribute in the data tuple.

In graphics system, to draw a standard geometric model or a molecule, the molecular bond will be drawn first between its related spheres from center of sphere to the center of another sphere, then according to hidden surface removal effective to determine drawing which sphere or which stick first from rear to front in z-axis direction. The graphic system also can find exactly connective point between sphere (atom) and stick (bond) to make molecule more stereography.

When the system draws a molecule on graphic screen, the ideal of parallel projection was selected. But why not the ideal of perspective projection? Because there is a ambiguity for molecular structure. Usually, the interpretation of perspective projection is often based on the assumption that a smaller object is further away. If there are two houses on the screen. One appears bigger than the other. It would probably be assumed that the larger house is nearer to the viewer. However the house which appears larger (a mansion, perhaps) may actually be more distant than the one which appears small (a cottage, perhaps), at least so long as there are no other cues, such as trees and windows. When the viewer knows that the projected objects have many parallel lines, perspective

does convey depth, because the parallel lines seems to converge at their vanishing points. This convergence may actually be a stronger depth cue than the decreasing size effect.

The perspective projections stick-figure molecule representations were less convincing than parallel projections, explaining this by the lack of converging parallel lines in molecular structures (spherical structures). Also atom A (Aluminum, perhaps) may be bigger than atom B (Hydrogen, perhaps). The Aluminum may also have deeper Z coordinate value. If the system did a perspective projection to draw two spheres, The atom A would be smaller than atom B. On the screen The viewer will assume that Hydrogen is larger than Aluminum. So the system preferred to select parallel projection in this project. But how can it provide depth cues to the viewer, The system used hidden-surface removal ideal to draw picture. In other words, it always draw deep value in Z coordinate first from rear object to front object. In this way it does convey depth.

This thesis combined computer graphics and a relational data base management system to implement a Computer Aided Molecular Design system. Its importance can be realized in a variety of applications. This package may be used in chemical education to understand chemistry concepts in molecular structure.

6. Bibliography.

(1) Bauer, Johannes and Schubert, Wolfgang. "Computer Modeling of Molecular Structures", Computers & Chemistry. Vol. 7, No. 2. 1983.

(2) David Kroenke, "DATABASE processing: Fundamentals, Design, Implementation", 2nd Edition, 1983.

(3) Fred R. McFadden and Jeffrey A. Hoffer, "DATA BASE MANAGEMENT", The Benjamin/Cummings Publishing Company, 1985.

(4) Alfonso F. Cardenas, "Data Base Management System", Allyn and Bacon, Inc., 1979.

(5) E. F. Codd, "A Relational Model of Data for Large Shared Database Books", CACM 13, No. 6, June 1970.

(6) Vectrix Corp., "Vectrix VX128-A and VX384-A : User's Guide and Reference Manual", 1984.

(7) William M. Newman and Robert F. Sproull "Principles of Interactive Computer Graphics", 2nd Edition, McGraw Hill, Inc. 1979.

(8) David F. Rogers, "Procedural Elements for Computer Graphics", McGraw Hill, Inc. 1985.

(9) Steven Harington, "Computer Graphics A Programming Approach", McGraw Hill, Inc. 1983.

(10) Wolfgang K. Gilol, "Interactive Computer Graphics, Data Structures, Algorithms, Languages", Prentice-Hall, Inc, 1978.

(11) Andrew S. Glassner, "Computer Graphics User's Guide" Howard W. Sams & Co., Inc. 1984.

(12) Staudhammer and Khurana, "Display of Molecular Models with Interactive Graphics", IEEE Computer Graphics and Applications, 1986.

(13) N. L. Max, "ATOMS LLL, Atoms with Shading and Highlights", Computer Graphics, 1979.

(14) J. Staudhammer, "On Display of Space-Filling Atomic Models in Real Time", Computer Graphics, 1978.

(15) T. K. Porter, "Spherical Shading", Computer Graphics, 1978.

(16) C. J. Date, "A Guide to DB 2", Addison-Wesley, 1984.

(17) C. J. Date, "An Introduction to Database Systems", volume I, Forth Edition, Addison-Wesley, 1986.

(18) C. J. Date, "An Introduction to Database Systems", volume II, Addison-Wesley, 1983.

(19) Kernighan and Ritchie, "The C Programming Language", Prentice-Hall, Inc., 1978.

(20) S. R. Bourne, "The UNIX System". Bell Telephone Laboratories, Incorporated, 1983.

(21) James E. Haheey, "Inorganic Chemistry Principles of Structure and Reactivity", 1983.

(22) Andrew S. Glassner, "Computer Graphics User's Guide", Howard W. Sam & Co., Inc. 1984.

(23) Van Norstrand Reinhold, Company Inc. "Encyclopedia of Chemistry", 4th Edition, 1984.

(24) Robert C. Smoot and Jack Price, "Chemistry: A Modern Course", Charles E. Merrill Publishing Co., 1982.

(25) Ken Knowlton and Lorinda Cherry, "ATOMS-A THREE-D OPAQUE MOLECULE SYSTEM-FOR COLOR PICTURES OF SPACE-FILLING OR BALL-AND-STICK MODELS", Computers and Chemistry, Vol 1. 1977.

(26) David N J White and John E Pearson, "Comprehensive molecular modelling system", Journal of Molecular Graphics, Vol 4, No. 3, 1986.

(27) Johannes Bauer and Wolfgang Schubert, "COMPUTER MODELING OF MOLECULAR STRUCTURES", Computers and Chemistry, Vol 7, No. 2, 1983.

7. Appendix.

7.1. Some Output Examples.

```
Script started on Mon Dec 15 13:44:29 1986
(1)% run
```

```
================================================================================

                        ----WELCOME TO CAMD SYSTEM----

                        (Computer Aided Molecular Design System)

================================================================================

        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==)  2

--------------------------------------------------------------------------------
1. To define table and attribute name.
2. To add data into table.
3. To exit CAMD system

--------------------------------------------------------------------------------

Input data --)  1

To create ---  MOLECULE  --- file.
To create ---  GEOMETRY  --- file.
To create ---  LINEAR  --- file.
To create ---  TRIANGLE  --- file.
To create ---  TRIGONAL PLANE  --- file.
To create ---  SQUARE PLANE  --- file.
To create ---  TRIGONAL PYRAMID  --- file.
To create ---  TETRAHEDRAL  --- file.
To create ---  TRIGONAL BI-PYRAMID  --- file.
To create ---  SQUARE PYRAMID  --- file.
To create ---  OCTAHEDRAL  --- file.
To create ---  ATOM TABLE  --- file.

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)  n
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==)  2

--------------------------------------------------------------------------------
1. To define table and attribute name.
2. To add data into table.
3. To exit CAMD system

--------------------------------------------------------------------------------

Input data --)  2
Create data to --- 1. MOLECULE.
                   2. ATOM TABLE or GEOMETRY.
                   3. to exit CAMD system.
Input number --)  2

The current relation in CAMD system is as follows :
                                  1.  ==)   MOLECULE
                                  2.  ==)   GEOMETRY
                                  3.  ==)   LINEAR
```

```
                                4.   ==)    TRIANGLE
                                5.   ==)    TRIGONAL PLANE
                                6.   ==)    SQUARE PLANE
                                7.   ==)    TRIGONAL PYRAMID
                                8.   ==)    TETRAHEDRAL
                                9.   ==)    TRIGONAL BI-PYRAMID
                                10.   ==)    SQUARE PYRAMID
                                11.   ==)    OCTAHEDRAL
                                12.   ==)    ATOM TABLE

Enter relation name ==)   ATOM TABLE

--ATOM.NAME-- (char) ==)  H

--RADIUS-- (real) ==)  1.25

--COLOR-- (char) ==)  GREEN

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  Be

--RADIUS-- (real) ==)  2.20

--COLOR-- (char) ==)  GREEN

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  B

--RADIUS-- (real) ==)  2.17

--COLOR-- (char) ==)  GREEN

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  C

--RADIUS-- (real) ==)  1.85

--COLOR-- (char) ==)  PINK

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  N

--RADIUS-- (real) ==)  1.54

--COLOR-- (char) ==)  YELLOW

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  O

--RADIUS-- (real) ==)  1.40

--COLOR-- (char) ==)  RED

Add more tuple in --ATOM TABLE-- table ? (y or n) ==)  Y

--ATOM.NAME-- (char) ==)  F

--RADIUS-- (real) ==)  1.35

--COLOR-- (char) ==)  CYAN
```

```
Add more tuple in --ATOM TABLE-- table ? (y or n) ==>   N

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>   N
        -------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,    3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==>  2

--------------------------------------------------------------------------------
1. To define table and attribute name.
2. To add data into table.
3. To exit CAMD system

--------------------------------------------------------------------------------

Input data -->  2
Create data to --- 1. MOLECULE.
                   2. ATOM TABLE or GEOMETRY.
                   3. to exit CAMD system.
Input number -->  1

--MOLE.NAME-- (char) ==>  BeCl2

Input  --MODEL.TYPE--
1. LINEAR,    2. TRIANGLE,   3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,   6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,   8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number -->  1

Input ATOM.1 (char)  ==>  Cl

Input ATOM.2 (char)  ==>  Be

Input ATOM.3 (char)  ==>  Cl
add new molecule, type 'yes' or 'YES'  --->  yes

--MOLE.NAME-- (char) ==>  CO2

Input  --MODEL.TYPE--
1. LINEAR,    2. TRIANGLE,   3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,   6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,   8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number -->  1

Input ATOM.1 (char)  ==>  O

Input ATOM.2 (char)  ==>  C

Input ATOM.3 (char)  ==>  O
add new molecule, type 'yes' or 'YES'  --->  no

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>   n
        -------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,    3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------
```

```
Please enter one number to process  ==>  3

Enter relation name to process :   MOLECULE

-------------------------------------------------------------------------------
BeCl2                    CO2                      HgCl2                    HCN
                XeF2
next molecules (YES or NO) ==>  YES


-------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==>  1

-------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

-------------------------------------------------------------------------------

Input data  ==>  3

Enter relation name to process :   ATOM TABLE

-------------------------------------------------------------------------------
ATOM.NAME        RADIUS           COLOR
-------------------------------------------------------------------------------
H             1.250 GREEN
Be            2.200 GREEN
B             2.170 GREEN
C             1.850 PINK
N             1.540 YELLOW
O             1.400 RED
F             1.350 CYAN
Al            2.530 WHITE
Si            2.240 GREEN
P             1.960 PINK
S             1.850 BLUE
Cl            1.800 PINK
Co            1.590 CYAN
Ni            1.600 CYAN
Cu            1.410 YELLOW
Ga            2.010 YELLOW
As            2.020 RED
Se            2.010 BLUE
Br            1.950 RED
Pd            1.630 GREEN
Sb            2.210 YELLOW
Te            2.220 PINK
I             2.150 BLUE
Xe            2.190 RED
Pt            1.830 YELLOW
Au            1.690 RED
Hg            1.470 CYAN
```

```
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N

        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.

Please enter one number to process  ==>  1

1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

Input data  ==>  3

Enter relation name to process :   MOLECULE

MOLE.NAME            MODEL.TYPE
BeCl2                  LINEAR
CO2                    LINEAR
HgCl2                  LINEAR
HCN                    LINEAR
XeF2                   LINEAR

To exit CAMD system, please type 'quit' or 'QUIT' ==>  QUIT
(2)% exit
(3)%
script done on Mon Dec 15 14:03:13 1986
Script started on Mon Dec 15 14:28:00 1986
(1)%
(1)% run

                        ----WELCOME TO CAMD SYSTEM----

                      (Computer Aided Molecular Design System)

        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.

Please enter one number to process  ==>  2

1. To define table and attribute name.
2. To add data into table.
3. To exit CAMD system
```

```
Input data --)  2
Create data to --- 1. MOLECULE.
                   2. ATOM TABLE or GEOMETRY.
                   3. to exit CAMD system.
Input number --)  1

--MOLE.NAME-- (char) ==)  SO2

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  2

Input ATOM.1 (char)  ==)  S

Input ATOM.2 (char)  ==)  O

Input ATOM.3 (char)  ==)  O
add new molecule, type 'yes' or 'YES'  --->  YES

--MOLE.NAME-- (char) ==)  NF3

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  5

Input ATOM.1 (char)  ==)  F

Input ATOM.2 (char)  ==)  F

Input ATOM.3 (char)  ==)  N

Input ATOM.4 (char)  ==)  F
add new molecule, type 'yes' or 'YES'  --->  YES

--MOLE.NAME-- (char) ==)  BF3

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  3

Input ATOM.1 (char)  ==)  F

Input ATOM.2 (char)  ==)  B

Input ATOM.3 (char)  ==)  F

Input ATOM.4 (char)  ==)  F
add new molecule, type 'yes' or 'YES'  --->  yes

--MOLE.NAME-- (char) ==)  XeF4

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.
```

```
Select one model number --)  4

Input ATOM.1 (char)  ==)  F

Input ATOM.2 (char)  ==)  F

Input ATOM.3 (char)  ==)  Xe

Input ATOM.4 (char)  ==)  F

Input ATOM.5 (char)  ==)  F
add new molecule, type 'yes' or 'YES'  ---)  yes

--MOLE.NAME-- (char) ==)  CH4

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  6

Input ATOM.1 (char)  ==)  H

Input ATOM.2 (char)  ==)  H

Input ATOM.3 (char)  ==)  C

Input ATOM.4 (char)  ==)  H

Input ATOM.5 (char)  ==)  H
add new molecule, type 'yes' or 'YES'  ---)  YES

--MOLE.NAME-- (char) ==)  PCl5

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  7

Input ATOM.1 (char)  ==)  Cl

Input ATOM.2 (char)  ==)  Cl

Input ATOM.3 (char)  ==)  P

Input ATOM.4 (char)  ==)  Cl

Input ATOM.5 (char)  ==)  Cl

Input ATOM.6 (char)  ==)  Cl
add new molecule, type 'yes' or 'YES'  ---)  yes

--MOLE.NAME-- (char) ==)  BrF5

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number --)  8

Input ATOM.1 (char)  ==)  F
```

```
Input ATOM.2 (char)  ==>  F

Input ATOM.3 (char)  ==>  F

Input ATOM.4 (char)  ==>  Br

Input ATOM.5 (char)  ==>  F

Input ATOM.6 (char)  ==>  F
add new molecule, type 'yes' or 'YES'  --->  YES

--MOLE.NAME-- (char) ==>  SF6

Input  --MODEL.TYPE--
1. LINEAR,     2. TRIANGLE,    3. TRIGONAL PLANE.
4. SQUARE PLANE,  5. TRIGONAL PYRAMID,    6. TETRAHEDRAL
7. TRIGONAL BI-PYRAMID,    8. SQUARE PYRAMID,  9. OCTAHEDRAL.

Select one model number -->  9

Input ATOM.1 (char)  ==>  F

Input ATOM.2 (char)  ==>  F

Input ATOM.3 (char)  ==>  F

Input ATOM.4 (char)  ==>  S

Input ATOM.5 (char)  ==>  F

Input ATOM.6 (char)  ==>  F

Input ATOM.7 (char)  ==>  F
add new molecule, type 'yes' or 'YES'  --->  no

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  n
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,    3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==>  3

Enter relation name to process :   MOLECULE

-------------------------------------------------------------------------------
BeCl2                   CO2                     HgCl2                   HCN
                XeF2
next molecules (YES or NO) ==>  YES

SO2                     S2O                     H2O                     SCl2
                NH3
next molecules (YES or NO) ==>  YES

NF3                     NI3                     PCl3                    XeO3
                BF3
next molecules (YES or NO) ==>  YES

BCl3                    SO3                     GaI3                    OCF2
                XeF4
next molecules (YES or NO) ==>  YES

[PdCl4]--               [PtCl4]--               [AuCl4]-                CH4
```

```
                XeO4
next molecules (YES or NO) ==)  YES

CHCl3                   SiF4                    BrO4-                   NH4+
                FeCl4-
next molecules (YES or NO) ==)  YES

CH2ClF                  PCl5                    SbCl5                   PF5
                NiPBrS3
next molecules (YES or NO) ==)  YES

OSF4                    BrF5                    TeF5-                   [SbF5]--
                SF6
next molecules (YES or NO) ==)  YES

CFe6                    [SiF6]--                AsF6-
-------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)  N
        ---------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==)  1

-------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

-------------------------------------------------------------------------------

Input data  ==)  3

Enter relation name to process :   MOLECULE

-------------------------------------------------------------------------------
MOLE.NAME            MODEL.TYPE
-------------------------------------------------------------------------------
BeCl2                  LINEAR
CO2                    LINEAR
HgCl2                  LINEAR
HCN                    LINEAR
XeF2                   LINEAR
SO2                    TRIANGLE
S2O                    TRIANGLE
H2O                    TRIANGLE
SCl2                   TRIANGLE
NH3                    TRIGONAL PYRAMID
NF3                    TRIGONAL PYRAMID
NI3                    TRIGONAL PYRAMID
PCl3                   TRIGONAL PYRAMID
XeO3                   TRIGONAL PYRAMID
BF3                    TRIGONAL PLANE
BCl3                   TRIGONAL PLANE
SO3                    TRIGONAL PLANE
GaI3                   TRIGONAL PLANE
OCF2                   TRIGONAL PLANE
XeF4                   SQUARE PLANE
[PdCl4]--              SQUARE PLANE
[PtCl4]--              SQUARE PLANE
```

[AuCl4]-	SQUARE PLANE
CH4	TETRAHEDRAL
XeO4	TETRAHEDRAL
CHCl3	TETRAHEDRAL
SiF4	TETRAHEDRAL
BrO4-	TETRAHEDRAL
NH4+	TETRAHEDRAL
FeCl4-	TETRAHEDRAL
CH2ClF	TETRAHEDRAL
PCl5	TRIGONAL BI-PYRAMID
SbCl5	TRIGONAL BI-PYRAMID
PF5	TRIGONAL BI-PYRAMID
NiPBrS3	TRIGONAL BI-PYRAMID
OSF4	TRIGONAL BI-PYRAMID
BrF5	SQUARE PYRAMID
TeF5-	SQUARE PYRAMID
[SbF5]--	SQUARE PYRAMID
SF6	OCTAHEDRAL
CFe6	OCTAHEDRAL
[SiF6]--	OCTAHEDRAL
AsF6-	OCTAHEDRAL

To exit CAMD system, please type 'quit' or 'QUIT' ==> N

1. HELP, 2. Create Molecular Table, 3. List Molecular Name
4. Display a Molecular Model, 5. Delete a Molecule, 6. QUIT.

Please enter one number to process ==> 1

---

1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

Input data ==> 2

The current relation in CAMD system is as follows :

1. ==> MOLECULE
2. ==> GEOMETRY
3. ==> LINEAR
4. ==> TRIANGLE
5. ==> TRIGONAL PLANE
6. ==> SQUARE PLANE
7. ==> TRIGONAL PYRAMID
8. ==> TETRAHEDRAL
9. ==> TRIGONAL BI-PYRAMID
10. ==> SQUARE PYRAMID
11. ==> OCTAHEDRAL
12. ==> ATOM TABLE

Enter relation name to DESCRIBE ==> ATOM TABLE

DESCRIBE ATOM TABLE

```
RELATION NAME           ATTRIBUTE NAME          TYPE          LENGTH    KEY
-------------------------------------------------------------------------
                                                                        //,
ATOM TABLE              ATOM.NAME               CHARACTER     9         YES
ATOM TABLE              RADIUS                  DECIMAL       0         NO
ATOM TABLE              COLOR                   CHARACTER     10        NO

No more attribute field in attribute table !
-------------------------------------------------------------------------

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ---------------------------------------------------------
        1. HELP,     2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------

Please enter one number to process  ==>  2

-------------------------------------------------------------------------
1. To define table and attribute name.
2. To add data into table.
3. To exit CAMD system

-------------------------------------------------------------------------

Input data -->  2
Create data to --- 1. MOLECULE.
                   2. ATOM TABLE or GEOMETRY.
                   3. to exit CAMD system.
Input number -->  2

The current relation in CAMD system is as follows :
                                    1.  ==>   MOLECULE
                                    2.  ==>   GEOMETRY
                                    3.  ==>   LINEAR
                                    4.  ==>   TRIANGLE
                                    5.  ==>   TRIGONAL PLANE
                                    6.  ==>   SQUARE PLANE
                                    7.  ==>   TRIGONAL PYRAMID
                                    8.  ==>   TETRAHEDRAL
                                    9.  ==>   TRIGONAL BI-PYRAMID
                                    10.  ==>   SQUARE PYRAMID
                                    11.  ==>   OCTAHEDRAL
                                    12.  ==>   ATOM TABLE


Enter relation name ==>   GEOMETRY

--GEOM.NAME-- (char) ==>  OCTAHEDRAL

--X.1-- (real) ==>  0.3

--Y.1-- (real) ==>  0

--Z.1-- (real) ==>  -0.6

--RADIUS.1-- (real) ==>  1.6

--COLOR.1-- (char) ==>  BLUE

--X.2-- (real) ==>  -0.3

--Y.2-- (real) ==>  0
```

```
--Z.2-- (real) ==>  -0.6
--RADIUS.2-- (real) ==>  1.6
--COLOR.2-- (char) ==>  GREEN
--X.3-- (real) ==>  0
--Y.3-- (real) ==>  0.3
--Z.3-- (real) ==>  0
--RADIUS.3-- (real) ==>  1.6
--COLOR.3-- (char) ==>  RED
--X.4-- (real) ==>  0
--Y.4-- (real) ==>  0
--Z.4-- (real) ==>  0
--RADIUS.4-- (real) ==>  1.6
--COLOR.4-- (char) ==>  PINK
--X.5-- (real) ==>  0
--Y.5-- (real) ==>  -0.3
--Z.5-- (real) ==>  0.0
--RADIUS.5-- (real) ==>  1.6
--COLOR.5-- (char) ==>  GREEN
--X.6-- (real) ==>  0.3
--Y.6-- (real) ==>  0
--Z.6-- (real) ==>  0.6
--RADIUS.6-- (real) ==>  1.6
--COLOR.6-- (char) ==>  CYAN
--X.7-- (real) ==>  -0.3
--Y.7-- (real) ==>  0
--Z.7-- (real) ==>  0.6
--RADIUS.7-- (real) ==>  1.6
--COLOR.7-- (char) ==>  YELLOW
Add more tuple in --GEOMETRY-- table ? (y or n) ==>  Y
--GEOM.NAME-- (char) ==>  SQUARE PYRAMID
--X.1-- (real) ==>  0.3
--Y.1-- (real) ==>  0
--Z.1-- (real) ==>  -0.6
```

```
--RADIUS.1-- (real) ==>  1.6
--COLOR.1-- (char) ==>  YELLOW
--X.2-- (real) ==>  -0.3
--Y.2-- (real) ==>  0
--Z.2-- (real) ==>  -0.6
--RADIUS.2-- (real) ==>  1.6
--COLOR.2-- (char) ==>  BLUE
--X.3-- (real) ==>  0
--Y.3-- (real) ==>  0.3
--Z.3-- (real) ==>  0
--RADIUS.3-- (real) ==>  1.6
--COLOR.3-- (char) ==>  GREEN
--X.4-- (real) ==>  0
--Y.4-- (real) ==>  0.0
--Z.4-- (real) ==>  0.0
--RADIUS.4-- (real) ==>  1.6
--COLOR.4-- (char) ==>  RED
--X.5-- (real) ==>  0.3
--Y.5-- (real) ==>  0
--Z.5-- (real) ==>  0.6
--RADIUS.5-- (real) ==>  1.6
--COLOR.5-- (char) ==>  CYAN
--X.6-- (real) ==>  -0.3
--Y.6-- (real) ==>  0.0
--Z.6-- (real) ==>  0.6
--RADIUS.6-- (real) ==>  1.6
--COLOR.6-- (char) ==>  PINK
--X.7-- (real) ==>  -99
--Y.7-- (real) ==>  -99
--Z.7-- (real) ==>  -99
--RADIUS.7-- (real) ==>  0
--COLOR.7-- (char) ==>  *
Add more tuple in --GEOMETRY-- table ? (y or n) ==>  Y
```

```
--GEOM.NAME-- (char) ==>  TRIGONAL BI-PYRAMID
--X.1-- (real) ==>  0.15
--Y.1-- (real) ==>  0.0
--Z.1-- (real) ==>  -0.259
--RADIUS.1-- (real) ==>  1.6
--COLOR.1-- (char) ==>  BLUE
--X.2-- (real) ==>  0.0
--Y.2-- (real) ==>  0.3
--Z.2-- (real) ==>  0.0
--RADIUS.2-- (real) ==>  1.6
--COLOR.2-- (char) ==>  GREEN
--X.3-- (real) ==>  0.0
--Y.3-- (real) ==>  0.0
--Z.3-- (real) ==>  0.0
--RADIUS.3-- (real) ==>  1.6
--COLOR.3-- (char) ==>  RED
--X.4-- (real) ==>  -0.3
--Y.4-- (real) ==>  0.0
--Z.4-- (real) ==>  0.0
--RADIUS.4-- (real) ==>  1.6
--COLOR.4-- (char) ==>  PINK
--X.5-- (real) ==>  0.0
--Y.5-- (real) ==>  -0.3
--Z.5-- (real) ==>  0.0
--RADIUS.5-- (real) ==>  CYAN
--COLOR.5-- (char) ==>  0.15
--X.6-- (real) ==>  0
--Y.6-- (real) ==>  0.259
--Z.6-- (real) ==>  YELLOW
--RADIUS.6-- (real) ==>  -99
--COLOR.6-- (char) ==>  *
--X.7-- (real) ==>  *
--Y.7-- (real) ==>  *
```

```
--Z.7-- (real) ==>  *

--RADIUS.7-- (real) ==>  *

--COLOR.7-- (char) ==>  *

Add more tuple in --GEOMETRY-- table ? (y or n) ==>  Y

--GEOM.NAME-- (char) ==>  TRIGONAL BI-PYRAMID

TRIGONAL BI-PYRAMID  is a duplicate primary key !
Duplicate key, Do you want to try again ? (y or n) ==>  N

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==>  1

--------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

--------------------------------------------------------------------------------

Input data  ==>  3

Enter relation name to process :   GEOMETRY

--------------------------------------------------------------------------------
GEOM.NAME X.1       Y.1       Z.1       RADIUS.1  COLOR.1   X.2       Y.2
Z.2       RADIUS.2  COLOR.2   X.3       Y.3       Z.3       RADIUS.3  COLOR.3
X.4       Y.4       Z.4       RADIUS.4  COLOR.4   X.5       Y.5       Z.5
RADIUS.5  COLOR.
--------------------------------------------------------------------------------
OCTAHEDRAL              0.300  0.000 -0.600  1.600 BLUE      -0.300  0.000 -0.6
00   1.600 GREEN        0.000  0.300  0.000  1.600 RED        0.000  0.000  0.0
00   1.600 PINK         0.000 -0.300  0.000  1.600 GREEN      0.300  0.000 -0.6
00   1.600 CYAN
SQUARE PYRAMID          0.300  0.000 -0.600  1.600 YELLOW    -0.300  0.000 -0.6
00   1.600 BLUE         0.000  0.300  0.000  1.600 GREEN      0.000  0.000  0.0
00   1.600 RED          0.300  0.000  0.600  1.600 CYAN      -0.300  0.000  0.6
00   1.600 PINK
TRIGONAL BI-PYRAMID     0.150  0.000 -0.259  1.600 BLUE       0.000  0.300  0.0
00   1.600 GREEN        0.000  0.000  0.000  1.600 RED       -0.300  0.000  0.0
00   1.600 PINK         0.000 -0.300  0.000  0.000 0.15       0.000  0.259  0.0
00 -99.000 *

--------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==>  5
```

```
Please select one number to DELETE one tuple.
    1.  ==>  DELETE MOLECULE.
    2.  ==>  DELETE ATOM TABLE or GEOMETRY.
    3.  ==>  QUIT_JOB      : To exit delete job.
    4.  ==>  To exit CAMD system.

    Enter your selective number  ==>  2

Enter relation name to process :   GEOMETRY

Input data ==>   TRIGONAL BI-PYRAMID

DELETE   GEOMETRY   WHERE  GEOM.NAME =   TRIGONAL BI-PYRAMID
-------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==>  1

-------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

-------------------------------------------------------------------------------

Input data  ==>  3

Enter relation name to process :   GEOMETRY

-------------------------------------------------------------------------------
GEOM.NAME X.1        Y.1        Z.1        RADIUS.1  COLOR.1   X.2        Y.2
Z.2       RADIUS.2  COLOR.2    X.3        Y.3       Z.3       RADIUS.3  COLOR.3
X.4       Y.4       Z.4        RADIUS.4   COLOR.4   X.5       Y.5       Z.5
RADIUS.5  COLOR.
-------------------------------------------------------------------------------
OCTAHEDRAL              0.300   0.000 -0.600   1.600 BLUE        -0.300   0.000 -0.6
00   1.600 GREEN        0.000   0.300  0.000   1.600 RED          0.000   0.000  0.0
00   1.600 PINK         0.000  -0.300  0.000   1.600 GREEN        0.300   0.000  0.6
00   1.600 CYAN
SQUARE PYRAMID          0.300   0.000 -0.600   1.600 YELLOW      -0.300   0.000 -0.6
00   1.600 BLUE         0.000   0.300  0.000   1.600 GREEN        0.000   0.000  0.0
00   1.600 RED          0.300   0.000  0.600   1.600 CYAN        -0.300   0.000  0.6
00   1.600 PINK

-------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  N
        ------------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------
```

```
Please enter one number to process  ==>  4

Draw    1. molecule.  or   2. geometric model.
Input number -->  2

Input data ==>   TRIANGLE

DISPLAY   TRIANGLE
-------------------------------------------------------------------------------

Drawing type ->  TRIANGLE

--- 1st atom ---
x =  0.0000,  y =  0.0000,  z =  0.0000     radius =  1.6000,  color = CYAN

--- 2nd atom ---
x = -0.3000,  y =  0.0000,  z =  0.0000     radius =  1.6000,  color = GREEN

--- 3rd atom ---
x =  0.1500,  y =  0.0000,  z =  0.2590     radius =  1.6000,  color = RED

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  NO
        ------------------------------------------------------------
        1. HELP,     2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==>  TRIANGLE
Error number, Please try again!

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  NO
        ------------------------------------------------------------
        1. HELP,     2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==>  1

-------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

-------------------------------------------------------------------------------

Input data  ==>  3

Enter relation name to process :   TRIANGLE

-------------------------------------------------------------------------------
MOLE.NAME       ATOM.1          ATOM.2          ATOM.3
-------------------------------------------------------------------------------
SO2                     S       O       O
S2O                     O       S       S
H2O                     O       H       H
SC12                    S       Cl      Cl

-------------------------------------------------------------------------------
```

```
To exit CAMD system, please type 'quit' or 'QUIT' ==)  N
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.

Please enter one number to process  ==)  1

1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

Input data  ==)  3

Enter relation name to process :   SQUARE PYRAMID

MOLE.NAME ATOM.1    ATOM.2    ATOM.3    ATOM.4    ATOM.5    ATOM.6
BrF5                F         F         F         Br        F         F
TeF5-               F         F         F         Te        F         F
[SbF5]--            F         F         F         Sb        F         F

To exit CAMD system, please type 'quit' or 'QUIT' ==)  N
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.

Please enter one number to process  ==)  1

Script started on Mon Dec 15 16:22:38 1986
(1)% run

                        ----WELCOME TO CAMD SYSTEM----

                      (Computer Aided Molecular Design System)

        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.

Please enter one number to process  ==)  4

Draw    1. molecule.  or    2. geometric model.
Input number --)  2

Input data ==)   LINEAR
```

```
DISPLAY   LINEAR
-------------------------------------------------------------------------------
                                                                          19.
Drawing type -)  LINEAR

--- 1st atom ---
x =  0.3000,   y =  0.0000,   z =  0.0000      radius =  1.6000,   color = PINK

--- 2nd atom ---
x =  0.0000,   y =  0.0000,   z =  0.0000      radius =  1.6000,   color = YELLOW

--- 3rd atom ---
x = -0.3000,   y =  0.0000,   z =  0.0000      radius =  1.6000,   color = GREEN

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)
      ---------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==)  4

Draw    1. molecule.  or   2. geometric model.
Input number --)  2

Input data ==)   TRIANGLE


DISPLAY   TRIANGLE
-------------------------------------------------------------------------------

Drawing type -)  TRIANGLE

--- 1st atom ---
x =  0.0000,   y =  0.0000,   z =  0.0000      radius =  1.6000,   color = CYAN

--- 2nd atom ---
x = -0.3000,   y =  0.0000,   z =  0.0000      radius =  1.6000,   color = GREEN

--- 3rd atom ---
x =  0.1500,   y =  0.0000,   z =  0.2590      radius =  1.6000,   color = RED

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)
      ---------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
-------------------------------------------------------------------------------

Please enter one number to process  ==)  4

Draw    1. molecule.  or   2. geometric model.
Input number --)  2

Input data ==)   TRIGONAL BI-PYRAMID


DISPLAY   TRIGONAL BI-PYRAMID
-------------------------------------------------------------------------------

Drawing type -)  TRIGONAL BI-PYRAMID

--- 1st atom ---
```

```
x =  0.1500,  y =  0.0000,  z = -0.2590     radius =  1.6000,  color = BLUE

--- 2nd atom ---
x =  0.0000,  y =  0.3000,  z =  0.0000     radius =  1.6000,  color = GREEN

--- 3rd atom ---
x =  0.0000,  y =  0.0000,  z =  0.0000     radius =  1.6000,  color = RED

--- 4th atom ---
x = -0.3000,  y =  0.0000,  z =  0.0000     radius =  1.6000,  color = PINK

--- 5th atom ---
x =  0.0000,  y = -0.3000,  z =  0.0000     radius =  1.6000,  color = CYAN

--- 6th atom ---
x =  0.1500,  y =  0.0000,  z =  0.2590     radius =  1.6000,  color = YELLOW

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)
      ------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==)  4

Draw    1. molecule.  or   2. geometric model.
Input number --)  2

Input data ==)   2


DISPLAY   2
--------------------------------------------------------------------------------

Drawing type -)  TRIGONAL BI-PYRAMID
No this model!

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==)
      ------------------------------------------------------
        1. HELP,      2. Create Molecular Table,   3. List Molecular Name
        4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==)  4

Draw    1. molecule.  or   2. geometric model.
Input number --)  1

Input data ==)   H2O


DISPLAY   H2O
--------------------------------------------------------------------------------

Drawing type -)  TRIANGLE
                 ------------------------------------------

Drawing Atom  = O
x =  0.0000,  y =  0.0000,  z =  0.0000,    radius =  1.4000,  color = RED

Drawing Atom  = H
x = -0.3000,  y =  0.0000,  z =  0.0000,    radius =  1.2500,  color = GREEN
```

```
Drawing Atom  = H
x =  0.1500,  y =  0.0000,  z =  0.2590,    radius =  1.2500,  color = GREEN

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  NO
       --------------------------------------------------------------
       1. HELP,     2. Create Molecular Table,   3. List Molecular Name
       4. Display a Molecular Model,  5. Delete a Molecule,  6. QUIT.
--------------------------------------------------------------------------------

Please enter one number to process  ==>  1

--------------------------------------------------------------------------------
1. Print description
2. Describe attributes properties.
3. Display any data in whole table.
4. To create table, list name, display model, delete molecular.
5. To exit CAMD system

--------------------------------------------------------------------------------

Input data  ==>  3

Enter relation name to process :   TRIGONAL

Unable to find this relation : TRIGONAL
Do you want to try again ? ( y or n )  :  Y

Enter relation name to process :   TRIA

Unable to find this relation : TRIA
Do you want to try again ? ( y or n )  :  Y

Enter relation name to process :   TRIANGLE

--------------------------------------------------------------------------------
MOLE.NAME      ATOM.1         ATOM.2         ATOM.3
--------------------------------------------------------------------------------
SO2                  S        O       O
S2O                  O        S       S
H2O                  O        H       H
SC12                 S        C1      C1

--------------------------------------------------------------------------------


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
To exit CAMD system, please type 'quit' or 'QUIT' ==>  QUIT
(2)% exit
(3)%
script done on Mon Dec 15 18:46:44 1986
```