TOWARDS THE RATIONAL DESIGN OF MATERIALS:
The development of an integrated software package

A Thesis presented to the University of Surrey
for the degree of Doctor of Philosophy in the
Faculty of Science.

By

David Andrew Cole

Structural and Computational Chemistry Group
Department of Chemistry
University of Surrey

September 1992
One of the continuing scandals in the physical sciences is that it remains in general impossible to predict the structure of even the simplest crystalline solids from a knowledge of their chemical composition.
CONTENTS

1 Aims. .......................... 1

2 Introduction. ................. 2
   Computational Chemistry. .... 4
   Areas of application of computers to chemistry. 7
   Computers and computation. 13
   Introduction to solid-state materials design. 23

3 The atom-atom potential method. 27
   Solid-state simulation. ...... 27
   Application of atom-atom potential methods to crystal structure prediction. 27
   Crystal structure determination through packing considerations. 29

4 Molecular mechanics. .......... 30
   Overview. ..................... 30
   The Force-field. .............. 32
   Minimisation of the energy. .. 40

5 Development of the molecular mechanics program MOLMECH. 49
   Problems with the molecular mechanics technique and inorganic compounds. 49
   The program MOLMECH. .......... 50

6 Application of MOLMECH to isolated molecules. 70
   Introduction. ................ 70
   Method. ...................... 71
   Results and discussion. ..... 73

7 Application of MOLMECH to unit-cells. 75
   Introduction. ................ 75
   Method. ...................... 76
   Force-constants. ............ 76
   Assumptions. ................. 80
   Results. ..................... 80
   Conclusion. .................. 82
Display of molecular vibrations. 136
Animations. 138
Files manipulation. 138
Towards more realistic images. 138

13 Graphics programming using Dore'
   Introduction. 151
   Basic steps in programming with the Dore' library. 151
   Objects and methods. 152
   Groups and in-line groups. 153
   Callback groups. 155
   Display groups and studio groups. 156
   Producer-group. 156
   The picking functionality in Dore' 156

14 CRYSTAL - solid state modelling software for the Stardent Titan graphics supercomputer.
   Introduction. 158
   General Features. 159
   General flow diagram. 161
   Crystal 'C' modules. 162
   Program overview 165
   New functions 185

15 Conclusions. 190
(See Volume II for Users/Programers guide, technical reference and program listings).
CHAPTER ONE

AIMS
AIMS

The use of computer graphics for visualising results of scientific and engineering applications has spread widely in the last few years. This growth has spanned a broad variety of fields, including, though not necessarily limited to, computer-aided design, computer-integrated manufacturing, fluid dynamics, physics, biology, biochemistry and chemistry. The particular applications of computer graphics are quite varied within each field, however, they all have the common objective of providing a medium for manipulating and interpreting data from numerical simulation. A key ingredient towards achieving this goal is the need to develop software and hardware tools and to be able to apply the use of computer graphics towards understanding the various scientific fields. In particular, software, must be efficient, user-friendly and, as much as possible, programmed to take advantage of the interactive aspects of the man-to-machine interface. Therefore, optimal programming of highly interactive computer graphics application, remains as one of the major tasks in advancing the use of visualisation in understanding scientific and engineering research.

The aim of this project, then, was primarily to design, code and implement, visualisation and graphics software for both Personal Computer and Graphics Supercomputer environments, to assist in the rational design of solid-state materials. The project also involved an investigation into the use of molecular-mechanical techniques for assisting in crystal structure prediction, which may lead to the development of molecular mechanics computer programs to deal with inorganic compounds.
Overview of the development of the project

Familiarisation period

Computer science has evolved exponentially over the last ten years or so, and the number of programming languages and software tools has likewise grown. In view of this, the initial requirement of the project was to investigate the suitability of computer languages, for molecular modelling applications, and the hardware platforms on which to implement them.

To achieve this, and to produce useful code at the same time, a period was spent producing computer-aided-learning software on P.C's for undergraduate teaching purposes. Familiarisation with computer languages such as Microsoft QuickBasic, FORTRAN and C was gained, together with an understanding of the graphics and processing capabilities of modern personal computers. This introductory period also resulted in the design and implementation of a database of spectroscopic information for teaching purposes.

Molecular mechanics development

Attention was then turned to the main task of inorganic structure design, to provide software tools that would enable this sort of research. This meant carrying out an investigation of simulation techniques such as molecular mechanics. The code for such investigations did not exist, however, so two options were available; firstly to alter an existing molecular mechanics program, such as MM2, so that it could deal with inorganic structures, or to design totally new code. The second option was chosen, as it was felt that code would be more flexible and amenable to modification than that of an existing system.

Molecular mechanics code (MOLMECH) was thus developed in the QuickBasic language for use on a Personal Computer. The implementation was found to be slow in dealing with fairly large molecules, however. It was thus re-coded in VAX-FORTRAN to be run on a Micro-Vax II minicomputer.

Molecular mechanics investigations

The applicability of the code to inorganic systems and metalloproteins was investigated. These investigations highlighted the limitations of molecular mechanics techniques as a tool for prediction in crystal systems, and raised the question of the possibility of making predictions based on the large body of structural information contained within the Cambridge Crystallographic Database.
Structural design package coding

Prediction and Molecular mechanics software needed to be integrated into a molecular modelling package for inorganic crystal 'engineering'. Since no suitable package existed, it was necessary to design and code a package from 'scratch'. A period was then spent in evaluating computer graphics techniques, languages and hardware. The resulting graphics program 'CRYSTAL-PC' enabled the visualisation of crystal structures via an interface to the Cambridge Crystallographic Database. Hardware limitations of PC's were soon realised in attempting to make this application interactive, for altering and modifying structures.

Structural design package coding on state of the art hardware

At this point it was clear that the planned structural design package would need to run on sophisticated graphics hardware, in order to be able to do useful structural visualisations and manipulations. The purchase of a Stardent TITAN graphics supercomputer enabled vectorised code to produce exceptionally high-quality images able to be rotated and interacted with, by a user, in real time. Thus after familiarisation with graphics programming using the 'C'-language and the TITAN graphics language DORE, a complete modelling/visualisation system was designed and coded to run on the TITAN, featuring novel functions to aid in the design of solid-state materials.
CHAPTER TWO

INTRODUCTION
2.1 Computational Chemistry

The computer is now an integral part of every area of modern living. It is not surprising, therefore that computer technology is now being increasingly used in the sciences to aid research in problems that would be too difficult or time consuming to approach in any other way. This is particularly true of many areas of chemistry and the last ten years has seen the computer come to the leading edge of chemical research both in Academia and in Industry.

Modern theoretical science is best described mathematically in terms of a series of complex mathematical equations. Solutions of these equations for particular physical problems are difficult if not impossible to do by manual methods. In fact it is the advent of the increase of computational power that has made solutions to many physical problems possible.

Chemistry is, and has in fact been, one of the primary areas of application for computers in scientific research. Developments in computational quantum chemistry, in particular, have frequently gone hand-in-hand with developments in computer technology and in computer science. Other fields of computational chemistry have been maturing in recent years as chemists continue to identify the advantages that may result from the application of computers and computational techniques to improve their research capabilities.

Computational chemistry can often afford information which is not available from experiment. It can sometimes provide a deeper understanding of chemical problems. Further, it can frequently be more economic to adopt a computational rather than an experimental approach to a particular problem. However, computational and experimental studies often afford complementary information about a chemical system.

Today, the situation has been reached where, in some cases, the computational chemist can substitute the computing machine for the test-tube. Not that the computational approach to the study of chemistry should be regarded as a rival to the traditional experimental techniques. The two approaches are complementary, one approach providing support and data which is not available from the other, and vice-versa.
Sometimes an experiment may be considered too dangerous, or indeed impossible, to perform in the laboratory, and the methods of computational chemistry then provide the only route to the required chemical information. For example, beryllium compounds are notoriously difficult to work with in the laboratory, not least because of their toxicity, while the beryllium atom is rather easily handled by the methods of computational quantum chemistry, since it has only four electrons. Many compounds are too reactive to be isolated and cannot be studied by standard laboratory techniques, such as infrared spectroscopy or nuclear magnetic resonance. Computational studies of unstable species can usually be performed with no more difficulty than the study of similar stable species.

When computational studies of a series of atoms, molecules, or ensembles of atoms or molecules are performed, information can often be obtained that is of help in rationalising the observed properties. Useful chemical concepts emerge and can be used to rationalise vast quantities of data. Such concepts can be of great value in the planning of future studies of similar atoms and molecules.

Laboratory experiments can be both expensive and time consuming. The use of the methods of computational chemistry rather than an experimental approach to the determination of chemical properties can increasingly be justified on economic grounds.

The description of a given chemical system by computational techniques requires the formulation of a physical model. The complexity of this model will be constrained by the complexity of the chemical system being considered. If we are interested in the properties of a single molecule, such as a radical in interstellar space or an ion which is believed to catalyse certain reactions in the upper atmosphere, then we can treat the entire system quantum mechanically from first principles. We can perform ab-initio calculations to determine the structure and properties of a small molecule from the basic laws of quantum mechanics. If, on the other hand, we are interested in bulk properties, such as defects in a crystal or the thermodynamic properties of liquid argon, then having no possibility of studying the motion of a mole of argon atoms in a computer simulation (1 mole contains $10^{23}$ atoms) even within the framework of classical mechanics, we have to resort to statistical methods. To perform such a study, we need to have a detailed knowledge of the interactions between a single pair of atoms or molecules; this can often be obtained from ab-initio calculations. Clementi and co-workers\textsuperscript{1} summarised the situation as follows:
"Theoretical chemistry should attempt to provide an overlapping set of models to describe a chemical system of any degree of complexity. Computational chemistry, a much younger subject, attempts to provide operational techniques for solving such models and to test for the validity of the models by comparing simulated and experimental data."
2.2 Areas of application of computers to chemistry

The computer is now extensively used throughout chemical research and development both academically and industrially. The areas of application have spread rapidly and it is now highly likely that a research chemist will use a computer at some stage of his everyday work. The general areas of application of computer technology to research are listed below, although many applications overlap these general areas:

A \ Calculation and computation
B \ Simulation and visualisation
C \ Control and instrumentation
D \ Education
E \ Artificial Intelligence and expert systems
F \ Databases
G \ Administration

2.2.1 Calculation and computation

The formulation of quantitative models which have predictive value is an integral part of chemistry. Few such models can be solved analytically and it is usually necessary to resort to numerical methods.

During the past forty years there has been an enormous increase in the power and efficiencies of computers and it is now possible to obtain by numerical methods chemical information that would be impossible or very difficult to obtain by other means.

2.2.1.1 X-ray crystallography

Progress in crystallography is easy to understand, as it is related to progress in computer and diffractometer technology. Computers are used both to control the diffractometer and to provide an aid to the solution of the equations necessary to discover the structure of crystals. Before the use of computers, structures solved by hand calculation took a great many months to solve. Computer technology has reduced this time to a number of days.

The computer is an irreplaceable tool for the driving of a diffractometer. However, some problems cannot at present be solved by the computer alone, for
instance, the unambiguous determination of the space-group; therefore the computer must be placed under the control of a qualified operator. The role of the operator is essential in order to avoid faults in the unit cell determination, to detect systematic errors and to reach the optimum accuracy in the intensity measurements.

For medium sized molecules, e.g. 50 to 100 atoms, it is true that the solution of the structure can often be given by direct methods. Thus the computer is able to calculate phases for normalised structure factors and to compute Fourier-series, and it can output a diagram as a representation of the largest peaks of the E-maps, but it is unable to translate this diagram into terms of molecular structure. This interpretation requires some imagination and intuition which are still not included in any software. This must still be done by the chemist, who tries to build up the molecule by correlating each peak to an atomic species in such a way that the emerging molecule is chemically consistent and could eventually be completed by Fourier analysis and properly refined.

An important consequence of the efficiency of modern crystallography is that more investigations can be dedicated to the preparation and to the significance of the results rather than to the means used to obtain them. Thus, scientists involved in structural studies can devote their time to the best choice of molecules to be investigated and can design experiments in order to obtain crystals suitable for an X-ray study.

2.2.1.2 Quantum chemistry

At the heart of quantum chemistry, molecular biology and quantum pharmacology is the Schrodinger equation. The Schrodinger equation for a molecule containing more than one electron and two nuclei cannot be obtained exactly and the systems become progressively more complex and difficult to solve as the number of atoms and their atomic weight increase. The solution of such problems undoubtedly requires the calculating power of modern computers.

2.2.1.3 Physical Chemistry

The computer is used as a calculation tool in other areas of physical chemistry such as the study of chemical reactions, kinetics and reaction dynamics.

2.2.2 Simulation and visualisation

Modern theoretical science relies on the solution of complicated mathematical equations the results of which produce vast amounts of data in the form of numbers
which must be interpreted by the scientist to extract form and meaning. Dealing with raw data does not provide an intuitive way of assessing the results of calculations. Computer graphics is revolutionising the way in which results from theoretical calculations can be visualised and understood. Multidimensional data can be viewed by using combinations of axes and colours to represent dimensions allowing patterns to emerge out of the mass of data. In chemistry, visualisation can allow the results of quantum mechanical calculations to be viewed in meaningful ways.

With the advent of powerful computer hardware it has become possible to devise real time simulations of model systems allowing the interactive changing of parameters while the simulation is running allowing the effects to be visualised.

2.2.2.1 Simulation of liquids and solids

Monte Carlo techniques and molecular dynamical techniques are used in the modelling of a small number of particles, typically 100 to 1000, and allowing for the fact that the real system contains many more particles than can actually be included in the simulation, by adopting periodic boundary conditions and the minimum image convention.

Computer simulation techniques have now been developed to a level of sophistication where they can provide a reliable predictive tool. They enable computational chemists to study the thermodynamic, structural and transport properties of liquids.

A solid is essentially an ordered array of atoms, with amorphous solids being less ordered and a crystalline solid being highly ordered. Although the numbers of atoms in a typical solid is very large indeed, the high symmetry that is present simplifies the theoretical description of the solid state immensely. The basic computational problem in performing a static simulation of a solid is essentially one of minimising the energy of a static lattice, (ignoring thermal vibrations) with respect to the coordinates of the component atoms and/or ions. This is achieved by use of optimisation techniques of the type discussed in more depth in Chapters 3 and 4.

2.2.2.2 Proteins and peptides

In contrast to studies of the liquid state or of polymers, the Monte-Carlo method has not been widely used for proteins or nucleic acids. The main techniques for
investigating these molecules is molecular dynamics. However, calculations are very
demanding in terms of computational resources, so simulations cannot be run for times
much longer than a few ns. Thus processes which occur over larger times cannot be
easily investigated.

2.2.3 Control and instrumentation

Nearly all modern chemical instrumentation is micro-processor controlled. Nmr
(Nuclear Magnetic Resonance spectroscopy), M.S. (Mass Spectrometry) and other
analytical instrumentation use computers for control, data storage and processing.
Microcomputers are commonly used for data capture in laboratory experiments and
most analytical laboratories now have a LIMS (Laboratory Information Management
System) for data processing.

2.2.4 Education

2.2.4.1 CAL

In recent years experiments have been made to test the use of computers as a
media for teaching chemistry. This ranges from the most basic drill and test packages
to complex interactive games and simulations. More recently computers have been
interfaced to video and CD Rom hardware to provide a multimedia environment within
which to interactively explore and participate within a field of inquiry. The use of
interactive graphics and animation can make computers a useful tool in conveying
chemical concepts.

2.2.4.2 Expert systems

Described in 2.2.5.1 expert systems can be used as a training tool for certain
diagnostic processes.

2.2.5 Artificial intelligence

2.2.5.1 Knowledge based systems

Every program contains knowledge about some problem. What makes knowledge
systems different from conventional programs is that they represent the knowledge in
a higher level form. Instead of encoding knowledge in low-level statements, they store
it in a knowledge base of rules and facts that stay close to the way people think about a problem. Expert systems are problem solving systems that reach expert or at least highly competent levels of performance.

In chemistry, expert systems have been developed to assist in, for example, areas of computer-aided organic synthesis and computer-aided metabolism prediction.

2.2.6 Administration

Microcomputers are used extensively in report writing, spreadsheeting and scientific communications, in the designing and production of highly effective posters and in the preparation of papers etc.

2.2.7 Databases and archives

The enormous amount of new compounds being synthesised each year together with the vast amount of research data being generated has brought about the need for a flexible storage system which can be searched quickly and efficiently, and requires the minimum of storage space. Computers provide the ideal solution to the problem providing electronic storage of vast amounts of data and rapid searching of required data.

The most widely known chemical structural databases are the Cambridge Structural Database (CSD), containing organic & organometallic structural data, the Inorganic Chemical Structural Database (ICSD) containing inorganic structural data and the PDB (Protein Data Base) containing bio-macromolecular structural data.

Databases are now a central tool in any modern modelling system.

2.2.8 Information systems

Chemical information is a large field proceeding from simple scientific journals to patents, reports, dissertations, conference proceedings, books etc. The growth of chemical and biochemical information is tremendously rapid. This means that the modern chemist, if he really wants to cope with his discipline, must give a large portion
of his time to chemical information. The number of various publications is now so great that computer help becomes increasingly necessary. Moreover, if we want to deal with information concerned with structures in combination with properties, synthesis, etc, the amount of information rapidly becomes vast. Systems such as CAS online (Chemical Abstracts) allow interactive searching for relevant information.
2.3 Computers and Computation

2.3.1 Computers, Computer Science, and the Computational Sciences

Recent years have seen considerable progress in computer technology, in computer science, and in the computational sciences, which include computational chemistry. To a large extent developments in these fields have been mutually dependent. Progress in computer technology - such as the use of very large-scale integrated circuits - has led to increasingly larger and faster computing machines, the so-called supercomputers, as well as powerful minicomputers. At the same time, research in computer science has explored new methods for the optimal use of these resources, such as the formulation of new algorithms that allow for the maximum use of parallel computation.

Developments in computer technology and computer science have had a very significant effect on the computational sciences, and in particular on computational chemistry.

Well established techniques have been reformulated to make more efficient use of the new computer technology. Algorithms, which only a few years ago were considered to be computationally intractable, have now been successfully implemented. This research has given new and exciting insight into molecular structure and molecular processes by enabling small systems to be studied in greater detail than was previously possible and by allowing larger systems to be studied by computational methods for the first time.

Technological developments have not only extended the accuracy and the range of problems that can be tackled by means of computational methods, but have also enhanced our ability to visualise the results of such calculations.

Computer graphics are continually increasing both in quality and in sophistication. Graphics methods, and in particular interactive graphics, provide an invaluable method for displaying the large amounts of information often derived from complicated calculations in a form that can be readily understood. In computational chemistry, graphics are often employed to display molecular wave functions, potential energy curves and surfaces, molecular conformations etc. Indeed, graphics have already proved to be of great value in displaying the results of ab-initio studies in small molecules, in studies of
chemical reactions, in displaying the results of molecular mechanics calculations, in quantum pharmacology, in molecular biochemistry, and in many other areas.

2.3.2 Hardware

In order to make efficient use of a mainframe computer, the computational chemist must be aware of the fundamentals of its architecture. The simplest model of a computer consists of the following basic units:

1. A central processing unit (CPU)
2. Memory
3. Input/output
4. Data lines or data busses.

This is illustrated in fig. 2.1

At the most basic level a computer consists of a central processing unit (CPU).
which actually executes the program and performs the computation; a memory, in which the program and any data it is to process are stored, and input/output (I/O) devices, by means of which the machine communicates with the user. Transfer of programs and data between the CPU, the memory, and I/O devices takes place by means of the data lines or data busses.

Programs and the data which may be processed are entered into the computer by means of the I/O devices and stored in the memory. The CPU executes the program sequentially, that is serially. The instructions contained in the computer program must be transferred from the memory to the CPU together with any data that may be required. The resulting data must be returned to the memory. Transfer of data between the various devices takes place by means of the data lines or data busses.

The simple model of a computer described above becomes much more complicated in reality. The memory may include random access storage devices, such as fixed disks or access through the computer operator to large banks of data stored on magnetic tape. Input devices may include input from visual display units (VDU's), from experiments, from magnetic tapes, or from other computers. Output devices may include line printers, graphics units, or microfiche. The configuration of a typical mainframe computer is shown in figure 2.2.
The mainframe computer communicates directly only with the magnetic disks, which are used for storing programs and data, and with the front-end computer. It is this front-end computer that handles the communications with a wide variety of input/output devices, which may include magnetic disks, or, if large quantities of data are to be stored, magnetic tapes, visual display units (VDU's) by means of which a user may enter programs and data, card readers, line printers, micro-fiche output, and graph-plotting facilities. If a number of computers are linked together to form a network, then the front-end computer will also handle network communications.
In the simple model of the computer given above it is clear that, for example, the CPU may be idle while data are being transferred to and from memory, and the data lines may be idle while the CPU executes the instruction. This performance can be significantly improved if we organise the computer so that various tasks can be performed simultaneously or in parallel.

Parallelism can be introduced in essentially four different ways:

1. By using several processors, each with its own instruction set and communicating via a common memory. This is usually referred to as multiprocessing.

2. By using an array of identical processing elements under common control. This is termed array processing and involves the performance of the same operations on different data.

3. By using independent units under common control for performing different functions. This is functional parallelism.

4. By using independent units for the various subtasks of the same operation processing different data in parallel and under common control. This is usually referred to as pipelining.

Fig 2.3 illustrates the relation between a serial computer, a machine which uses pipelining, and an array processor. Frequently, some or indeed all of these different methods for performing concurrent computation are employed in actual computer architectures.
2.3 Serial computers, vector-processing computers, and array processing computers. The addition of two numbers by a computer involves four steps:

1) The comparison of the exponents
2) A shift of the mantissa
3) The actual addition
4) Normalisation

The execution of each of these steps requires one clock period. The process of adding two numbers is represented in the figure (top) by four numbered boxes, one for each of the sub-operations listed above.

In a serial computer (center), the suboperations are performed sequentially and thus four clock periods are needed to obtain a result. If a number of additions must be performed, then a higher rate of computation may be achieved. For a vector-processing computer (left), the suboperations for different additions are overlapped; the operations (i)-(iv) may be performed for different pairs of numbers, which are to be added simultaneously. A vector-processing computer, therefore, requires one clock period for each addition. For an array-processing computer (right), which consists of N serial processors working in parallel, four clock periods are required to produce N results.

For an array processor consisting of N vector processors working in parallel, only one clock period would be required per result.

2.3.3 Supercomputers

Supercomputers have revolutionised scientific computing over the past decade by bringing vast computational power to bear on scientific problems. In fields such as biotechnology, earth sciences, meteorology, and inorganic chemistry, supercomputers have reduced by several orders of magnitude the time required to solve problems, as well as permitting analyses that would have been considered impractical a few years ago. Supercomputer technology has also helped create a revolution in design and
manufacturing, making feasible, device, process, and circuit simulations and design rule checking.

To model the physical world scientists and engineers must analyse highly complex systems of equations with enough precision to allow accurate forecasts and other measurements to be made.

Graphical analysis and visualisation are essential. They provide understanding and insight only possible with the eye, they save development time and cost, and they have the potential to make comprehensible the voluminous quantities of data continually being generated by earth and space monitoring equipment. As highlighted in the recent National Science Foundation Visualisation Report, visualisation is emerging within its own right, joining mathematics and computer science as analytical tools for science and engineering. Supercomputers have made this possible.

The transition to supercomputer analysis requires a change in thinking for scientists and engineers. In addition to faster machines, supercomputer technology uses parallel and vector processing to make algorithms run efficiently. Parallel processing allows multiple programs or pieces of a program to execute at once, while vector processing (combined with pipelining) allows multiple iterations of an operation to be executed efficiently, with few machine instructions. To take full advantage of supercomputers, scientists and engineers must analyse problems with parallel processing in mind and use vectors as the primitives for operations and algorithms. Once the commitment is made, however, the benefits are significant and lasting.

2.3.4 The Titan Supercomputer

Titan, the graphics supercomputer used for the code development, combines the vector performance of a supercomputer and the graphics performance of a graphics workstation together in one machine.

fig. 2.4 shows the hardware setup for the Titan in the University of Surrey Chemistry department.
Titan's implementation of two fundamental hardware architecture objectives contributes to its supercomputer performance and high speed graphics.

- Have multiple pieces of hardware that run in parallel
- Keep all the hardware continually busy

Each Titan processor board has an integer unit, which keeps continually busy by pipelining integer instructions, and a vector unit, which keeps continually busy by pipelining and chaining vector instructions.
Titan's integer unit is a Reduced Instruction Set Computer (RISC) processor that operates on integers. A RISC processor contains only the simple instructions that are used for over 95% of processing. A Complex Instruction Set Computer (CISC) processor contains more instructions than RISC processor, but many of the CISC processors instruction are seldom used. Also, many CISC processor instructions are complex and require more time (machine cycles) to analyse and execute. Because it takes fewer machine cycles to complete simple instructions on a RISC processor, the same program runs faster on a RISC machine than on a CISC machine. Even a complex operation, which requires more instructions on a RISC machine than on a CISC machine, often runs faster on a RISC processor because of the reduced cycle time.

Titan's vector unit is a vector register machine that operates on floating point vectors, integer vectors, and floating point scalars. Vectorised functions may run more than ten times as fast as comparably coded scalar functions. However not all code can be vectorised. Code that can't be vectorised requires fast scalar processing. Titan achieves fast scalar processing by pipelining floating point scalar operations.

Furthermore, since Titan's vector unit processes both floating point scalars and vectors, Titan has less hardware than machines with separate vector and floating point units.

Titan can be configured with up to four identical processor boards. Titans multiple independent processor boards coupled with Titans fast synchronisation and low synchronisation overhead make parallel processing possible. Titan supports three types of parallel processing:

1. Multiprocessing - the parallel execution of multiple processes on multiple processors.

2. Microtasking - the division of a process into multiple threads (microtasks) that can be executed in parallel on multiple processors.

3. The parallel execution of integer operations (address calculations, bitwise operations, etc.) and floating point operations within each Titan processor.

Titan's operating system is based on the standard AT&T System V Release 3 UNIX operating system with Berkeley 4.3 UNIX operating system extensions.
2.3.5 TITAN PERFORMANCE SUMMARY

Processing power
TITAN can be configured with multiple processors. Each of TITANS processors comprises an integer unit and a vector unit. The peak performance of one TITAN processor is:

- 16 MIPS for the integer unit
- 16 MFLOPS for the vector unit

(MIPS; Million instructions per second, MFLOPS; Million floating point operations per second)

Graphics performance
When processing graphics, TITAN'S

Vector units perform the floating point operations for graphics transformations, clipping, shading, and perspective divides.

Integer units act as display list processors.

TITAN'S peak hardware Gouraud shading rate is 50 million pixels per second.

TITAN'S peak hardware vector drawing rate is 11.6 million pixels per second.

TITAN allows 24 or 48 colour planes, plus four overlay planes, three control planes and a hardware Z-buffer.

TITAN allows the rendering of 21 thousand independent triangles per second (culled, trivially clipped, full colour, Gouraud shaded).

Graphics software
To support development of graphics applications, TITAN provides the Dore library, TITAN'S dynamic object rendering environment. Dore allows an application to support numerous objects, multiple light sources, and a full range of rendering styles.

(See Chapter 14 Graphics programming using Dore)
2.4 Introduction to Solid-State Materials design. 'Crystal Engineering'.

Computational methods have initially concentrated on problems in organic chemistry. Today the largest use of computational chemistry is in protein research. With nmr data now being added to the information from X-ray crystallography, coupled with the advances in protein engineering, the area seems most promising for molecular design. Pharmaceutical companies have invested heavily in computational equipment and research. It is estimated that, starting from scratch it can take a thousand compounds to be synthesised, screened and tested, before a drug can be brought on to the market, with a cost of about ten million pounds and a time scale of up to fifteen years. Obviously any rational steps in the design stages, that can cut down these overheads is a great advantage.

The same principles can be applied to the design of inorganic solid-state materials, with special reference to the design of semi-conductors, super-conductors and catalysts.

2.4.1 Square Planar Complexes as Possible Semiconductors

Square planar complexes, and in particular square planar macrocyclic complexes, of transition metals have already proved to be good conductors. Several examples of compounds which conduct are glyoximates, dibenzotetraazannulenes, hemiporphyrazines and phthalocyanines.

In all these cases the central metal atom has the possibility of bond formation
perpendicular to the square coordination plane and partial oxidation with a halogen, (iodine is most commonly used), afforded molecular conductors.

A few square planar chromium(II) complexes are known, but until recently none were known to be able to interact strongly with the chromium ion on an adjacent molecule to afford a conducting pathway. The X-ray crystallographic investigation of \([\text{Cr(NCS)}_2(\text{thiourea})_2]\) shows the structure to comprise stacked trans-square planar units with a direct, though distant (3.97 Å), Cr-Cr interaction. However, as the steric requirements of \(\text{Cr}^{8+}\) are nearly always octahedral, and the attached ligands are only simple monodentate ones, the square planar structure would be destroyed on partial oxidation.

Quadridentate ligands e.g. macrocycles, should hold the chromium ion firmly in a planar environment, even on oxidation. Consequently, research has aimed to synthesise square planar complexes using a variety of nitrogen donor ligands including macrocycles such as tetraazaanulenes, as well as sterically constrained ligands e.g. tetramethylated phenanthrolines.

The underlying theme of all this work was to produce one dimensional chain structures with metal to metal distances close enough for an interaction to occur. See fig 2.5
Obviously it would be of great advantage to be able to engineer such structures with the correct M-M distance, rather than to synthesise numerous macrocyclic compounds in the hope that they will form the required chain structures in the solid-state.

2.4.2 Crystal Engineering

Most physical and chemical properties of organic and inorganic solids are controlled by intermolecular orientation and packing. Thus we would like to engineer the required structure that will produce the required properties. In order, therefore, to engineer or design materials with optimal properties it is imperative to be able to understand the laws governing the packing in organic and inorganic crystals. These laws are, however, subtle and complex. While some relationship must exist between molecular and crystal structure, effectively bridging the viewpoints of organic.
inorganic and solid-state chemists, its discovery remains elusive.

We are very far from the ab-initio prediction of solid-state structure given the corresponding molecular structure. We must therefore turn to empirical methods, to use our knowledge of known structures to derive new ones.

There are currently two avenues of exploration in this question:


2. Using statistical techniques - chemical databases.
CHAPTER THREE

THE ATOM-ATOM POTENTIAL METHOD
3.1 Solid State Simulation

The basis of the simulation technique is the specification of an interatomic potential for the system, i.e. an analytical or possibly a numerical description of the energy of the system as a function of atomic coordinates. For polar materials, the model must include a coulombic term, a short-range energy term and an ionic polarisation term.

The calculation of crystal structures with the atom-atom potential method concerns, as with molecular mechanics, the prediction of the minimum-energy configuration (i.e. cell dimensions and unit-cell coordinates) of a crystal structure. To achieve this, lattice energy calculations are coupled with minimisation procedures based where possible on Newton-like methods, but employing conjugate gradient techniques for large, complex structures.

3.2 Application of atom-atom potential methods to Crystal Structure prediction

The application of the atom-atom potential method to the prediction of stable crystal structures has been a very active area of research. In their monograph, Pertsin and Kitaigorodskii mention that there are at least 200 publications which use the method to rationalise or predict equilibrium crystal configurations. An earlier review by Ramdas and Thomas surveys the literature up to 1978. Simple applications of the atom-atom method confirm that observed structures correspond to potential energy minima. It is more challenging to use the method to compare energies of observed and hypothetical structures and, in what is a variant of the same theme, to compare the energies of polymorphs. Another application, most successful for hydrocarbons, involves the use of potential parameters to predict an unknown crystal structure given the unit-cell dimensions. With respect to crystal structure prediction there has been much debate as to the significance of each of the terms in the potential energy expression. The disappointing aspect of all these computational methods is that only very small energy differences are obtained between alternate packings for which clear preferences exist in the crystal.

The usual procedure in empirical energy calculations in crystals is to vary the lattice parameters and molecular orientations slightly, retaining the original crystal
The position of the local minimum closest to the observed structure is determined. The agreement between the experimental structure and the energy minimum is usually excellent, about 0.1 Å for the cell parameters and translations and 1° for the orientations. Such agreement is not surprising. We should recall that the potential parameters are usually derived from the observed crystal structures of closely related compounds, in other words, they have been carefully chosen and tested. Such parameters, it should be stressed, will usually confirm an energy minimum in the vicinity of the observed structure but do not locate other, more distant minima in the potential energy surface. So there is no way of determining if the observed structure corresponds to a local or global minimum. Failure to locate alternative energy minima is due to the fact that the minimum corresponding to the observed structure is usually deep enough so that any energy minimisation from a nearby starting point will collapse into the observed structure.

In practice, not many of these calculations have general predictive value. It is true that they reproduce observed crystal structures but this is only to be expected from the nature of the method. In many cases, the intermolecular interactions are so complex and varied that the derived potential parameters are not valid outside the basis set of compounds. It would seem that indiscriminate application of the atom potential method to all sorts of crystals including highly ionic inorganic materials and the use of the energies derived there from to draw structural conclusions and predictions is highly misleading.

In general, the basic dilemma in these methods is that alternative structures are seen to differ very slightly in energy and these differences are quite dependent on the potential parameters. Therefore the numerical significance of these calculated energy values must be critically evaluated. However, small energy differences can lead to dramatic chemical effects. There is reason to believe that the aggregation of molecules and their nucleation in the pre-crystallisation stage is controlled by small energy differences. It is not that the atom potential method is incorrect but rather that it is not appropriate, only providing limited information about factors governing the packing of molecules in crystals. It is important to ascertain just why the method is so insensitive to structural variations and the situations where it may be used successfully to predict rather than simply confirm observed crystal structures.
3.3 Crystal Structure Determination Through Packing Considerations

The method commences with the isolated molecule whose geometry is estimated. This intramolecular geometry may be quite approximate. The molecule is shifted in the unit cell so as to vary three translations (along the cell axes) and three Eulerian angles of rotation. The positions of the symmetry related molecules are generated since the method presumes knowledge of the space-group. The energy of the starting structure is determined with a 6-exp or any other potential scheme of choice. Typically, several repulsive contacts are found in the beginning. After small variations in translation and/or rotation, the energy of the structure is recalculated. Ideally, the number of these repulsive contacts and therefore the potential energy decreases as the structure approaches a local minimum. The method works quite well for hydrocarbons and somewhat less satisfactorily for hetero-atom derivatives.

There is no doubt that these methods work well when coupled with a minimum of chemical or topochemical information. Without a knowledge of the unit-cell on the other hand, that is when no crystal structural data is available, it is a formidable task to determine the crystal structure with packing calculations. Accordingly, it does not appear that packing analysis affords an improved method of predicting crystal structure in either case.

A serious objection to the use of simple packing calculations in deriving crystal structures concerns the inability to model hetero-atoms and electrostatic interactions, of obvious importance in inorganic chemistry. Much work has been done in the modelling of hydrogen bonding interactions but they have not led to a predictive insight in all but the simplest of cases. The situation with regard to weaker electrostatic interactions is equally unsatisfactory.

Ideally, theoretical models of intermolecular interactions should be simple and of wide applicability or else be based on a well-defined and rigorous physical model. With regard to molecular crystals, the former approach gives an approximate description but does not stress adequately many structure defining interactions; the latter approach seems too difficult with present computer capabilities. Problems with either approach result mainly from the fact that hetero-atom interactions are directional. These interactions are, however, of crucial importance to crystal engineering because of their long range character.
CHAPTER FOUR

MOLECULAR MECHANICS
4.1 Molecular Mechanics Overview

For large molecular systems the methods of computational quantum chemistry become intractable with present-day computing equipment and, indeed, with computers that may become available in the foreseeable future. By abandoning the strictly ab-initio approach and developing semi-empirical formalisms, we can extend the range of problems to which the methods of computational quantum chemistry can be applied.

These semi-empirical methods are faster than the corresponding ab-initio calculations by perhaps one or two orders of magnitude for small molecules and by more for larger systems. However for large molecules, with many degrees of freedom even these semi-empirical approaches can become intractable. We are still a long way from the ideal of obtaining chemical results by solving the Schrodinger equation for a system of arbitrary size and complexity.

In order to study problems involving large extended structures, the computational chemist turns to a technique known as molecular mechanics.

Molecular mechanics is basically a computational realisation of the ball-and-stick models of molecules that are familiar to every chemist. It is based on the Born-Oppenheimer separation of the electronic and nuclear motion. The electrons are not considered explicitly. They provide an effective potential, which is experienced by the nuclei. Molecular mechanics treats the interaction between the nuclei and the effective potential generated by the electrons according to the laws of classical mechanics.

The techniques of molecular mechanics have a number of advantages over the traditional ball-and-stick models. New models can be generated quickly and easily. Many models can be stored in a data-base.

When used in conjunction with computer graphics, molecular mechanics can afford considerable insight into complex problems. Interactive computer graphics allow the chemist to manipulate the model quickly and easily.

The crucial first step in the computational implementation of the method of molecular mechanics is the development of the force-field, that is, a description of the various interactions within the system being studied. To obtain an estimate of the
equilibrium geometry from this force-field it is necessary to minimise the energy with respect to the structural parameters.

In using the method of molecular mechanics we begin, just as we do when using the traditional ball-and-stick models of molecules, by constructing a mechanical model for a molecule that consists of a number of masses, representing the atoms, joined together by springs, which represent the bonds. Any distortion of the model results in an energy change, which we can, of course, calculate provided we know the force laws and force constants involved.

When we use the ball-and-stick models of molecules the force laws and constraints are fixed by the nature of the materials employed in the construction of the model, which may bear little relation to the situation in the real molecule. The computer model is not subject to these restrictions. In the computer model, we can modify the force laws and force constants to reflect the properties of the actual molecular systems we are studying more closely. The key to the successful implementation of the molecular mechanics method lies in the development of reliable descriptions of the interactions present in molecules. We have to obtain reliable force-fields.

Force-field calculations have their origins in vibrational spectroscopy. However, the force-fields developed by vibrational spectroscopists are found to be of limited use in molecular mechanics calculations because they were not parameterised with structure and energy calculations in mind. For example, vibrational force-fields typically neglect Van der Waals interactions, which can be of crucial importance in, say, the study of biological systems.

Allinger summarises the fundamental idea behind the use of the force-field in molecular mechanics as follows; "We have a great deal of experimental information regarding small molecules, such as bond lengths, angles, strain energies, and so on. A large molecule consists of the same features we already know about in small molecules, but combined and strung together in various ways. Can we, with the help of current structural theory, formulate the structure of a large molecule in terms of the elementary features of small molecules?". The answer to this question is generally yes; we have to construct a force field.
4.2 The Force-Field

In general, the force field for an arbitrary molecule may be written as

\[ V = \frac{1}{2} \sum_{ij} k_{ij} x_i x_j + \frac{1}{3!} \sum_{ijk} k_{ijk} x_i x_j x_k + \ldots \]  \hspace{1cm} (4.1)

where \( k_{ij} \) are the quadratic force constants, \( k_{ijk} \) the cubic force constants etc., and \( x_i \) are the coordinates of the nuclei, the displacements from their equilibrium positions. Not all the force constants in equation 4.1 are independent. Chemists usually isolate the independent force constants by making the "valence force field" approximation in which the force constants are directly related to changes in the bond lengths and bond angles.

A molecular force-field describes the potential energy of a molecule relative to the energy of some reference geometry. The force-field contains parameters, the force constants, which are derived inductively by a systematic comparison of calculated and observed molecular properties. We aim to be able to handle a wide range of organic and biological structures with a reasonably small number of transferable parameters. The transferability of parameters becomes a problem when dealing with inorganic structures, as described in chapter 9.

The force-fields employed in molecular-mechanics calculations are almost invariably taken to have the form

\[ V = V_b + V_\theta + V_\phi + V_\chi + V_{ab} + V_a + V_{hb} + (V_{13}, V_{26}) \]  \hspace{1cm} (4.2)

in which the first term, \( V_b \), is associated with bond stretching, \( V_\theta \) with bond angle bending, \( V_\phi \) with bond torsion, \( V \) with out-of-plane bending, \( V_{ab} \) with nonbonded interactions, \( V_a \) with electrostatic interactions and \( V_{hb} \) with hydrogen bonding.
The bond-stretching force-field is often assumed to be given by Hook's law, namely

\[ V_b = \frac{1}{2} \sum_{\text{bonds}} K_b (r - r_0)^2 \ldots 4.3 \]

in which \( k_b \) is the harmonic force constant for the stretching of a particular bond, \( r \) is the bond length and \( r_0 \) is some reference bond length. Both \( K_b \) and \( r_0 \) are taken to be adjustable parameters in the determination of the force-field. The Hook's law description of bond-stretching is shown in fig 4.1.

![fig 4.1 Hooke's law description of bond stretching](image)

Of course Hook's law is only a good approximation for small displacements from the reference configurations. For larger displacements, cubic and quartic terms should be added to equation 4.1. (see fig 4.2).

![fig 4.2 More realistic lower curve by adding cubic terms](image)
Alternatively, the force-field for the bond stretching may be represented by the Morse function

\[ D(1 - \exp[-a(r - r_0)])^2 \ldots .4.4 \]

in which D and a are parameters. The form of the Morse function is displayed in fig 4.3

![Morse curve](image)

*fig 4.3 The Morse curve*

The bond-angle bending energy is also represented to a first approximation by Hooke's law:

\[ V_\theta = \frac{1}{2} \sum_{\text{all bend angles}} k_\theta (\theta - \theta_0)^2 \ldots .4.5 \]
in which $k_0$ is the harmonic force constant for bond-angle bending, $\theta$ is the bond angle, and $\theta_0$ is some reference bond angle. As in the case of bond stretching, $k_0$ and $\theta_0$ are adjustable parameters. It is a well-known fact that it takes more energy to stretch a bond than it does ions of the bond angles than the bond lengths are therefore common, and the addition of cubic and quartic terms,

$$V_0 = \frac{1}{3!} k_0 \theta_0^3, V_0' = \frac{1}{4!} k_0' \theta_0' \theta_0'^3 \ldots$$

where $V_0$ is more frequently necessary.

Bond torsion is of great importance in structural chemistry. The torsional angle $\phi$ (also known as the dihedral angle, or sometimes the twist angle) is defined in fig 4.4

![Fig 4.4](image)

**Fig 4.4 The torsional or dihedral angle**

A simple approximation to the energy associated with bond torsion is

$$V_\phi = \frac{1}{2} \sum_{\text{dihedrals}} k_\phi [1 + m \cos(n \phi)] \ldots$$

in which $K_\phi$ are adjustable parameters and the values of $m$ and $n$ are determined by the nature of the torsional motion. The value of $m$ is set equal to +1 if the staggered conformation has the lowest energy and to -1 if the eclipsed conformation has the
lowest energy; \( n \) determines the periodicity of the torsional motion. For ethane, the torsional term has the form

\[
V_\phi = -\frac{1}{2} k_4 [1 - \cos(3\phi)]
\]

This energy function is illustrated in fig 4.5

More generally, the torsional energy is expressed as a Fourier series

\[
V_\phi = -\frac{1}{2} \sum_{\text{dihedrals}} \sum_n k_4^{(n)} [1 + n \cos(n\phi)]
\]

An example of the application of this more general form for the torsional energy is given in fig 4.3.

Torsional motion requires less energy than bond-angle bending, and therefore the distortion of a molecule by rotation about a dihedral angle is usually more pronounced than bond-angle bending. Indeed, in the treatment of very large molecules, such as polypeptides, the bond lengths and bond angles are often held fixed and only the dihedral angles are varied.
Non-bonded interactions are recognised to be one of the most important aspects of structural chemistry and yet, of all the terms in the force-field, the model from which they are determined is the most uncertain. It is frequently assumed that the nonbonded interactions are pairwise additive and that the interaction can be represented either by

\[ V_{nb} = \sum_{\text{Nonbonded pairs}} \sum A \frac{r^{-a}}{r_0^{-a}} - B \frac{r^{-6}}{r_0^{-6}} \ldots 4.10 \]

or by

\[ V_{nb} = \sum_{\text{Nonbonded pairs}} A \exp(-b r) - B \frac{r^{-6}}{r_0^{-6}} \ldots 4.11 \]

In the first expression, eqn 4.10, the repulsive part of the potential is taken to be inversely proportional to the distance separating the atoms raised to the ath power, where a is in the range 9 to 12. In the second expression, (4.11), the repulsive part of the potential is represented by an exponential term. A third potential function, which is sometimes employed for non-bonded interactions, is the Hill equation, which has the form

\[ V_{nb} = e^{-c_1 (\frac{r}{r_0})^6 + c_2 \exp\left(-\frac{c_3 r}{r_0}\right)} \ldots 4.12 \]
Here $c_1, c_2$ and $c_3$ are frequently taken to be universal constants, $E$ is an energy parameter, and $r$ is the interatomic distance, $r^*$ is the distance at which the function exhibits a minimum and is defined as the sum of the Van der Waals radii of the two interacting atoms. The potential-energy function 4.2 is found to work well for inert gases and for interactions between small molecules, such as $N_2$. In molecular-mechanics applications, it is assumed that nonbonded interactions can be summed over pairs of atoms and, furthermore, that the interaction is independent of the details of the molecular environment. The potential-energy functions employed for nonbonded interactions are derived from studies of intermolecular interactions.

The choice of the value of $E$ in the Hill equation has been the subject of some discussion. Hill originally took it to be

$$e^{-(e_k e_j)^2} \ldots 4.13$$

where $E_k$ is an atomic parameter for the $k$th atom and is sometimes referred to as the "hardness" of the atom.

The sixth term in the force-field expression (4.2) is the electrostatic interaction $V_e$. This is usually represented by an expression of the form

$$V_e = \sum_{all pairs} \frac{q_i q_j}{r_{ij}} \ldots 4.14$$

in which the charges $q_i$ are treated as variable parameters. The electrostatic term (4.14) is usually included in the force-field whenever there are heteroatoms present.

The term $V_{hb}$ arises from hydrogen bonding and is frequently neglected in organic chemistry, but is of utmost importance in the conformational analysis of certain types of macromolecular systems, such as biopolymers. Various empirical representations of $V_{hb}$
have been proposed, mainly based on fitting fixed hydrogen bond lengths, as observed in crystals, and the energy, from thermodynamic data, but also based on semiempirical quantum-chemical calculations, such as CNDO/2.

Finally, we come to the terms appearing in parentheses in equation (4.2). These terms arise from the fact that geminal interactions contribute to the steric energy. For example, when a bond angle is compressed the two associated bond lengths will become longer. One method for including these effects is the so-called Urey-Bradley force field, which has the form

\[ V_{1,3} = \frac{1}{2} \sum_{\text{all,3 interactions}} F(r-r_0)^2 + \sum_{\text{all,2 interactions}} F'(r-r_0) \ldots 4.15 \]

Here \( F, F' \) and \( r_0 \) are adjustable parameters and \( r \) is the distance between the geminal atoms. Alternatively, we can introduce cross-terms of the form

\[ V_{ct-Hxj} = F_0(r_j-r_0)(\theta_j-\theta_0) \ldots 4.16 \]

in which \( F, r_0 \) and \( \theta_0 \) are adjustable parameters. Such terms may account, for example, for the interaction between a CH stretch and a HCH angle bend.

It is important not to lose sight of the fact that the functions employed in the force-field calculations are empirical. They are chosen because they are easy to handle numerically and because they are reasonably transferable from system to system. It must be realised however that different parameters may give equally good descriptions of the properties of a particular molecular system. The parameters are most often determined by trial and error, although somewhat more systematic approaches have been advocated.
4.3 Minimisation of the Energy

The approach described for constructing the force-field of a given molecular system enables one to calculate the energy of that system for any arrangement of the component atoms that does not vary too significantly from the reference conformation. Frequently, it is necessary to minimise the energy of the system with respect to the internal degrees of freedom. Thus we obtain an estimate of the equilibrium structure of the molecule.

The computational problem to be solved may be written as

\[ V_{\text{minimum}} = \text{Min}[V(x)] \quad 4.17 \]

where \( V \) is the total force-field defined in eqn (4.2) and the vector \( x \) represents the parameters defining the structure of the molecule, that is, either the bond lengths, bond angles, torsional angles etc., or the cartesian coordinates of each of the component atoms.

In order to tackle the minimisation problem (4.17), it is useful to define the gradient vector \( g \) with elements

\[ g_i = \delta V(x) / \delta x_i \quad 4.18 \]

and the matrix of second derivatives, the Hessian matrix,

\[ G_{ij} = \delta^2 V(x) / \delta x_i \delta x_j \quad 4.19 \]

The gradient vector and the Hessian matrix may be evaluated analytically or they may be determined by finite-difference methods. When equation (4.17) has been solved and the minimum-energy conformation found, the gradient vector will be equal to the null vector and the Hessian matrix will be positive definite.

There are basically two efficient classes of methods for performing these
1 Conjugate-direction methods.
2 Newton methods, which can be further divided into
   a) Restricted-step methods.
   b) Quasi-Newton methods.

The Simplex method will also be described as it is a very simple method which has
an advantage of not requiring the use of any derivatives of the potential energy equation,
but has the serious disadvantage of being extremely slow.

4.3.1 The Simplex Method

A Simplex is a figure that has one more vertex than the dimension of space in
which it exists. The procedure of minimisation is shown in fig 4.9 from a much simplified
case of two parameters; x1 and x2.
For two parameters, there are three vertices that can be ranked as to response as best (B), second best (S) and worst (W). The centroid (C) of all points but the worst is calculated as the sum of coordinates (a or b) of the points:

\[
C_x = \frac{1}{NV-1} \sum_{i=1}^{NV-1} a_i; \quad C_y = \frac{1}{NV-1} \sum_{i=1}^{NV-1} b_i;
\]

The worst point is then reflected through the centroid to produce the reflected point (1). For each coordinate, we calculate the reflected value:

\[
\text{reflected coordinate} = C + F(C - W)
\]

(F=1 at this stage). The energy is then calculated at the reflected point. Calling this energy response \( R_1 \), the procedure below is followed:

If the reflected point is better than the best (\( R_1 < B \)), then the simplex is expanded with \( F=2 \) (point 2 of fig 4.9). The energy at this point is calculated (\( R_2 \)). If this is better than the reflected point, (\( R_2 < R_1 \)), the expanded point is accepted in place of the worst point, and the new simplex will be (B, S, 2). If not, the reflected point is accepted in place of the worst and the new simplex is (B, S, 1).

If the reflected point is not as good as the best (\( R_1 > B \)), but is better than the worst (\( R_1 < W \)), then the reflected point is accepted in place of the worst point and the new simplex is (B, S, 1).

If the reflected response is worst of all (\( R_1 < W \)), then a contracted point (3, on fig 4.9) is calculated using \( F=-0.5 \). If this response (\( R_3 \)) is better than the worst (\( R_3 < W \)), it is accepted in place of the worst and the new simplex is (B, S, 3). If not, the original simplex is contracted towards the best point.

After the new simplex is defined (by one of the criterion mentioned above), the procedure is repeated.
4.3.2 Conjugate-Direction Methods

Consider first the method of steepest descent, which is illustrated in fig 4.10 for a function of two variables.

![Diagram of steepest descent](image)

**Fig 4.10 The method of steepest descent**

The gradient $g(x_1, x_2)$ at the point $x=(x_1, x_2)$ defines the direction along which the function increases most rapidly. In the method of steepest descent, a search is made for the minimum value of the function in the direction $-g$. This process is an iterative one, the $(k+1)$th iterate being given by

$$x^{k+1} = x^k + ag$$

where the value of the scalar $a$ is adjusted to minimise $V(x)$ in the direction $-g$. The convergence properties of this procedure are often found to be oscillatory. This characteristic is caused by the fact that no account is taken of the quadratic components of the force-field $V(x)$. 

43
In order to obtain a superior method, we consider the form

\[ V(x+\delta) = V(x) + g^T\delta + \frac{1}{2} \delta^T G\delta \ldots 4.21 \]

in which \( G \) is the Hessian matrix (4.19). We require a method that will minimise the quadratic function (4.21) in a finite number of steps and, more importantly, a method that will also work for a function that is not actually quadratic. Thus we are lead to the method of conjugate directions.

The method of conjugate directions can be understood as follows. Let the set of \( n \) vectors \( d_1, d_2, \ldots, d_n \) be defined such that

\[ d_i^T G d_j = 0 \text{ if } i \neq j \ldots 4.22 \]

with respect to the Hessian matrix \( G \). The vectors \( d_i \) are said to be conjugate with respect to \( G \). Now these conjugate directions have the useful property that if one searches along \( d_1, d_2, \ldots, d_n \) in turn, then, for a quadratic function, the minimum value of the energy function will be reached. If the function is not quadratic, as is the case in fact, then further steps may be required.

The method of conjugate directions is illustrated in fig 4.11 for a function of two variables.
4.3.2 Newton Methods

The Newton methods differ from those discussed above in that they attempt to reach the minimum value of the energy function $V(x)$ directly rather than by means of a sequence of steps. To see how they work, consider the Taylor expansion for the gradient of $V(x)$, namely

$$g(x+\delta) = g(x) + G\delta + O(\delta^2) ... 4.23$$

At the minimum we have, for small delta

$$g(x+\delta) = 0 ... 4.24$$
Combining equations 4.23 and 4.24 and ignoring terms of order \( \delta^2 \) and higher, we obtain the basic equation of the Newton methods in the form

\[
G \delta = -g \quad 4.25
\]

For a quadratic function, the Newton method arrives at the minimum value of the energy function immediately, while for functions that are not quadratic (the usual case in molecular mechanics calculations) the iteration process of the Newton method is given by.

\[
\delta = -G^{-1}g \quad 4.26
\]

and

\[
x^{i+1} = x^i + \delta \quad 4.27
\]

The Newton method, in its simple form presented above, has second-order convergence properties near the minimum, since the Hessian matrix is usually positive definite in this region. However, the simple Newton procedure does not encounter problems if the initial point \( x_0 \) is far from the conformation corresponding to the minimum energy. A further disadvantage of the Newton method is that it requires evaluation of the Hessian matrix. Restricted-step methods and quasi-Newton methods, which are now described, attempt to avoid these problems while retaining the general spirit of the Newton approach.

4.3.3 Restricted-Step Methods

The simple Newton iteration given above fails if at some point the Hessian matrix \( G \) is not positive definite. This problem may be overcome by employing a restricted-step method in which the basic equation of the Newton method, equation 4.25, is modified to read
where \( I \) is the unit matrix of dimension \( m \) by \( m \) (\( m \) being the number of parameters to be varied) and \( \lambda \) is a scalar. The quantity \( \lambda \) is a non-negative number, which is chosen so as make the matrix \((G+\lambda I)\) positive definite. This ensures that the energy-function values must decrease after each iteration. The method restricts the step length by interpolating between the method of steepest descent and the simple Newton method. If \( \lambda \) is large then equation 4.28 is effectively the method of steepest descent, while if \( \lambda \) is zero then it is clearly the straight-forward Newton procedure. Different implementations of the restricted-step Newton procedure are distinguished mainly by their strategies for choosing an appropriate value of \( \lambda \).

### 4.3.4 Quasi-Newton Methods

Direct application of the Newton method, eqn 4.25 requires evaluation of the Hessian matrix \( G \) in each iteration. This can frequently involve an unacceptably large amount of computation.

Quasi-Newton methods attempt to avoid the explicit construction of the Hessian matrix. The majority of the quasi-Newton algorithms require function and gradient evaluations only.

Let us introduce a matrix \( H^k \), which will be our approximation of \( G^{-1} \) during the \( k^{th} \) iteration, \( H^k \) is updated by taking into account information gained during that iteration.

If

\[
\gamma = g(x+\delta) - g(x) \quad 4.29
\]

then, to first order, we have

We therefore modify \( H^k \) so that \( H^{k+1} \) satisfies
A number of schemes have been proposed for updating \( H \). The Davidson-Fletcher-Powell scheme used the expression

\[
H^{k+1} = H^k + \frac{\delta \delta^T (H \gamma)(\gamma^T H)}{\gamma^T \gamma}.
\]

C.G. Broyden\textsuperscript{17} R.Fletcher\textsuperscript{18} and P.F.Shanno\textsuperscript{19} have suggested schemes that do not require gradient evaluations in each iteration.
CHAPTER FIVE

DEVELOPMENT OF THE MOLECULAR MECHANICS PROGRAM MOLMECH
5.1 Problems with the molecular mechanics-technique and Inorganic compounds

The method of molecular mechanics has been used for many years with great success in organic structural chemistry. Parameters derived for particular atom types can be used for a vast range of different structures.

When molecular mechanics calculations are used for inorganic structural chemistry, however, a number of important problems are encountered. These are:

1. Various types of bonding -

Bonding in inorganic compounds may be ionic, covalent, metallic, or a mixture, which presents us with the problem of how best to describe these interactions within the force-field equation.

2. Partial charges on metals and complexes are difficult to assign -

It is often not possible to use quantum mechanical methods reliably to assign partial charges. Thus it is not possible to calculate the ionic contribution to the force-field unless formal charges are used which assumes full ionic character.

3. There is a vast number of inorganic compounds with variable geometry, coordination numbers with different oxidation states and metal sizes -

This leads to a large amount of parameterisation needed. New parameters must be derived for every system studied as they are not strictly transferable from structure to structure as is the case for organic systems.

4. Problems dealing with some 5-coordinated compounds lead to unacceptable distortions -

This has been found to be the case particularly with five coordinated metal centres embedded in rigid backbone structures such as metalloproteins.\(^{20-22}\)
5.2 The program MOLMECH

Molecular mechanics code for dealing with inorganic compounds, does not generally exist. This left us with two options, firstly to modify an existing molecular mechanics program such as MM2, so that it could deal with inorganic structures, or to design novel code that would be more flexible and amenable to modification to that of an existing system.

In order to gain experience in the development of this type of code, a molecular mechanics program was written in Microsoft QuickBasic on a 386-based Hewlett Packard Vectra PC. The PC environment, however, was found to be limited due to speed reasons; for reasonably large molecules, the minimisation time was found to be unacceptable. The choice between optimising the code or moving it to a faster platform was made in favour of converting the code to FORTRAN and implementing it on the MicroVaxII, where calculations could be performed in an acceptable time. Volume 2 provides a MOLMECH program listing.

5.2.1 Program Description

The program consists of 2600 lines of Fortran code. It currently supports three minimisation methods (Chapter 4).

1 Simplex method.
2 Steepest Descents.
3 Conjugate Gradients.

It has a torsional driver, which forces torsional movement and helps to drive the minimiser from obvious local minima.

An empirical metal potential function was included in the force-field to help overcome problems with five coordinate distortion\textsuperscript{20-22}. The program can be used for nucleic acids and proteins/peptides with particular reference to metalloproteins using force-field parameters derived by Weiner et al\textsuperscript{23-24}.
5.2.2 The Force-Field

The force-field used was as follows:

\[ V = \frac{1}{2} \sum_{\text{bonds}} k_r (r - r_0)^2 + \frac{1}{2} \sum_{\text{angles}} k_\theta (\theta - \theta_0)^2 \]

\[ + \sum_{\text{dihedrals}} \frac{V_n}{2} [1 + \cos(\Phi) - \tau] \]

\[ + \sum_{i < j} \left[ \left( \frac{A_{ij}}{R_{ij}^3} \right) - \left( \frac{B_{ij}}{R_{ij}^6} \right) + \left( \frac{C_{ij}Q_i Q_j}{eR_{ij}} \right) \right] \]

\( k_r \)......bond stretch force constant
\( r \)......bond length
\( r_0 \)......reference bond length
\( K_{\theta} \)......angle bending force constant
\( \theta \)......bond angle
\( \theta_0 \)......reference bond angle
\( n \)......periodicity of torsional motion
\( \tau \)......180° if the eclipsed conformation favourable
\( \tau \)......0° if the staggered conformation favourable
\( A \)......\( E_{ij}^* (2r_{ij}^*)^{12} \)
\( B \)......\( E_{ij}^* (2r_{ij}^*)^6 \)
\( r^* \)......distance at which the nonbonded function exhibits a minimum and is defined as the sum of the Van der Waals radii of the two interacting atoms.
\( E \)......an energy parameter.

5.2.3 Empirical Metal Potential Function

Vedani and Dunitz\textsuperscript{20-22} have reported the use of a potential function for metal centres, in order to remove the need to assign partial atomic charges in calculations.
\[ E_{\text{net}} = \sum_{\text{bonds}} \left( A''_{\mu \nu} \frac{C''_{\mu \nu}}{r_{\mu \nu}^{10}} \right) \prod_{\text{angles}} \cos^k \theta_{\mu \nu \lambda} \cos \theta_{\text{obj}} \]

This potential function is discussed fully in chapter 8

5.2.4 Derivatives

The following derivatives were used in the program:

Bond stretching

\[ V = k_s(r-r_0)^2 \]

\[ r^{-\left((x-x_i)^2+(y-y_i)^2+(z-z_i)^2\right)^{\frac{1}{2}}} \]

\[ \frac{\delta V}{\delta x} = 2k_s(r-r_0)x_i \left((x-x_i)^2+(y-y_i)^2+(z-z_i)^2\right)^{-\frac{1}{2}}2(x-x_i) \frac{1}{r} \]

\[ \frac{\delta V}{\delta y} = 2k_s(r-r_0)y_i \left((x-x_i)^2+(y-y_i)^2+(z-z_i)^2\right)^{-\frac{1}{2}}2(y-y_i) \frac{1}{r} \]

\[ \frac{\delta V}{\delta z} = 2k_s(r-r_0)z_i \left((x-x_i)^2+(y-y_i)^2+(z-z_i)^2\right)^{-\frac{1}{2}}2(z-z_i) \frac{1}{r} \]

\[ \frac{\delta V}{\delta x_1} = \frac{\delta V}{\delta y_1} = \frac{\delta V}{\delta z_1} = \frac{\delta V}{\delta z} = \frac{\delta V}{\delta y} = \frac{\delta V}{\delta x} = \frac{\delta V}{\delta x_1} = \frac{\delta V}{\delta y_1} = \frac{\delta V}{\delta z_1} = \frac{\delta V}{\delta z} = \frac{\delta V}{\delta y} = \frac{\delta V}{\delta x} = \frac{\delta V}{\delta x_1} \]

52
Angle bending

\[ V = k_\theta (\theta - \theta_\circ)^2 \]

\[ \frac{\delta V}{\delta x_1} = 2k_\theta (\theta - \theta_\circ) \frac{\delta (\theta - \theta_\circ)}{\delta x_1} \]

\[ \frac{\delta V}{\delta x_1} = 2k(\theta - \theta_\circ) \cos^{-1} \left( \frac{(A, B)}{\|A\| \|B\|} \right) \]

where

\[ A = (x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2 \]

\[ B = (x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2 \]

\[ A.B = (x_1 - x_2)(x_3 - x_2) + (y_1 - y_2)(y_3 - y_2) + (z_1 - z_2)(z_3 - z_2) \]

using the function of a function rule.

\[ y = g(u) \text{ where } u = f(x) \]

\[ y = \cos^{-1}(u) \text{ where } u = \frac{(A, B)}{\|A\| \|B\|} \]

\[ \frac{dy}{du} = \frac{-1}{(1-u^2)^{1/2}} \quad \frac{du}{dx} = \frac{(A, B)}{\|A\| \|B\|} \]

\[ \frac{\delta y}{\delta x} = \frac{-1}{(1-x^2)^{1/2}} \quad \frac{d}{dx_1} \frac{(A, B)}{\|A\| \|B\|} \]

53
Therefore

\[
\frac{\delta V}{\delta x_1} = -2k_0(\theta - \theta_0) \cdot \frac{1}{1 - \left(\frac{A \cdot B}{|A||B|}\right)^2} \cdot \frac{d(A \cdot B)}{dx} \cdot \frac{d}{dx}\left|\frac{A \cdot B}{|A||B|}\right|
\]

Using the quotient rule:

\[
y = \frac{u}{v}, \quad \frac{dy}{dx} = \frac{\frac{du}{dx} - \frac{uv}{dx^2}}{v^2}
\]

\[
u = A \cdot B, \quad v = |A||B|
\]

Therefore:

\[
\frac{\delta V}{\delta x_1} = -2k_0(\theta - \theta_0) \cdot \frac{1}{1 - \left(\frac{A \cdot B}{|A||B|}\right)^2} \cdot \frac{A \cdot B}{|A||B|} \cdot \frac{d}{dx}|A||B| \cdot \frac{d}{dx}\left(\frac{A \cdot B}{|A||B|}\right)
\]
\[
\frac{\delta V}{\delta x_1} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(x_3 - x_2) - (A.B)}{(A||B|^3) \cdot B.(x_1 - x_2)} \right)
\]

\[
\frac{\delta V}{\delta y_1} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(y_3 - y_2) - (A.B)}{(A||B|^3) \cdot B.(y_1 - y_2)} \right)
\]

\[
\frac{\delta V}{\delta z_1} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(z_3 - z_2) - (A.B)}{(A||B|^3) \cdot B.(z_1 - z_2)} \right)
\]

\[
\frac{\delta V}{\delta x_3} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(x_1 - x_2) - (A.B)}{(A||B|^3) \cdot B.(x_2 - x_2)} \right)
\]

\[
\frac{\delta V}{\delta y_3} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(y_1 - y_2) - (A.B)}{(A||B|^3) \cdot B.(y_2 - y_2)} \right)
\]

\[
\frac{\delta V}{\delta z_3} - 2k_0(\theta - \theta_0) - \frac{1}{1 - \frac{A.B}{|A||B|}^2} \left( \frac{1}{|A||B|} \cdot \frac{(z_1 - z_2) - (A.B)}{(A||B|^3) \cdot B.(z_2 - z_2)} \right)
\]

\[
\frac{\delta V}{\delta x_2} = \left( \frac{\delta V}{\delta x_1} + \frac{\delta V}{\delta x_3} \right)
\]

\[
\frac{\delta V}{\delta y_2} = \left( \frac{\delta V}{\delta y_1} + \frac{\delta V}{\delta y_3} \right)
\]

\[
\frac{\delta V}{\delta z_2} = \left( \frac{\delta V}{\delta z_1} + \frac{\delta V}{\delta z_3} \right)
\]

**TORSION ANGLE TERM**

\[nfold=1\]

\[V = V_a + V_n \cos \phi\]
Using the Quotient rule

\[ \frac{\delta V}{\delta x_1} - \frac{A.B}{|A||B|} \delta x_1 \]

Therefore:

\[ \frac{\delta V}{\delta x_1} = \frac{\delta (A.B)}{\delta x_1} (\frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|}) \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]

\[ \frac{\delta V}{\delta x_1} = \frac{|A||B|}{A.B} \delta x_1 - \frac{A.B}{|A||B|} \delta x_1 \]
similarly for

\[ \frac{\delta V}{\delta y_1}, \frac{\delta V}{\delta z_1}, \frac{\delta V}{\delta x_2}, \frac{\delta V}{\delta y_2}, \frac{\delta V}{\delta z_2}, \frac{\delta V}{\delta x_3}, \frac{\delta V}{\delta y_3}, \frac{\delta V}{\delta z_3}, \frac{\delta V}{\delta x_4}, \frac{\delta V}{\delta y_4}, \frac{\delta V}{\delta z_4} \]

for \( n \text{fold}=2 \)

\[ V = V_n + V_n \cos 2\phi \]

\[ \cos 2\phi = 2\cos^2\phi - 1 \]

Therefore

\[ V = V_n + V_n (2\cos^2\phi - 1) \]

\[ \frac{\delta V}{\delta x_1} = 4\cos\phi V_n (\ldots \text{as for 1 fold}) - 4\cos\phi V_n (\ldots \text{as for 1 fold}) \]

for \( n \text{fold} = 3 \)

\[ \cos 3\phi = \cos(2\phi + \phi) \]

\[ = \cos 2\phi \cos \phi - \sin 2\phi \sin \phi \]

\[ = (2\cos^2\phi - 1)\cos \phi - (2\sin \phi \cos \phi)\sin \phi \]

\[ = 2\cos^3\phi - \cos \phi - 2\sin^2\phi \cos \phi \]

\[ = 2\cos^3\phi - \cos \phi - 2(1 - \cos^2\phi) \]

\[ = 2\cos^3\phi - \cos \phi - (2 - 2\cos^2\phi) \cos \phi \]
$$=2\cos^4\phi - \cos^2\phi - 2\cos\phi + 2\cos^2\phi$$

$$=4\cos^3\phi - 3\cos\phi$$

$$V - V_n + V_n(4\cos^3\phi - 3\cos\phi)$$

$$\frac{\delta V}{\delta x_1} = (12\cos^2\phi - 3)V_n (\text{as for 1 fold}) - (12\cos^2\phi - 3)V_n (\text{as for 1 fold})$$

For $n \text{fold} = 4$

$$V - V_n + V_n \cos^4\phi$$

$$\cos^4\phi = \cos(3\phi + \phi)$$

$$= \cos^3\phi \cos\phi - \sin^3\phi \sin\phi$$

$$= \cos(2\phi + \phi) \cos\phi - \sin(2\phi + \phi) \sin\phi$$

$$= (4\cos^3\phi - 3\cos\phi) \cos\phi - (3\sin^3\phi - 4\sin^4\phi) \sin\phi$$

$$= 4\cos^4\phi - 3\cos^2\phi - (3\sin^2\phi - 4\sin^4\phi)$$

$$= 4\cos^4\phi - 3\cos^2\phi - 3\sin^2\phi + 4\sin^4\phi$$

$$= 4\cos^4\phi - 3\cos^2\phi - 3(1 - \cos^2\phi) + (4\sin^2\phi \sin^2\phi)$$

$$= 4\cos^4\phi - 3\cos^2\phi - 3 + 3\cos^2\phi + 4((1 - \cos^2\phi)(1 - \cos^2\phi))$$

$$= 4\cos^4\phi - 3 + 4(1 - \cos^2\phi - \cos^2\phi + \cos^4\phi)$$

$$= 4\cos^4\phi - 3 + 4(1 - 2\cos^2\phi + \cos^4\phi)$$
\[V = V_n + V_s(8\cos^4\phi - 8\cos^2\phi + 1)\]

\[\frac{\delta V}{\delta x_1} = (32\cos^3\phi - 16\cos\phi)V_n(\ldots \text{as for } n\text{fold} = 1) - (32\cos^3\phi - 16\cos\phi)V_s(\ldots \text{as for } n\text{fold} = 1)\]

Nonbonded interaction term

\[V = \frac{B^{12}}{r^{12}} - \frac{A^6}{r^6}\]

first part:

\[\frac{\delta V}{\delta x_1} = B^{12}(r-r_0)^{-12}\]

\[-B^{12} \cdot 12(r-r_0)^{-13} \frac{\delta r}{\delta x_1}\]

\[-B^{12} \cdot 12(r-r_0)^{-13} \frac{1}{2}(x-x_i)^2 - (y-y_i)^2 + (z-z_i)^2)^{-\frac{1}{2}} 2(x-x_i)\]

\[-\frac{12B^{12}}{r^{13}}(x-x_i)\]

\[-\frac{12B^{12}}{r^{14}}(x-x_i)\]
similarly for

\[ \frac{\delta V}{\delta y_1}, \frac{\delta V}{\delta z_1}, \text{etc.} \]

second part:

\[ \frac{\delta V}{\delta x_1} - A^6 r^{-6} \]

\[ = -6A^6 r^{-7} \frac{\delta r}{\delta x_1} \]

\[ = -6A^6 r^{-7} r^{-1}(x-x_1) \]

\[ = -6A^6 \frac{x-x_1}{r^8} \]

\[ = \left(6A^6 r^8 - \frac{12B^{12}}{r^{14}}\right)(x-x_1) \]

\[ \frac{\delta V}{\delta x_1} = -\frac{12B^{12}}{r^{14}}(x-x_1) - \frac{6A^6}{r^8}(x-x_1) \]

\[ \frac{\delta V}{\delta y_1} = -\frac{12B^{12}}{r^{14}}(y-y_1) - \frac{6A^6}{r^8}(y-y_1) \]

\[ \frac{\delta V}{\delta z_1} = -\frac{12B^{12}}{r^{14}}(z-z_1) - \frac{6A^6}{r^8}(z-z_1) \]

Derivatives for the empirical metal potential function

where \( A'' = -5.0 E_0 r_0^{12} \), \( C'' = -6.0 E_0 r_0^{10} \)
\[ \text{Etot} = \sum \left( \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \right) \prod \cos^2(\theta - \theta_0) \]

(E_0 is the well depth, \( r_0 \) is the ideal bond length)

\[ d \frac{\cos^2(\theta_{\text{int}} - \theta_0)}{dx_1} = -ksin^{k-1}(\theta_{\text{int}} - \theta_0) \cdot \frac{d(\theta_{\text{int}} - \theta_0)}{dx_1} \]

\[ = -ksin^{k-1}(\theta_{\text{int}} - \theta_0) \cdot \frac{\cos^{-1}(A.B)}{dx_1} \]

\[ = -ksin^{k-1}(\theta_{\text{int}} - \theta_0) \cdot \frac{-1}{\left(1 - (A.B)^2/\left[(A.B)^2 + (A.B)^2 \cdot B.(x_1-x_2)\right]\right)^{1/2}} \cdot \frac{(x_2-x_1)}{\left[(A.B)^2 + (A.B)^2 \cdot B.(x_1-x_2)\right]^{1/2}} \]

\[ \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \]

\[ dx_1 \]

\[ = -12A''r^{-13} \cdot \frac{dr}{dx_1} - (-10)C''r^{-11} \cdot \frac{dr}{dx_1} \]

\[ = \frac{12A''(x-x_1)}{r^{14}} \cdot \frac{10C''(x-x_1)}{r^{12}} \]

\[ - (x-x_1) \left( \frac{12A''}{r^{14}} - \frac{10C''}{R^{12}} \right) \]

Using the product rule:

\[ \frac{d(u \cdot v)}{dx} = u \frac{dv}{dx} + v \frac{du}{dx} \]
\[
\frac{dE_{\text{cut}}}{dx_1} = \cos(\theta_{\text{int}} - \theta_0) \cdot (x-x_1) \left( \frac{12A''}{r^{14}} - \frac{10C''}{r^{12}} \right) + \frac{A''}{r^{14}} - C'' \cdot r^{10} \cdot -k \ \sin^{k-1}(\theta_{\text{int}} - \theta_0)
\]

\[
\cdot \frac{-1}{\left( 1 - \frac{(A \cdot B)}{\left| A \right| \left| B \right|} \right)^2} \cdot \frac{(x_2-x_2)}{\left( \left| A \right| \left| B \right| \right)^2} - \frac{A \cdot B}{\left( \left| A \right| \left| B \right| \right)^3} \cdot B \cdot (x_1-x_2)
\]
5.3 Program verification

The Program was tested by comparing test structure geometries after minimisation with those minimised by the proven molecular mechanics program MM2. The geometries were found to compare well.

Methane

<table>
<thead>
<tr>
<th>MM2</th>
<th></th>
<th>MOLMECH</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rms</td>
<td>0.01</td>
<td>Rms = 0.04</td>
<td></td>
</tr>
<tr>
<td>Total energy</td>
<td>0.00 Kcal/mol</td>
<td>Total energy = 0.02 Kcal/mol</td>
<td></td>
</tr>
<tr>
<td>All angles</td>
<td>109.5°</td>
<td>All angles 109.5°</td>
<td></td>
</tr>
<tr>
<td>All bonds</td>
<td>1.10 Å</td>
<td>All bonds 1.10 Å</td>
<td></td>
</tr>
</tbody>
</table>

Ethane

<table>
<thead>
<tr>
<th>MM2</th>
<th></th>
<th>MOLMECH</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rms</td>
<td>0.02</td>
<td>Rms = 0.06</td>
<td></td>
</tr>
<tr>
<td>Total energy</td>
<td>-0.36 Kcal/mol</td>
<td>Total energy = -0.01 Kcal/mol</td>
<td></td>
</tr>
<tr>
<td>All angles</td>
<td>109.5°</td>
<td>All angles 109.5°</td>
<td></td>
</tr>
<tr>
<td>bonds</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-C</td>
<td>1.52 Å</td>
<td>C-C</td>
<td>1.52 Å</td>
</tr>
<tr>
<td>C-H</td>
<td>all 1.10 Å</td>
<td>C-H</td>
<td>all 1.10 Å</td>
</tr>
</tbody>
</table>

Isobutane

<table>
<thead>
<tr>
<th>MM2</th>
<th></th>
<th>MOLMECH</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rms</td>
<td>0.08</td>
<td>Rms = 0.09</td>
<td></td>
</tr>
<tr>
<td>Total energy</td>
<td>-1.87 Kcal/mol</td>
<td>Total energy = -0.90 Kcal/mol</td>
<td></td>
</tr>
<tr>
<td>See fig 5.3.1 for angles</td>
<td></td>
<td>See fig 5.3.2 for angles</td>
<td></td>
</tr>
<tr>
<td>bonds</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-C</td>
<td>all 1.52 Å</td>
<td>C-C</td>
<td>all 1.52 Å</td>
</tr>
<tr>
<td>C-H</td>
<td>all 1.10 Å</td>
<td>C-H</td>
<td>all 1.10 Å</td>
</tr>
</tbody>
</table>
sample torsions MM2

mm2 torsion

-59.0

-179.1

59.9

60.8

-179.8

sample torsions MOLMECH

mm2 torsion

-59.2

-179.4

60.0

60.7

-179.9
Benzene

**MM2**
- Rms = 0.07
- Total energy = 0.35
- All angles 120°
- bonds
  - C-C
  - 1.40 Å
- All torsions 0°

**MOLMECH**
- Rms = 0.04
- Total energy = 0.98
- All angles 120°
- bonds
  - C-C
  - 1.08 Å
- All torsions 0°

Methyl-benzene

**MM2**
- Total energy = -0.07
- Rms = 0.07
- All angles MM2

**MOLMECH**
- Total energy = 0.12
- Rms = 0.10
- All angles MOLMECH
Propan-1-ol

*MM2*

Total energy = 0.20 Kcal/mol
Rms = 0.07

*bonds*
C-H
all 1.10 Å
C-C
all 1.52 Å
C-O
1.47 Å
O-H
0.95 Å

*angles MM2*

*MOLMECH*

Total energy = 0.40 Kcal/mol
Rms = 0.10

*bonds*
C-H
all 1.10 Å
C-C
all 1.52 Å
C-O
1.47 Å
O-H
0.95 Å

*angles MOLMECH*
Conclusion
MOLMECH reproduces geometries consistently with that of MM2. The incorporation of empirical metal potential functions into the force-field equations enables it to be used for Inorganic structural investigations. Firstly it was applied to the simpler case of isolated inorganic molecules. Then to unit-cell calculations and finally to investigations of metalloproteins. These investigations are discussed in the next chapters.
CHAPTER SIX

APPLICATION OF MOLMECH TO ISOLATED MOLECULES
A Molecular Mechanics Evaluation of the Influence Of Steric and Electronic Factors on the Re-Re Bond Length in a Series of $\text{Re}_2(\text{CO})_{10-n}(\text{CNR})_n (n=1-4)$ Structures

6.1 Introduction

An X-ray crystallographic study of the structures of a series of di-rhenium carbonyl - isonitrile complexes, $\text{Re}_2(\text{CO})_{10-n}(\text{CNR})_n (n=1, \text{R}=\text{Bu}^t; n=2,4, \text{R}=\text{C}_6\text{H}_3\text{Me}_2-2,6; n=3, \text{R}=\text{Me})$ had been carried out. One of the aims of this study was to investigate the effect of increasing isonitrile substitution on the metal-metal bond length. Since isonitriles are weaker π-acceptors than carbonyls, increasing isonitrile substitution would be expected to result in a weakening and hence a lengthening of the metal-metal bond. The Re-Re bond length data from the crystallographic study is given in table 6.1. As can be seen, there is a slight lengthening of the Re-Re bond with increasing $n$, but this only becomes significant at the $n=4$ level.

A lengthening of the metal-metal bond length on increasing isonitrile substitution could also be expected on purely steric grounds in order to reduce repulsion between bulky isonitrile groups, especially in the tetra substituted case. In order to quantify the relative importance of the various steric and electronic factors in these structures, molecular mechanics calculations were performed, using MOLMECH, on the complexes (I)-(IV) in table 6.1.

Table 6.1 Observed and calculated Re-Re bond lengths for the $\text{Re}_2(\text{CO})_{10-n}(\text{CNR})_n (n=0-4)$ structures.

<table>
<thead>
<tr>
<th>Structure</th>
<th>Crystal Structure</th>
<th>Calculated</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Re}<em>2(\text{CO})</em>{10}$</td>
<td>[30]</td>
<td>3.041</td>
</tr>
<tr>
<td>$\text{Re}_2(\text{CO})_9(\text{CNBu})^t$ (I)</td>
<td>[25]</td>
<td>3.042</td>
</tr>
<tr>
<td>$\text{Re}_2(\text{CO})_8(\text{CNC}_6\text{H}_3\text{Me}_2-2,6)_2$ (II)</td>
<td>[26]</td>
<td>3.044</td>
</tr>
<tr>
<td>$\text{Re}_2(\text{CO})_7(\text{CNMe})_3$ (III)</td>
<td>[26]</td>
<td>3.047</td>
</tr>
<tr>
<td>$\text{Re}_2(\text{CO})_6(\text{CNC}_6\text{H}_3\text{Me}_2-2,6)_4$ (IV)</td>
<td>[26]</td>
<td>3.050</td>
</tr>
</tbody>
</table>
6.2 Method

The sources of the force-field parameters are discussed below:

(a) Non-bonded parameters

The values of $r^*$ and $E^*$ for the C, O and N atoms were obtained from Kollman\textsuperscript{22-24}. For Re, the $r^*$ value was taken as the shortest Re-Re distance in Rhenium metal\textsuperscript{29}. The values of $r^*$ and $E^*$ used in the calculations are given in Table 6.2. $r^*$ is the Van der Waals minimum and $E^*$ the Van der Waals depth (see fig 6.1).

(b) Bond length and bond angle parameters

The values for the equilibrium bond length $r_{eq}$ and bond angle $\theta_{eq}$ were obtained from the X-ray crystal structure of Re$_2$(CO)$_{10}$\textsuperscript{29}. For the parent compound, Re$_2$(CO)$_{10}$, the Re-Re equilibrium bond distance was adjusted until the observed structure of Re$_2$(CO)$_{10}$\textsuperscript{29} was reproduced ($k_r = 118$ kcal/mol Å$^2$). This value of $r_{eq}$(Re-Re) (3.041 Å) was then used in the subsequent calculations for the Re$_2$(CO)$_{10-n}$(CNR)$_n$ (n = 1-4) derivatives. The other required equilibrium bond lengths and angles were taken from the X-ray crystal structures of the complexes (I)-(IV)\textsuperscript{25,26}. The values of $r_{eq}$ and $\theta_{eq}$ used in the calculations are given in Tables 6.3 and 6.4 respectively.
(c) Force Constants.

The bond angle force constants, $k_r$ and $k_{\theta\theta\theta}$, were obtained from infrared \cite{31} and Raman \cite{32} spectroscopy studies on Re$_2$(CO)$_{10}$. The value used for the force constants for the Re-Re bond was 118 kcal/mol Å$^2$, obtained from the Raman study of Re$_2$(CO)$_{10}$ by Spiro et al.\cite{32}. The values of the force constants $k_r$ and $k_{\theta\theta\theta}$ used in the calculations are given in table 6.3 and 6.4 respectively.

The force-field parameters were established by an analysis of the parent Re$_2$(CO)$_{10}$ structure, the X-ray crystal structure of which has been reported\cite{32}. These force-field parameters were then kept constant for the subsequent calculations on the derived Re$_2$(CO)$_{10-n}$(CNR)$_n$ (n=1-4) structures.

Table 6.2 Values of the force-field parameters used in the calculations:

<table>
<thead>
<tr>
<th>Atom type</th>
<th>$r^*$ (Å)</th>
<th>$E^*$ (kcal/mol)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>1.850</td>
<td>0.120</td>
</tr>
<tr>
<td>C (CN)</td>
<td>2.025</td>
<td>0.120</td>
</tr>
<tr>
<td>O</td>
<td>1.600</td>
<td>0.200</td>
</tr>
<tr>
<td>N</td>
<td>1.750</td>
<td>0.160</td>
</tr>
<tr>
<td>Re</td>
<td>2.761</td>
<td>0.250</td>
</tr>
</tbody>
</table>

Table 6.3 Values for the bond force constants $k_r$ and the equilibrium bond lengths $r_{eq}$ used in the calculations.

<table>
<thead>
<tr>
<th>Bond type</th>
<th>$K_r$ (kcal/mol Å$^2$)</th>
<th>$r_{eq}$ (Å)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-O</td>
<td>570</td>
<td>1.130</td>
</tr>
<tr>
<td>C-Re</td>
<td>350</td>
<td>1.987</td>
</tr>
<tr>
<td>C-N (double)</td>
<td>570</td>
<td>1.130</td>
</tr>
<tr>
<td>(single)</td>
<td>367</td>
<td>1.450</td>
</tr>
<tr>
<td>C-C (single)</td>
<td>317</td>
<td>1.552</td>
</tr>
<tr>
<td>(aromatic)</td>
<td>512</td>
<td>1.388</td>
</tr>
<tr>
<td>Re-Re</td>
<td>118</td>
<td>3.041</td>
</tr>
</tbody>
</table>
Table 6.4 Values for the angular force constants $k_{\theta}$ and equilibrium bond angles $\theta_{eq}$ used in the calculations.

<table>
<thead>
<tr>
<th>Angle type</th>
<th>$k_{\theta}$ (Kcal/mol rad$^2$)</th>
<th>$\theta_{eq}$ (°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$^{eq}$OC-Re-C$^{eq}$</td>
<td>58.0</td>
<td>172.5</td>
</tr>
<tr>
<td>trans</td>
<td>58.0</td>
<td></td>
</tr>
<tr>
<td>cis</td>
<td>58.0</td>
<td></td>
</tr>
<tr>
<td>$^{\alpha}$OC-Re-CO$^{eq}$</td>
<td>58.0</td>
<td>93.7</td>
</tr>
<tr>
<td>$^{eq}$OC-Re-CN</td>
<td>58.0</td>
<td></td>
</tr>
<tr>
<td>trans</td>
<td>58.0</td>
<td>174.9</td>
</tr>
<tr>
<td>cis</td>
<td>58.0</td>
<td>89.9</td>
</tr>
<tr>
<td>$^{\alpha}$OC-Re-CN</td>
<td>58.0</td>
<td>92.1</td>
</tr>
<tr>
<td>NC-Re-CN</td>
<td>58.0</td>
<td>89.3</td>
</tr>
<tr>
<td>$^{eq}$OC-Re-Re</td>
<td>58.0</td>
<td>86.0</td>
</tr>
<tr>
<td>$^{\alpha}$OC-Re-Re</td>
<td>58.0</td>
<td>167.3</td>
</tr>
<tr>
<td>NC-Re-Re</td>
<td>58.0</td>
<td>87.5</td>
</tr>
<tr>
<td>Re-C-O$^{eq}$</td>
<td>72.0</td>
<td>178.0</td>
</tr>
<tr>
<td>Re-C-O$^{\alpha}$</td>
<td>72.0</td>
<td>177.5</td>
</tr>
<tr>
<td>Re-C-N</td>
<td>72.0</td>
<td>175.5</td>
</tr>
<tr>
<td>C-N-C</td>
<td>90.0</td>
<td>175.0</td>
</tr>
<tr>
<td>N-C-C (Bu$^i$)</td>
<td>80.0</td>
<td>108.7</td>
</tr>
<tr>
<td>C-C-C (Bu$^i$)</td>
<td>63.0</td>
<td>109.1</td>
</tr>
<tr>
<td>C-C-C (benzene)</td>
<td>85.0</td>
<td>120.0</td>
</tr>
<tr>
<td>N-C-C (benzene)</td>
<td>70.0</td>
<td>120.0</td>
</tr>
</tbody>
</table>

6.3 Results and discussion

An analysis of the results revealed a variation in the Re-Re bond length. The values of the bond angles and other bond lengths did not change significantly. Table 6.4 shows the calculated Re-Re bond lengths, compared to the crystal structure values.

The calculated results indicate a slight increase in the Re-Re bond length down the series $\text{Re}_2(\text{CO})_{10-n}(\text{CNR})_n$ ($n=1$-$4$), but this effect is not as large as might be expected from the crystal structure data. Hence the role of non-bonded interactions, or steric effects, which might be expected to increase the Re-Re bond length, is not the dominant factor in these structures. There seems to be little steric effect observed in these structures as the Re-Re-C(equatorial) angles are acute, i.e. no bending out of the CNR (or CO) ligands.$^{25,26}$
The difference between the crystallographic Re-Re bond distances and the calculated values can be ascribed to the electronic effects which are not adequately modelled by using the force-field for the parent \( \text{Re}_2\text{(CO)}_{10} \). In particular, replacing carbonyls by isonitriles, which are poorer pi-acceptors\(^{27,28}\), would be expected to cause a weakening, and hence a lengthening, of the Re-Re bond.

Hence electronic i.e. bonding interactions, rather than steric effects i.e. non-bonded interactions, would appear to be the dominant factor influencing the length of the Re-Re bond in the \( \text{Re}_2\text{(CO)}_{10-n}\text{(CNR)}_n \) structures. This conclusion is supported by the observation that the Re-Re bond length in \( \text{Re}_2\text{(CO)}_8\text{(PMe}_2\text{Ph})_2 \) has been shown by X-ray crystallography\(^{23}\) to be 3.044(1)Å cf. 3.041(1)Å for \( \text{Re}_2\text{(CO)}_{10} \). This indicates that the Re-Re bond length is almost invariant on substitution of carbonyls by the bulkier phosphine ligands.

**Possible treatment of pi back-bonding effects in molecular mechanics.**

It may be possible to approximate pi back-bonding effects in molecular mechanics by developing an empirical potential function for pi back-bonding in ligands. The function could reduce the bonding force constants (hence the simulated strength of the bond) in an amount related to the degree and strength of PI-back bonding present in the ligands. Back bonding parameters could be developed for different ligands that reflected the strength of pi back-bonding in a particular ligand.
CHAPTER SEVEN

APPLICATION OF MOLMECH
TO UNIT-CELLS
7.1 Introduction

In order to investigate the applicability of the molecular mechanics program to the modelling of crystal structures, calculations were carried out on some chlorochromate and chlorocuprate unit-cells. X-ray studies of propane-diammonium tetrachlorochromate(II), diethane-tri-ammonium tetrachlorochromate(II) and propane-diammonium tetrachlorocuprate(II) have been carried out. Both these structures form 2-dimensional layers of linked $\text{MCl}_4^{2-}$ layers, separated by the di-cations, as shown in fig 7.1.

![fig 7.1 Propane-diammonium tetrachlorochromate(II) structure.](image-url)
7.2 Method

Molecular mechanics calculations were performed on the di-ethane-tri-ammonium(II) tetrachlorochromate unit-cell, and the force-field parameters were altered until the unit-cell was reproduced. These force-constants were then used in calculations on propane-diammonium tetrachlorochromate(II). Calculations were also performed, in order to reproduce the propane-diammonium tetrachlorocuprate(II) unit cell.

7.3 Force constants

Initial force constants were obtained from I.R. data for metal-ligand vibrations. Parameters used to reproduce the di-ethane-tri-ammonium(II) tetrachlorochromate unit-cell are shown in table 7.1.

Table 7.1 Parameters that reproduce the tetrachlorochromate unit cell.

<table>
<thead>
<tr>
<th>Van der Waals parameters.</th>
<th>( r_0 )</th>
<th>( E_0 )</th>
<th>Charge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cl</td>
<td>1.80</td>
<td>0.120</td>
<td>-0.99</td>
</tr>
<tr>
<td>Cr</td>
<td>0.83</td>
<td>0.120</td>
<td>+2</td>
</tr>
<tr>
<td>H</td>
<td>1.00</td>
<td>0.020</td>
<td>CNDO Calcn</td>
</tr>
<tr>
<td>N</td>
<td>1.85</td>
<td>0.080</td>
<td>CNDO Calcn</td>
</tr>
<tr>
<td>C</td>
<td>1.80</td>
<td>0.060</td>
<td>CNDO Calcn</td>
</tr>
</tbody>
</table>
7.1.1 Bond parameters for the tetrachlorochromate unit cell

V Cr-Cl stretch 315 cm\(^{-1}\) \(35\)

mass Cr = 52
mass Cl = 35.433

reduced mass = \((52 \times 35.5)/(52+35.5)\)
= \(21.097 \times 1.67 \times 10^{-24}\)

\(k = (2 \times \pi \times 315 \times 3 \times 10^8)^2 \times 3.53 \times 10^{-23}\)
= 12.445 dynes cm\(^{-1}\)

Aryl CH V = 3050 cm\(^{-1}\) \(24\)

\(k = (2 \times \pi \times 3050 \times 3 \times 10^8)^2 \times (12/13) \times 1.673 \times 10^{-24}\)
\(k = 3.3 \times 10^{25} \times (12/13) \times 1.673 \times 10^{-24}\)
\(k = 51.05 \text{ dynes cm}^{-1}\)

\((k = 346 \text{ Kcal mol}^{-\circ})\)

ratio (Kcal mol\(^{-\circ}\) / (dynes cm\(^{-1}\)) = 6.7

\(k \text{ Cr-Cl} = 84.34 \text{ Kcal mol}^{-\circ}\)

**Table 7.2 Parameters that reproduce the tetrachlorochromate unit cell bond stretching.**

<table>
<thead>
<tr>
<th>bond type</th>
<th>(k)</th>
<th>(r_0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cl (short)</td>
<td>84.3</td>
<td>2.380</td>
</tr>
<tr>
<td>Cr-Cl (long)</td>
<td>1.7</td>
<td>2.380</td>
</tr>
<tr>
<td>N-H</td>
<td>331.0</td>
<td>1.04</td>
</tr>
<tr>
<td>C-N</td>
<td>367.0</td>
<td>1.471</td>
</tr>
<tr>
<td>C-H</td>
<td>331.0</td>
<td>1.090</td>
</tr>
</tbody>
</table>
7.1.2 Angle parameters for the tetrachlorochromate unit cell.

Cl-Cu-Cl bend

\[ V_{\text{bend}} = 125 \text{ cm}^{-1} \]

\[ \text{reduced mass} = \frac{(63.54 \times 35.5 \times 35.5)}{(63.54 + 35.5 + 35.5)} \]

\[ \text{reduced mass} = 595.2 \times 1.673 \times 10^{-24} \]

\[ k = (2 \times \pi \times 125 \times 3 \times 10^8)^2 \times 9.96 \times 10^{-22} \]

\[ k = 55.294 \text{ dynes cm}^{-1} \]

Aryl bend \( V = 671 \text{ cm}^{-1} \)

\[ \text{reduced mass} = \frac{((12 \times 12 \times 1)}{(12 + 12 + 12)} = 0.8571 \]

\[ k = (2 \times \pi \times 671 \times 3 \times 10^8)^2 \times 9.96 \times 10^{-22} \]

\[ k = 31.995 \text{ dynes cm}^{-1} \]

\[ k = 35 \text{ kcal mol} \text{ Å}^{-2} \]

\[ \text{ratio (kcal mol} \text{ Å)/(dynes cm}^{-1}) = 1.0939 \]

\[ k = 60.487 \text{ kcal mol} \text{ Å} (\text{Cl-Cu-Cl}) \]

assuming Cl-Cr-Cl bend = 125 cm\(^{-1}\)

\[ \text{reduced mass} = \frac{(35.5 \times 35.5 \times 51.996)}{(135.5 + 35.5 + 51.996)} \]

\[ \text{reduced mass} = 532.76 \times 1.673 \times 10^{-24} \]

\[ k = (2 \times \pi \times 125 \times 3 \times 10^8)^2 \times 8.91 \times 10^{-22} \]

\[ k = 54.11 \text{ Kcal mol} \text{ Å} \]

Table 7.3 Parameters that reproduce the tetrachlorochromate unit cell.

Angle bending parameters.

<table>
<thead>
<tr>
<th>angle type</th>
<th>( k )</th>
<th>( \theta_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cr-Cr (bridge)</td>
<td>35</td>
<td>163.5</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>54.11</td>
<td>90.0</td>
</tr>
<tr>
<td>Cl-Cr-Cl trans</td>
<td>54.11</td>
<td>180.0</td>
</tr>
<tr>
<td>C-N-H</td>
<td>35</td>
<td>118.4</td>
</tr>
<tr>
<td>N-C-H</td>
<td>35</td>
<td>109.5</td>
</tr>
<tr>
<td>C-C-H</td>
<td>35</td>
<td>109.5</td>
</tr>
</tbody>
</table>
Table 7.4 Parameters that reproduce the tetrachlorocuprate unit cell.

**Van der Waals parameters.**

<table>
<thead>
<tr>
<th>atom type</th>
<th>rO</th>
<th>E0</th>
<th>Charge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cl</td>
<td>1.8</td>
<td>0.120</td>
<td>-0.99</td>
</tr>
<tr>
<td>Cu</td>
<td>0.72</td>
<td>0.120</td>
<td>+2</td>
</tr>
</tbody>
</table>

7.1.3 Bond parameters for the tetrachlorocuprate unit cell

mass Cu = 63.54  
mass Cl = 35.5

reduced mass = (63.54 x 35.5)/(63.54 + 35.5)  
= 22.77 x 1.67339 x 10^{-24}

k = (2 x PI x 275 x 3 x 10^8)^2 x 3.81 x 10^{-23}  
= 10.237 dynes cm^{-1}

= 69.38 kcal mol Å  
ratio to Cr-Cl stretch = 0.9068

therefore for weak bond

k = 1.7 x 0.9068 Kcal mol Å  
k = 1.541 Kcal mol Å

Table 7.5 Parameters that reproduce the tetrachlorocuprate unit cell.

**Bond stretching parameters.**

<table>
<thead>
<tr>
<th>Bond type</th>
<th>k</th>
<th>rO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu-Cl (short)</td>
<td>69.38</td>
<td>2.26</td>
</tr>
<tr>
<td>Cu-Cl (long)</td>
<td>1.40</td>
<td>2.26</td>
</tr>
<tr>
<td>N-H</td>
<td>331.0</td>
<td>1.04</td>
</tr>
<tr>
<td>C-N</td>
<td>367.0</td>
<td>1.47</td>
</tr>
<tr>
<td>C-H</td>
<td>331.0</td>
<td>1.09</td>
</tr>
</tbody>
</table>
Table 7.6 Parameters that reproduce the tetrachlorocuprate unit cell.

Angle bending parameters.

<table>
<thead>
<tr>
<th>Angle type</th>
<th>k</th>
<th>$\theta_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu-Cl-Cu</td>
<td>35</td>
<td>165.3</td>
</tr>
<tr>
<td>Cl-Cu-Cl(cis)</td>
<td>60</td>
<td>90.0</td>
</tr>
<tr>
<td>Cl-Cu-Cl(trans)</td>
<td>60</td>
<td>180.0</td>
</tr>
<tr>
<td>C-N-H</td>
<td>35</td>
<td>118.4</td>
</tr>
<tr>
<td>N-C-H</td>
<td>35</td>
<td>109.5</td>
</tr>
<tr>
<td>C-C-H</td>
<td>35</td>
<td>109.5</td>
</tr>
</tbody>
</table>

7.4 Assumptions

Since we are unable to calculate partial charges on metal ions by quantum chemical techniques, it was assumed that as the chlorochromate structure would be highly ionic, formal charges on the chlorine and chromium ions would adequately describe the electrostatic contribution to the energy.

Partial charges were calculated for the organic cations using CNDO.

7.5 Results

Table 7.5.1: Diethane-tri-ammonium tetrachlorochromate(II)

<table>
<thead>
<tr>
<th>bonds</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cl long</td>
<td>2.89</td>
<td>2.80</td>
</tr>
<tr>
<td>Cr-Cl short</td>
<td>2.39</td>
<td>2.39</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>angles</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cr-Cr bridging</td>
<td>162.70</td>
<td>162.26</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>89.00</td>
<td>88.26</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>91.00</td>
<td>90.74</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>181.00</td>
<td>180.02</td>
</tr>
</tbody>
</table>
Table 7.5.2: Propane-diammonium tetrachlorochromate(II)

<table>
<thead>
<tr>
<th>bonds</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cl long</td>
<td>2.80</td>
<td>2.85</td>
</tr>
<tr>
<td>Cr-Cl short</td>
<td>2.39</td>
<td>2.41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>angles</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr-Cl-Cr bridging</td>
<td>164.74</td>
<td>164.07</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>80.95</td>
<td>80.70</td>
</tr>
<tr>
<td>Cl-Cr-Cl cis</td>
<td>99.76</td>
<td>99.29</td>
</tr>
<tr>
<td>Cl-Cr-Cl trans</td>
<td>180.54</td>
<td>179.99</td>
</tr>
</tbody>
</table>

Table 7.5.3: Propane-diammonium tetrachlorocuprate(II)

<table>
<thead>
<tr>
<th>bonds</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu-Cl long</td>
<td>2.95</td>
<td>3.00</td>
</tr>
<tr>
<td>Cu-Cl short</td>
<td>2.27</td>
<td>2.30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>angles</th>
<th>calculated</th>
<th>crystal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cu-Cl-Cu bridging</td>
<td>165.70</td>
<td>164.00</td>
</tr>
<tr>
<td>Cl-Cu-Cl cis</td>
<td>80.85</td>
<td>80.91</td>
</tr>
<tr>
<td>Cl-Cu-Cl cis</td>
<td>99.84</td>
<td>99.34</td>
</tr>
<tr>
<td>Cl-Cu-Cl trans</td>
<td>180.43</td>
<td>180.00</td>
</tr>
</tbody>
</table>

7.6 Conclusion

It can be seen that unit-cells may be modelled reasonably well by these techniques and predictions can be made if the structures are similar. However as mentioned in 9.1 the intermolecular interactions are so complex and varied that the derived potential parameters are not valid outside the basis set of compounds. Thus in deriving chlorochromate type parameters, there is no justification in using these parameters on structurally dissimilar
unit-cells. Thus the predictive power of molecular-mechanics as a general tool for crystal engineering is currently severely limited by the inability to describe molecular interactions in sufficient enough detail.
CHAPTER EIGHT

APPLICATION OF MOLMECH TO METALLOPROTEINS
Application of the molecular mechanics program MOLMECH to metalloproteins

Most molecular mechanics programs in current use include Coulombic interactions among point charges placed at atomic centres as part of the force-field acting on the various atoms. However, there seems to be no general agreement about the numerical values of the charges to be assigned to the atoms or even about how these values are to be ascertained in principle from experimental or other evidence. There is also a lack of consensus about the value of the dielectric constant to be applied in such calculations. These uncertainties are particularly troublesome in attempts to estimate the relative energies of different arrangements of ligands around a central metal atom, as they might occur in the binding of substrate molecules to metalloproteins. Here, calculations based on inappropriate atomic charges can lead to atomic arrangements around metal centres that lack any resemblance to the more frequently observed types found in small-molecule crystal structures. They can also fail to reproduce commonly occurring arrangements.

Thus, although tetrahedral and octahedral arrangements of ligand atoms around a metal centre present no problems, there are sometimes difficulties in dealing with five coordination. Calculations based on partial atomic charges often fail to duplicate the most commonly observed coordination types, the trigonal bipyramid and the square pyramid.

These arrangements are indeed obtained as stable isolated structures for monoatomic ligands, but they may become unstable when the ligand atoms are embedded in rigid substructures that contain other charged atoms. In such cases, the calculation involving monopoles may lead to a (4+1) distorted tetrahedron, i.e. to an arrangement with one ligand atom much more distant from the metal centre than the other four, contrary to what is observed in small molecule crystal structures.

The Empirical Potential Metal function, reported by Vecdani et al., assumes that, whatever the detailed nature of the atom-atom interactions may be, the environment of the metal atom in the unknown target structure cannot deviate much from one or other of the arrangements observed in known structures. Thus by constructing the appropriate energy functions, the target can be forced to be as close as possible to one of these observed arrangements. In practice, the Cambridge Structural Database was used to identify common intra and intermolecular bonding patterns and
derive therefrom features of corresponding potential functions that model the observed distributions.

The potential function proposed by Vedani et al is:

\[ E_{\text{total}} = \sum_{k=1}^{n} \left( A_k \cdot \frac{1}{r_{k,i}} - C_k \cdot \cos^2(\theta_{k,i}) \right) \]

with:
\[ A_k = -5.0 \cdot E_0 \cdot r_0 \]
\[ C_k = -6.0 \cdot E_0 \cdot r_0 \]

The sum term allows each metal-ligand distance to be varied separately. The coefficients \( A_k \) and \( C_k \) depend on the equilibrium distance \((r_0)\) and on the well depth \((E_0)\) for each M..L interaction, both of which depend on the coordination type as well as on the nature of the atoms involved. The values assigned to \( E_0 \) were chosen on a rather ad-hoc basis.

**Analysis of Zn(II) coordination spheres**

Vedani et al arbitrarily defined the coordination shell of Zn to include all first row atoms (N,O) closer than 2.40Å and all second row atoms (S,Cl) closer than 2.60Å. With information gathered from the CSD they have analyzed the geometry of four-, five, and six-coordinated zinc in this way. The criteria for accepting an entry from the CSD were: (1) no disorder, (2) no strong metal-metal interactions in the structure, (3) a crystallographic R factor less than 0.06.

They found 118 fragments containing four coordinated zinc, all showing a tetrahedral arrangement of the ligands. For five-coordinated zinc, the 25 fragments found can be classified on the basis of their bond angles, see fig 8.1, as belonging essentially to two coordination types, square pyramids and the trigonal bipyramids, with three fragments intermediate between these types. In the trigonal bipyramids, the Zn atom is always close to the plane of the equatorial ligands (rms displacement 0.039Å), whereas in the square pyramids it deviates markedly from the basal plane (mean displacement 0.370Å). For six-coordinated Zn, 60 fragments with predominantly octahedral coordination were found. The Zn-ligand distances increase with increase in the coordination number as indicated in Table 8.1

The variation in Zn-ligand distance is larger for axial than for equatorial
ligands; indeed, in a few square pyramids, trigonal bipyramids and octahedra, the axial ligands are so distant from the Zn atom (but still included in the above definition) that the coordination types could also be described as (4+1), (3+2), and (4+2) respectively. These fragments have been excluded from the data given in table 8.1.

\[
\frac{a_{14} + a_{15}}{2}
\]

\[
\frac{a_{12} - a_{13}}{2}
\]

fig 8.1 Distribution of \((a_{12} + a_{13})/2\) vs. \((a_{14} + a_{15})/2\) for five-coordinated Zn compounds with N, O and S ligands.

Table 8.1 Zinc-ligand distances observed in small-molecule crystal structures retrieved from the CSD.

<table>
<thead>
<tr>
<th>Ligand</th>
<th>Four-coordination</th>
<th>Five-coordination</th>
<th>Six-coordination</th>
</tr>
</thead>
<tbody>
<tr>
<td>O</td>
<td>1.89..2.05 - mean 1.98(5)Å</td>
<td>1.93..2.21 - mean 2.04(7)Å</td>
<td>1.99..2.23 - mean 2.11(5)Å</td>
</tr>
<tr>
<td>2.11(5)Å</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>1.92..2.14 - mean 2.04(6)Å</td>
<td>1.99..2.33 - mean 2.10(6)Å</td>
<td>2.08..2.28 - mean 2.15(5)Å</td>
</tr>
<tr>
<td>2.15(5)Å</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>2.26..2.45 - mean 2.35(6)Å</td>
<td>2.31..2.46 - mean 2.38(7)Å</td>
<td></td>
</tr>
<tr>
<td>2.36..2.46 - mean 2.38(5)Å</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cl</td>
<td>2.17..2.35 - mean 2.26(5)Å</td>
<td>2.23..2.31 - mean 2.26(5)Å</td>
<td>2.36..2.46 - mean 2.40</td>
</tr>
<tr>
<td>(5)Å</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Br</td>
<td>2.32..2.40 - mean 2.38(3)Å</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\*Distal polar ligands observed in a few crystal structures were omitted.
8.1 Calculations on HCA II

Carbonic anhydrase, a monomeric zinc-enzyme consisting of 260 amino acids, catalyses the reversible hydration of carbon dioxide to bicarbonate. The crystal structure of human carbonic anhydrase II has been determined to a resolution of 2.0 Å.

The refinement of the active site region with the program 'MOLMECH' included 525 atoms of the enzyme and up to 25 atoms of the substrate molecule. A distant dependent dielectric constant was used. The rationale is that by using a distance dependence the polarisation effects for closer interactions are weighed more heavily and longer-range interactions are dampened more than shorter-range interactions. The dielectric constant used was $E=4\pi$.

The angle term in the reported empirical potential function for metal centres reported by Vedani appears by inspection to only work efficiently when the $12-10$ term is negative, deviations from observed angles reducing the well depth of the function. fig(8.2-8.3).

fig 8.2 potential energy map for the potential function
fig 8.3 potential energy map for the potential function

The severity of the function for the ideal angles \( \theta_0 \) may be too small and can be increased by using the following function which penalises deviations from ideal angles to a degree dependent on the constant \( k \).

\[
E_{\text{net}} = \sum_{\text{bonds}} \left( \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \right) \cdot \sum_{\text{angles}} k(\theta - \theta_0)^2
\]

\[\text{When } \sum \left( \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \right) + \text{ve} \]

\[
E_{\text{net}} = \sum_{\text{bonds}} \left( \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \right) \cdot \sum_{\text{angles}} k(\theta - \theta_0)^2
\]

\[\text{When } \sum \left( \frac{A''}{r^{12}} - \frac{C''}{r^{10}} \right) - \text{ve} \]

This function is more likely to reproduce commonly observed distributions of ligands around the metal centre as found in the CSD. The potential map for this function is shown in fig 8.4
Refinements were thus carried out using:

1/ Metal function 1. Vedani\textsuperscript{24}

2/ Metal function 2.

3/ No metal function, with electrostatic charges.

\textit{8.1.1 Method}

A 10 A window centred on the zinc atom was extracted from the crystal coordinates of the enzyme. Appropriate charges, atomtypes and force-field parameters were assigned using a specially written pre-formatting program. Charges and parameters were taken from Weiner and Kollman\textsuperscript{25} Nucleic acid and protein force-field. They used quantum mechanical calculations of the electrostatic potential to derive charges for atoms in salient molecules\textsuperscript{25}. This method uses quantum mechanically calculated electrostatic potentials to numerically fit atomic charge models.

Charges for substrate molecules were calculated using CNDO quantum mechanical techniques. Parameters used for the metal function were taken from Vedani et al and are shown in table 8.2.
Table 8.2 Equilibrium zinc...ligand distances $r_0$ (in A) and well-depths $E_0$ (in Kcal/mol)

<table>
<thead>
<tr>
<th>Ligand type</th>
<th>Tetrahedron</th>
<th>Square Pyramid</th>
<th>Trigonal Bipyramid</th>
<th>Octahedron</th>
</tr>
</thead>
<tbody>
<tr>
<td>O</td>
<td>1.95</td>
<td></td>
<td>2.05</td>
<td>2.10</td>
</tr>
<tr>
<td></td>
<td>-6.00</td>
<td></td>
<td>-5.50</td>
<td>-5.25</td>
</tr>
<tr>
<td>N</td>
<td>2.00</td>
<td></td>
<td>2.10</td>
<td>2.15</td>
</tr>
<tr>
<td></td>
<td>-5.75</td>
<td></td>
<td>-5.25</td>
<td>-5.00</td>
</tr>
<tr>
<td>S</td>
<td>2.35</td>
<td></td>
<td>2.45</td>
<td>2.50</td>
</tr>
<tr>
<td></td>
<td>-5.50</td>
<td></td>
<td>-5.00</td>
<td>-4.75</td>
</tr>
</tbody>
</table>

Table 8.3 Equilibrium lig$_i$...Zn...lig$_j$ angles (in degrees)

<table>
<thead>
<tr>
<th>Type</th>
<th>Tetrahedron</th>
<th>Square Pyramid</th>
<th>Trigonal Bipyramid</th>
<th>Octahedron</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>109.5</td>
<td></td>
<td>180.0</td>
<td>180.0</td>
</tr>
<tr>
<td>ax.Zn.ax</td>
<td>ax.Zn.eq</td>
<td>eq.Zn.eq</td>
<td></td>
<td></td>
</tr>
<tr>
<td>109.5</td>
<td>100.3</td>
<td>159.5</td>
<td>88.2</td>
<td></td>
</tr>
<tr>
<td>180.0</td>
<td>90.0</td>
<td>120.0</td>
<td>90.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>180.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.1.2 Results

8.1.2.1 Native Human Carbonic Anhydrase II (HCA II)

The Zn atom in native HCA II is tetrahedrally coordinated by three histamine N atoms and by the O atom of a water molecule. Refinements of the active site yielded an almost ideal tetrahedron by all three functions. (fig 8.5)

8.1.2.2 The natural substrate - bicarbonate

The accommodation of the natural substrate is found to involve the substrate binding to the zinc close to square pyramid geometry (see fig 8.6)
fig 8.5

fig 8.6 The natural substrate Bicarbonate

90
8.1.2.3 Sulphonamide Inhibitors

It was for complexes of HCA II with heterocyclic sulphonamides, (very specific inhibitors of carbonic anhydrase), that Vedani et al reported, resulted in distorted tetrahedral structures. These inhibitors are believed to bind as bidentate ligands to the metal. Although coordination of the sulphonamide grouping via NH₂ to the Zn has been generally assumed, there is no solid experimental basis for this. Indeed, not a single SO₂N(RH)...Zn or -CON(RH)...Zn interaction is known to occur in small molecule crystal structures with amides. In R-SO₂NH₂ crystal structures where H positions have experimentally determined, the N atom is clearly pyramidal, usually 0.25 - 0.30 Å out of plane of its three bonded neighbours. Based on the putative -NH₂...Zn interaction, this arrangement has been used in the refinements.

Table 8.3 Pre-minimised geometry

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-N</td>
</tr>
<tr>
<td>94.4</td>
<td>1.843</td>
</tr>
<tr>
<td>97.9</td>
<td>1.090</td>
</tr>
<tr>
<td>107.88</td>
<td>1.727</td>
</tr>
<tr>
<td>70.65</td>
<td>Zn-O</td>
</tr>
<tr>
<td>78.41</td>
<td>2.321</td>
</tr>
<tr>
<td>91.8</td>
<td>1.806</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td></td>
</tr>
<tr>
<td>115.36</td>
<td></td>
</tr>
<tr>
<td>124.64</td>
<td></td>
</tr>
<tr>
<td>115.36</td>
<td></td>
</tr>
<tr>
<td>ax-Zn-ax</td>
<td></td>
</tr>
<tr>
<td>164.39</td>
<td></td>
</tr>
</tbody>
</table>
Table 8.5 Metal function 1 results (trigonal bipyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-N</td>
</tr>
<tr>
<td>96.2</td>
<td>2.104</td>
</tr>
<tr>
<td>89.6</td>
<td>2.059</td>
</tr>
<tr>
<td>93.6</td>
<td>2.119</td>
</tr>
<tr>
<td>66.8</td>
<td>Zn-O</td>
</tr>
<tr>
<td>93.7</td>
<td>2.079</td>
</tr>
<tr>
<td>101.3</td>
<td>2.162</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td></td>
</tr>
<tr>
<td>115.7</td>
<td></td>
</tr>
<tr>
<td>123.8</td>
<td></td>
</tr>
<tr>
<td>115.79</td>
<td></td>
</tr>
<tr>
<td>ax-Zn-ax</td>
<td></td>
</tr>
<tr>
<td>162.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.6 Metal function 2: Results (trigonal bipyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-N</td>
</tr>
<tr>
<td>97.47</td>
<td>2.104</td>
</tr>
<tr>
<td>88.87</td>
<td>2.072</td>
</tr>
<tr>
<td>94.60</td>
<td>2.117</td>
</tr>
<tr>
<td>67.09</td>
<td>Zn-O</td>
</tr>
<tr>
<td>92.04</td>
<td>2.078</td>
</tr>
<tr>
<td>101.00</td>
<td>2.145</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td></td>
</tr>
<tr>
<td>117.29</td>
<td></td>
</tr>
<tr>
<td>126.41</td>
<td></td>
</tr>
<tr>
<td>114.93</td>
<td></td>
</tr>
<tr>
<td>161.10</td>
<td></td>
</tr>
</tbody>
</table>
Table 8.7 No metal centre function with partial charges: Results (trigonal bipyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-N 2.093</td>
</tr>
<tr>
<td>96.49</td>
<td>2.074</td>
</tr>
<tr>
<td>87.70</td>
<td>2.111</td>
</tr>
<tr>
<td>67.61</td>
<td></td>
</tr>
<tr>
<td>93.05</td>
<td>Zn-O 2.066</td>
</tr>
<tr>
<td>98.67</td>
<td>2.129</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td></td>
</tr>
<tr>
<td>114.26</td>
<td></td>
</tr>
<tr>
<td>128.90</td>
<td></td>
</tr>
<tr>
<td>116.0</td>
<td></td>
</tr>
<tr>
<td>ax-Zn-ax</td>
<td></td>
</tr>
<tr>
<td>164.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.8 Metal function 1: results (square pyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-cq</td>
<td>Zn-O 2.168</td>
</tr>
<tr>
<td>104.11</td>
<td>2.007</td>
</tr>
<tr>
<td>129.84</td>
<td>2.103</td>
</tr>
<tr>
<td>89.63</td>
<td>2.092</td>
</tr>
<tr>
<td>86.94</td>
<td></td>
</tr>
<tr>
<td>eq-Zn-cq</td>
<td></td>
</tr>
<tr>
<td>110.85</td>
<td>2.130</td>
</tr>
<tr>
<td>68.79</td>
<td></td>
</tr>
<tr>
<td>75.48</td>
<td></td>
</tr>
<tr>
<td>104.60</td>
<td></td>
</tr>
<tr>
<td>trans</td>
<td>142.37</td>
</tr>
<tr>
<td>trans</td>
<td>142.95</td>
</tr>
</tbody>
</table>
Table 8.9 Metal function 2: Results (square pyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-O</td>
</tr>
<tr>
<td></td>
<td>104.68</td>
</tr>
<tr>
<td></td>
<td>118.83</td>
</tr>
<tr>
<td></td>
<td>98.03</td>
</tr>
<tr>
<td></td>
<td>86.13</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td>Zn-N</td>
</tr>
<tr>
<td></td>
<td>99.15</td>
</tr>
<tr>
<td></td>
<td>68.57</td>
</tr>
<tr>
<td></td>
<td>83.34</td>
</tr>
<tr>
<td></td>
<td>101.86</td>
</tr>
<tr>
<td>trans</td>
<td>149.87</td>
</tr>
<tr>
<td>trans</td>
<td>153.85</td>
</tr>
</tbody>
</table>

Table 8.10 No metal function with partial charges (square pyramid)

<table>
<thead>
<tr>
<th>Angles</th>
<th>Bond lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>ax-Zn-eq</td>
<td>Zn-O</td>
</tr>
<tr>
<td></td>
<td>96.83</td>
</tr>
<tr>
<td></td>
<td>137.02</td>
</tr>
<tr>
<td></td>
<td>92.93</td>
</tr>
<tr>
<td></td>
<td>88.35</td>
</tr>
<tr>
<td>eq-Zn-eq</td>
<td>Zn-N</td>
</tr>
<tr>
<td></td>
<td>110.30</td>
</tr>
<tr>
<td></td>
<td>68.39</td>
</tr>
<tr>
<td></td>
<td>78.51</td>
</tr>
<tr>
<td></td>
<td>108.09</td>
</tr>
<tr>
<td>trans</td>
<td>140.55</td>
</tr>
<tr>
<td>trans</td>
<td>134.47</td>
</tr>
</tbody>
</table>
fig 8.7 trigonal bipyramid geometry
with acetazolamide inhibitor

fig 8.8 square pyramid geometry
with acetazolamide inhibitor
8.1.3 Conclusion

It can be seen from the results that the distorted structures observed by Vedani at al did not occur and that the minimisation appears to be fairly insensitive to the partial charges assigned to atoms in the model.

This implies that the charges obtained by Weiner et al\textsuperscript{1} for nucleic acids and proteins are good approximations and do not lead to distortions at metal centres.

However, since it is better to leave out inappropriately assigned charges than take the risk to include them in the calculation, the use of an Empirical metal potential function is useful.

On the whole, as expected, the metal function 2 produced structures closest to the required geometry. Molmech is, therefore, a useful tool in molecular mechanics investigations of metalloproteins.
8.2 Modelling of TFIIIA Transcription factor

8.2.1 Transcription in Eukaryotes. Introduction

The fact that DNA is transcribed into RNA has long been known and has become part of the 'central dogma' of molecular genetics. The mechanisms responsible for controlling this process at the individual genes of higher cells (eukaryotes), however, are still not completely understood. RNA polymerases apparently require a number of auxiliary factors (transcription factors) for gene recognition. These factors combine with the enzyme at the gene to form a transcription complex. The structures of these complexes are starting to become clearer, most is known about the control of RNA polymerase III, the enzyme responsible for the synthesis of certain small RNA molecules. TFIIIA is an especially studied protein, which is a positive regulator for the expression of ribosomal 5s RNA and possesses structural properties that were previously unknown in DNA-binding proteins. It is becoming increasingly evident that the architecture of TFIIIA is not an exotic curiosity but exemplifies a general structural plan.

8.2.2 Structure of TFIIIA

TFIIA requires zinc ions as a cofactor in order to recognise the 5s RNA genes. This was first demonstrated in footprint experiments with Xenopus factor A. It was later shown that the human A factor likewise needs Zinc in order to function.

The cDNA of TFIIIA was sequenced at about the same time, which meant that its amino-acid sequence was also known. TFIIIA is a basic protein containing 344 amino-acids. Computer-aided analysis of its primary sequence showed that, starting at its N-terminus, TFIIIA consists of nine repetitions of a specific sequence, the remaining C-terminal sequence may be viewed as being composed of three degenerate units of the same structural element.

The main feature of this motif is the constant arrangement of a pair of cysteine residues and a pair of histidines (fig8.9), which are assumed to bind a Zinc ion by means of tetrahedral coordination. The intervening sequence generally consists of twelve amino-acids and is in contact with a specific sequence of DNA bases. Two positions, a phenylalanine and a leucine residue, are highly conserved in this finger.
Since experimental data on the three-dimensional folding of TFIIIA are not yet available, one has to rely on predictions of secondary structure. Predictions have been made by Brown et al.\textsuperscript{44} using calculations based on the models proposed by Chou and Fasman\textsuperscript{45} and Robson et al.\textsuperscript{46}

These studies indicate that an alpha-helix consisting of two to three turns is formed in the second half of the finger sequence. This helix includes the amino acids between the conserved phenylalanine and leucine residues. The cysteine residues, on the other-hand, are located in regions that are less structurally organised or are characterised by a higher beta-turn probability so that they can adopt an arrangement that is suitable for binding zinc.
Helices of similar length, which embed themselves in the major groove of the DNA helix, are also responsible for establishing specific contacts between DNA and bacterial or viral repressors. The 'recognition helix' of those proteins also has polar amino acids on the side facing the DNA. The hydrophobic side chains, on the other hand, are found on the other side facing the rest of the protein. Stabilising contact is thus established with a second alpha-helix, which is connected to the first by a turn (helix-turn-helix motif).

The TFIIIA structure, in contrast, is stabilised by the presence of a zinc ion; at any rate the structural prediction models do not suggest the existance of other regions with significant alpha-helix probabilities. The hydrophobicity of the "back", therefore, does not need to be very pronounced and is often restricted to the conserved leucine residue.

8.2.3 The model

The sequence of amino acids used for the model is shown below.\textsuperscript{74}

\begin{verbatim}
PVYKRICSFA DC G A Y N K N W K L Q L A H L C R K H
\end{verbatim}

The amino-acid residues shown in underlined italics are essential for the finger structure and are thus particularly well conserved. Well conserved residues are shown in italics. (Individual amino-acids are designated according to the one-letter code proposed by Dayhoff.\textsuperscript{75})

A three turn alpha-helix was constructed between the conserved phe and leu residues. The model was minimised using MM2 and refined using MOLMECH, with parameters for the metal function as defined previously. Plate 8.1 and 8.2 show the resulting model.

Total energy = 20.45 Kcal/mol  
Rms = 0.14  
Zn-S bonds - 2.34 Å  
Zn-N bonds - 2.06 Å  
The L-Zn-L angles show distortions from tetrahedral symmetry with S-Zn-S = 109.5° and N-Zn-N 101.9° and the four S-Zn-N angles in the range of 108-115°.
Conclusions
The energetics of the model suggest that the proposed system is a reasonable one. At the
time of this work, no further structural studies had been carried out, thus preventing
verification of the model. Recently some nmr studies\textsuperscript{76-77} have produced some
structures which are stored on the Brookhaven protein database. Restricted access to
these files, however, has again prevented verification of the model. The use of
molecular mechanics in this instance is to test the viability of a proposed structure on
purely energetic grounds.
CHAPTER NINE

LIMITATIONS OF THE ATOM-ATOM POTENTIAL 
AND MOLECULAR MECHANICAL TECHNIQUES 
IN 'CRYSTAL ENGINEERING'
9.1 Limitations of the Atom-Atom potential and Molecular Mechanical techniques in Crystal Engineering.

It has been shown that molecular mechanical techniques can be used to model structurally similar systems. Attempting to use these techniques in a wider more general way as a predictive tool in crystal engineering raises a number of serious limitations, which can be summarised as follows:

Polymorphism / Local minima.

Very small energy differences are obtained between alternative packings for which clear preferences exist in the crystal.

The intermolecular interactions are so complex and varied that the derived potential parameters are not valid outside the basis set of compounds.

Interactions between hetero-atoms in structures (H-bonding) are highly directional and tend to cause deviations from close packing in the lattice.

9.1.1 Polymorphism

Polymorphs are crystal structures containing the same compound, but which differ in the packing arrangement in the crystal.

In a chemical system such as a crystal lattice, the different possible arrangements of atoms have different potential energies owing to the electrical and other interactions between them. The system will spontaneously tend to take up the structure with the minimum potential energy. In a simple system with only a few possible structures, one may have a distinctly lower energy than the others; in fig 9.1A this is represented by the minimum at the bottom of the potential well. Other less stable possibilities are represented by local minima on the side of the well. In systems of increasing complexity, the number of possible structures increases (fig 9.1 B,C,D). As it does, so the chance of there being a unique minimum energy structure seems likely to diminish. In the situation represented by fig (9.1.D) several different structures would be equally stable from an energetic point of view. These could correspond to possible polymorphic structures. In attempting to make structure predictions by energy considerations, then, it would not be possible to be sure that the structure predicted corresponded to the properties we desired.
9.1.2 Small energy differences between alternate packings.

Clearly the inability to distinguish between alternate packings energetically by these methods, although clear preferences may exist in the structure, means an inability to predict a preferred packing. This is another way of putting the problem of polymorphism (9.1.1).
9.1.3 Derived parameters are not valid outside the basis set.

In the calculations performed in chapter 6, the parameters derived for chlorochromate and chlororcuprate lattices succeeded in reproducing analogous structures, but they could not be used to predict structures to any degree of success outside that set of compounds. New parameters would have to be derived for different structure types. This is obviously a serious limitation for a general structure prediction technique.

9.1.4 Directional character of some interactions.

The fact that hetero-atom interactions are directional, causes subtle interactions that are long range in character and these must be modelled adequately to stand a chance of predicting structure. Atom-atom potential models and molecular mechanical techniques do not adequately model these interactions and therefore cannot distinguish subtle interactional differences that occur between different packings. Hetero-atom interactions tend to cause deviations from close-packing. Molecular mechanics and atom-atom potential techniques tend to assume close packing.
9.2 Towards a general crystal engineering approach using structural databases.

9.2.1 Intermolecular patterns in crystals

In view of the numerous problems associated with the atom-atom potential and molecular-mechanical methods, qualitative pattern recognition from retrieved crystal structures has become an attractive method of crystal structure prediction and design. It has been commented by Dauber and Hagler\textsuperscript{48} that while crystal structures may be accurately derived from correct potentials, the converse need not be true. A knowledge of the crystal structure does not necessarily lead to the correct potentials since a variety of potentials may lead to the same approximate structure.

It has become quite evident that we will not be in a position, at least in the near future, to have a set of accurate and general potentials with which to predict a large number of hetero-atom organic and inorganic structures. Hence, crystallographers and structural chemists have tended toward the more imperfect but pragmatic solution of employing known crystal structures to derive unknown ones. In the process one need not be necessarily cognisant of the exact nature of all the intermolecular forces involved. Significant structural results, which have often suggested changes in the theoretical model, have thus been obtained.

The ever-increasing number of accurate crystal structures and our ability to retrieve them rapidly and selectively make the statistical method of crystal structure design an attractive one. As new structures continue to be solved and reported, structural chemists have been finding in them, novel and intriguing patterns. Simultaneously, the almost instinctive urge of the chemist to classify and rationalise has resulted in new interpretations of older data. At the present time, computational methods cannot handle the breadth and variety of organic and inorganic crystal structures. However, a database of 70,000 crystal structure determinations is probably large enough to yield statistically significant subsets to evaluate most structural conjectures.

9.2.2 Crystallographic databases and the recognition of intermolecular patterns.

The disturbing and problematic feature regarding many hetero-atom containing structures is that they are difficult to predict from a set of atom-atom potential
parameters obtained from an unrelated group of structures. Of course, a set of potentials, well-parameterised for a homogeneous group of compounds, will almost always produce an energy minimum for an observed crystal-structure of a closely related compound. This is an inherent feature of the atom-atom method even when the forces responsible for the particular packing adopted are far from isotropic. Conversely, the breakdown of theory is an indication that the intermolecular forces in non-hydrocarbon crystals have their own chemical individuality which cannot be transferred from compound to compound in an indiscriminate way.

Although the physical nature of many types of intermolecular interactions in molecular solids is far from understood, the consequences of these interactions, in other words, the 70,000 odd experimentally determined crystal structures are known quite accurately. In principal, therefore, it should be possible to work backwards from observed crystal structures to formulate empirical rules about crystallisation patterns. This type of approach has particular relevance to crystal engineering. Application of X-ray crystallography over the past fifty years, has amassed a vast mine of information, encoding therein the intermolecular interactions. If a particular motif or pattern of molecules were to recur often enough in a group of crystal structures, it would, more likely than not, be found in a crystal structure of a related compound.

9.2.3 The proposed structural design software

The approach taken in designing novel software for the prediction of solid-state materials is shown in fig 9.2. Since the Cambridge database contains the consequences of intermolecular interactions, it should be possible to obtain empirical rules and to embed these rules in an artificial intelligence system (see 1.2.E), which can then use these rules to make predictions of crystallisation patterns.
work backwards formulating 
EMPIRICAL RULES ABOUT 
CRYSTALLISATION PATTERNS
CAMBRIDGE DATABASE 
CONTAINS THE CONSEQUENCES 
OF INTERMOLECULAR INTERACTIONS

choice of 
ligands & environment. 

rules of 
crystallisation embedded in artificial intelligence system 

probable patterns of crystallisation based on empirical statistics

VISUALISATION
VIEW PATTERNS
SIMULATE X-RAY PATTERNS
INTERACTIVE EDITING AND DESIGN
ENERGY CALCULATIONS

fig 9.2 The proposed structural design package
The structural design package depends heavily on the ability to visualise structures, this in turn depends on the ability to produce high quality, interactive, graphical representations, thus a thorough understanding of computer graphics programming is needed to produce the necessary software. This is discussed in the next two chapters.
CHAPTER TEN

COMPUTER GRAPHICS PROGRAMMING
Graphics workstations often support powerful graphics libraries which enable calls to transformation, rotation and clipping routines. Graphics on PC's must, however, be programed from first principles.

Modern compilers do support very comprehensive graphics features which allow the implementation of three dimensional molecular graphics.

10.1 The graphic process

To enable any view of an object to be taken on demand, four separate tasks, each of which requires the use of mathematics needs to be carried out. Fig 10.1.
These are:

1. Modelling - where the object is described in sufficient detail to enable the computer to 'understand' its geometry; also called object description.

2. Transformation - where the model is manipulated by movements, rotations and viewing calculations in order to put it in the right form for display.

3. Clipping - where the drawing is trimmed to the dimensions of the display surface to make sure it will fit within the boundaries.

4. Display - where the view is drawn on some output device.

The precise point in the process at which clipping takes place is important and this depends on the capabilities of the output device. In some cases, the clipping must take place before the drawing information is sent to the display. In other cases, it is possible to send the unclipped information and for the device itself to do the clipping.

Although, in terms of the final drawing, the results will be the same, the actual performance will be different. A device which can handle its own clipping is likely to be faster than one not able to do this. With a PC we must clip the model before it is displayed.

10.1.1 Modelling: Coordinates

At the heart of almost any method of geometric modelling is the coordinate system. The coordinate system is a framework for properly locating points and one form of these frameworks is a pair or triple axes, (depending on whether two-dimensional or three-dimensional representations are required), set orthogonally to one another.

There are a variety of other possible coordinate systems that can be used to describe objects such as the polar system. However, the Cartesian coordinate system is the most common, a name derived from that of Descartes (1596 - 1650), the French philosopher/mathematician who did much to establish the principles of coordinate geometry.
10.1.2 Three-Dimensional Coordinate Systems

In three-dimensional coordinate systems, whilst the first two coordinates are normally represented in the same way as in the two-dimensional case, the third coordinate can be represented in one of two ways.

If the two dimensional part of the coordinate system is thought of as lying in the plane of this page and going through the origin of the page. The numbers can either increase as they go away from the eye towards the plane of the page, defining a left-handed coordinate system (fig 10.2), or they can increase as they come towards the eye, in which case a right-handed system is defined (fig 10.3).
Although programs using left-handed systems can be converted to right-handed with little difficulty, it is essential that a convention is adopted and adhered to, to avoid confusion.

10.1.3 Coordinate forms

For computer graphics purposes, two basic forms of coordinates need to be considered.

1 World coordinates
The coordinate used to define the objects and elements to be depicted: they are the coordinates used to model the real world.
2 Device coordinates

The coordinates used by the device itself to display the representation; they are the coordinates used to view the model. Because of the difference of units used, it is rare for these two forms to coincide even in the two-dimensional case.

10.1.4 Device Coordinates

Device coordinates relate to the dimensions of the two-dimensional display surface we use. Many display's assume a coordinate system with the origin at the bottom left-hand corner and the positive axes running right and upwards. A few raster displays (the Apple II and the Macintosh), for instance, have their origins at the top left-hand corner with the positive axes running to the right and downwards. A rarer few locate their origins in the centre of the screen.

The newly adopted GKS standard for computer graphics requires that the device coordinates be in real dimensions, measured in metres, with the origin at the bottom left, but there has not yet been enough time for this standard to take widespread effect. Part of the job of a computer graphics program is to convert the world coordinates to device coordinates without users having to trouble themselves with the actual details of the process.

In well-structured graphics systems, three related concepts assist in this task. Normalised device coordinates, windows and viewports.

10.1.5 Normalised Device Coordinates (NDC's)

NDC's assume that the dimensions of the display surface run from 0.0 to 1.0 in both directions, regardless of its actual size or shape. NDC's are used as the intermediate units between world and device coordinates with the aim of separating the computation of a picture from its display. Thus we can compute the picture as if it were to be displayed on a notional device whose dimensions are 1.0 x 1.0 and then scale this information to the dimensions of the actual device to be used. This has the added advantage that, if the normalised information is filed away on a disk or tape, it can later be displayed on any new device with no alteration other than appropriate scaling.
When we wish to create a model, we initially have to decide on how much of the real world to depict. Designers of, say, table lamps who want to do working drawings of their products might conclude that the only part of the world they need consider is a cube of one metre in each direction. In other words, they have a window one metre wide and one metre high. In order to exploit the fact that any product is symmetrical about its vertical axes, it is likely that they will want to define their window in the horizontal direction as ranging from -500mm to +500mm and in the vertical direction from 0 to 1000mm fig (10.4).

Historians, on the other hand, might have real and estimated data on population changes from 85B.C. to A.D. 2085 and, if they wish to plot all of this, will have a window which extends horizontally, from -85 to +2085 and vertically from, say, -20000 to +1000000.

From these examples, we can see that windows are a two-dimensional concept and can be symmetrical or asymmetrical as well as of any proportion and dimension. In addition, they can be bigger or smaller than the dimension of the things they are to portray. When they are smaller, only the section of the world falling within the window will be displayed. By varying the size of the window, we can change the scale and proportion of the view - doubling the window dimensions halves the scale of the view; halving the dimensions doubles the scale. However, when we increase the scale, part of
the view will fall outside the display surface and this can cause difficulties unless the program takes the necessary precautions.

One way of dealing with the problem would be to search through the data and omit anything which is outside the window. This solution, however, would not work in most cases. In particular, to draw a perspective view of an object requires that all the data about its geometry be processed even though only part of the projection is displayed. Thus it is necessary to compute the whole drawing and then clip any of the lines which fall outside the window.

This is a simple but not a trivial mathematical process - lines running across the corner of the window, for instance, have to be clipped at both ends, as it is impossible to decide ahead of time which lines must be clipped, all have to be passed through the clipping routine.

10.1.7 Viewports

The discussion on windowing tacitly assumed the whole display surface would be used to show the windowed drawing. Moreover, it is quite likely that we will wish to use only part of the display-either to leave room for text or to allow multiple drawings to be shown on the same viewing surface. In order to accommodate this requirement we have the concept of the viewport, which is a rectangular portion of the display surface defined in device or normalised device coordinates onto which we map the desired window (fig.10.5)
Three important points have to be noted about viewpoints:

1. Unless distortion is to result, the proportions of both windows and viewports must match.

2. Whilst viewports can be defined as smaller than display surfaces, in most systems they must not be defined as larger or unexpected results can occur.

3. Viewports can, if needed, overlap on the display surface.

### 10.2 Three dimensional transformations

In three dimensions, the transformations that can be applied are translation, rotation, scaling and shearing as in the two dimensional case. In addition, there is another: the perspective transformation, taking the display from 3-D to 2-D in a way which allows the appearance of 3-D objects to be pictured from a viewpoint.

#### 10.2.1 Translation

To translate the point \((X_{old}, Y_{old}, Z_{old}, 1)\) by amounts, \(X_{mv}, Y_{mv}\) and \(Z_{mv}\), we use the matrix, \(T\):

\[
T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ X_{mv} & Y_{mv} & Z_{mv} & 1 \end{bmatrix}
\]

so that

\[
(X_{new}, Y_{new}, Z_{new}, 1) = (X_{old}, Y_{old}, Z_{old}, 1) * T = (X_{old} + X_{mv}, Y_{old} + Y_{mv}, Z_{old} + Z_{mv}, 1)
\]
10.2.2 Rotation

Rotation by an angle $A$ about the negative $Z$-axis is given by the application of $R_z$.

$$R_z = \begin{bmatrix} C & S & 0 & 0 & 1 \\ -S & C & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where $C=\cos(A)$ and $S=\sin(A)$

Thus, 
$$(X_{new}, Y_{new}, Z_{new}, l) = (X_{old}, Y_{old}, Z_{old}, l) \cdot R_z$$

$$= (X_{old} \cdot C - Y_{old} \cdot S, X_{old} \cdot S + Y_{old} \cdot C, Z_{old}, 1)$$

Rotation about the positive $X$-axis is given by applying $R_x$.

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & C & -S & 0 & 1 \\ 0 & S & C & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus, 
$$(X_{new}, Y_{new}, Z_{new}, l) = (X_{old}, Y_{old}, Z_{old}, l) \cdot R_x$$

$$= (X_{old}, Y_{old} \cdot C + Z_{old} \cdot S, Z_{old} \cdot C - Y_{old} \cdot S, l)$$

Rotation about the positive $Y$-axis is given by applying $R_y$

$$R_y = \begin{bmatrix} C & 0 & S & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ -S & 0 & C & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus, 
$$(X_{new}, Y_{new}, Z_{new}, l) = (X_{old}, Y_{old}, Z_{old}, l) \cdot R_y$$

$$= (X_{old} \cdot C - Z_{old} \cdot S, Y_{old}, X_{old} \cdot S + Z_{old} \cdot C, l)$$

116
When we use these rotations with a positive angle, the following effects occur if we are standing on the negative Z-axis:

RX: the object tilts towards us  
RY: the right hand side of the object swings away  
RZ: the object tilts over in an anticlockwise direction.

10.2.3 Scaling

To scale the position of a point by the amounts Xsc, Ysc, Zsc, we use:

\[
S = \begin{bmatrix}
Xsc & 0 & 0 & 0 \\
0 & Ysc & 0 & 0 \\
0 & 0 & Zsc & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

This gives:

\[
(X_{\text{new}}, Y_{\text{new}}, Z_{\text{new}}, 1) = (X_{\text{old}}, Y_{\text{old}}, Z_{\text{old}}, 1) \cdot S \\
= (X_{\text{old}} \cdot X_{\text{sc}}, Y_{\text{old}} \cdot Y_{\text{sc}}, Z_{\text{old}} \cdot Z_{\text{sc}}, 1)
\]

10.2.4 Perspective

When we want to display objects with some degree of realism the techniques of perspective drawing must be employed. In conventional manual drawings, perspective relies on the correct position of vanishing points - the points from which we must draw all lines that are parallel to given planes (Figure 10.6). In computer graphics, as might be expected, we use a mathematical transformation rather than vanishing points to give perspective. Like the other transformations, the perspective transformation is applied to each vertex of the object we wish to display, but, unlike the others, this transformation converts every point in 3-D space to an appropriate position in 2-D space.
To see how to derive the perspective transformation, we imagine looking at an object through a sheet of glass. A point can be marked on the sheet corresponding to each vertex in the scene (Fig 10.7).
If we assume that our viewpoint is on the negative Z-axis at a distance of D world coordinate units from the origin and that the sheet of glass (technically called the picture plane) is in the X-Y plane at the origin as in Figure (10.8), we note that any point with coordinate, X, in the scene has the coordinate, XP, on the picture plane and that, by similar triangles,

\[ XP = \frac{X \cdot D}{Z + D} \]
\[ = X \cdot \frac{1}{((Z/D) + 1)} \]

The corresponding result for YP is,

\[ YP = \frac{Y \cdot D}{Z + D} \]
\[ = Y \cdot \frac{1}{((Z/D) + 1)} \]

Figure 10.8 Position of point xp calculated by means of similar triangles

Because the picture plane is situated at Z=0, all the coordinates in the scene become zero in the picture. Thus, the X- and Y- positions on the picture plane depend on their X- and Y- positions in space together with a factor composed of the eye distance and the Z-positions in space. So that,

\[(X_{\text{new}}, Y_{\text{new}}, Z_{\text{new}}, l) = (X_{\text{old}} \cdot F, Y_{\text{old}} \cdot F, 0, l)\]

where \(F = \frac{1}{((Z_{\text{old}}/D) + 1)}\).

The matrix formulation of this is:

\[(X_{\text{new}}, Y_{\text{new}}, Z_{\text{new}}, l) = (X_{\text{old}}, Y_{\text{old}}, Z_{\text{old}}, l) \cdot P\]
where \( P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/D \\ 0 & 0 & 0 & 1 \end{bmatrix} \)

Applying \( P \) to \((X_{\text{old}}, Y_{\text{old}}, Z_{\text{old}}, 1)\), we see that we get \((X_{\text{old}}, Y_{\text{old}}, 0, Z_{\text{old}}/D+1)\) but, this is equivalent to \((X_{\text{old}}*F, Y_{\text{old}}*F, 0, 1)\) where \( F \) is as previously defined.

To create a perspective view of an object, the following is performed:
1. Translate the object to the position we want relative to the viewpoint.
2. Rotate the object to present the desired faces.
3. Apply the perspective transformation to the coordinates of the vertices in order to map them onto the picture plane.
4. If needed, scale the result to make the drawing fit the drawing surface.
5. Display the result by connecting the transformed points with the correct lines (clipping as necessary).

We can perform the transformations separately or compound them into a single matrix. However, the translation, rotation and scaling matrices are the same for each object coordinate but the perspective matrix depends on the Z-coordinate value at each point. Thus it is probably more convenient to concatenate the translation and rotation matrices but to apply the scaling and perspective matrices separately.

10.2.5 Z-clipping

If the eye is at \(-D\) on the Z-axis then we can clip to the plane \(Z=-D\). To do this we need to check the Z-coordinates of each line to be drawn. Three possibilities arise as Figure (10.9) shows:

1. If the Z-coordinates of both endpoints are less than or equal to \(-D\), then the whole of the line is behind the eye point and can be ignored.
2. If the Z-coordinates of both endpoints are greater than \(-D\), the whole of the line is in front of the eyepoint and need not be clipped at all.
3. Only if the Z-coordinate of one endpoint is greater than \(-D\) and the other is less than or equal to it, need we perform clipping. In this case, it is sufficient to substitute the value \(Z=-D\) into the equation of the line.
Thus, given the line going from P1:(8,3,-10) to P2:(0,4,10) with the eye at five units along the -Z axis as in fig 10.10, we see that the point P2 is in front. The parametric equations of this line are:

\[
\begin{align*}
X &= 8 + t \cdot (0-8) = 8 - 8t \\
Y &= 3 + t \cdot (4-3) = 3 + t \\
Z &= -10 + t \cdot (10-(-10)) = -10 + 20t
\end{align*}
\]

Then, when \(Z = -5, t = 0.25\), so that, at \(Z = -5\), \(X = 6\) and \(Y = 3.25\). This shows that the clipped line runs from \(PC:(6,3.25,-5)\) to \(P2:(0,4,10)\).
CHAPTER ELEVEN

THE PROGRAM CRYSTAL FOR THE IBM PC
11.1 Introduction

It was seen from a previous study of the design of software for Computer Assisted Learning software, that the graphics capabilities and speeds of modern 386-based PC's are extremely good.

Initially then, it seemed reasonable to explore the possibility of the use of these, relatively inexpensive, machines in the design of software for solid-state modelling. A basic package was thus written and developed in Microsoft QuickBasic, and implemented on an Hewlett Packard 386 based Vectra.

11.2 The program

The code was produced that enabled an FDAT file from the Cambridge Crystallographic database to be read in and the unit cell generated. Thus the program contains a library of space-group symmetry information. This was then developed to allow an extended lattice to be generated.

The speed of the 386-based machines allowed representations to be rotated and translated reasonably smoothly, in real time.

Three model representations are available:
1. wireframe
2. ball and stick
3. red/green stereo image

The display is automatically depth cued, portions of the structure at the rear of the display being rendered in a darker colour than those in the foreground. Smooth animation is obtained by alternating the active and visual pages of video memory. This allows flicker free screen update.

11.3 Program Operation

A file of structures obtained in an FDAT type format from a search of the Cambridge Crystallographic database can be read via a local area network from the Micro Vax II computer (on which the database resides) to the PC. The program can then read entries in the database file sequentially and display the structures as required. On
selection, coordinates, unit-cell data and space group data are read in. A library of space groups is consulted and the appropriate symmetry operators are installed.

Initially the unit-cell is displayed with a wireframe box indicating the boundaries of the unit-cell. (This box can be selected or de-selected by choice). The display can be rotated, in real time, and a choice of ball and stick, wireframe, and red/green stereo view is available. Each can be switched on and off by using the appropriate switching key. On choice a larger portion of the lattice can be calculated and displayed using the symmetry operators.

11.4 Evaluation of software on the 386 PC

The limitations of using a PC based approach to this type of software soon became apparent, since large structures viewed on relatively low resolution (by graphics workstation standards) small screens become very confused and jumbled. The speed of interactive rotation and translation was more than adequate, however when routines were added to allow interactive editing and manipulation, the display speed and resolution soon became a very limiting factor. Thus for serious applications, a superior hardware capability was needed. The software, however, makes a very good teaching package for the display and understanding of crystal chemistry.

11.5 Technical details

11.5.1 Flow diagram

The flow diagram for the software is shown in fig 11.1

11.5.2 Generation of symmetry

Symmetry operators are stored in the form of two matrices:

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>-1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Y</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>Z</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
</tr>
</tbody>
</table>

rotational part       translational part
fig 11.1 Flow diagram
11.5.3 Program Modules

MAIN
Main control routines.

DATAREAD
Reads in unit cell parameters a,b,c,alpha,beta,gamma and the space group.
Creates the coordinates for the unit-cell box.
Reads in asymmetric unit coordinates.
Reads in space group symmetry operators (stored in DATA statements within the program).

CENTERDISP
Centres the structure in the graphics viewport.

ORTHOG
Routine to orthogonalise coordinates.

AUTOBOND
Provides connectivity vectors for the unit-cell box.
Provides connectivities for atoms that are within a specified bond distance.

SYMOP
applies symmetry operator matrices to data

SYMM
Reads in symmetry operator matrices to data

ROTAMOL
main display routine allowing real-time rotation and translation of structures. Produces smooth animation by using a video-page swapping method.
Refer to Volume II for the program listing.
CONCLUSIONS OF CRYSTAL-PC

The use of a 386-based PC provided a reasonably good platform for an initial modelling system, unit cells and even extended lattices could be visualised and rotated and translated in real-time (i.e. the movement occurs smoothly, as you watch, without any time delay). The limitations that become apparent, however, are more to do with the size and resolution of a PC's video display. The application was developed to a stage where the next logical step after implementing visualisation code was the design of interactive routines to modify and edit structures using the mouse. Incorporation of these routines began to show up the hardware limitations. In large structures the display is rather cluttered, making picking and editing confusing and frustrating. Whilst PC molecular modelling packages are perfectly adequate for building and editing smaller molecules, the resolution and size of screen becomes a limiting factor for larger molecules, particularly crystal structures for which a clear idea of depth is necessary for unambiguous visualisation. The inclusion of mouse polling routines for user interaction also caused the application to slow noticeably. This could be overcome with the current hardware, by using faster languages, i.e. 'C' or 386 machine code.

PC design and production is advancing rapidly and a much enhanced application could run on a 80486 50 MHz PC with super-VGA graphics. Auxiliary graphics cards costing about £2000 can give resolutions of 1280 x 1024 whilst also improving speed performance greatly. 17 inch and 19 inch monitors are also available.

For ultimate graphical representations we turn to the new generation of graphics supercomputers.
CHAPTER TWELVE

VISUALISATION TECHNIQUES FOR SOLID STATE MATERIALS DESIGN
12.1 Introduction

The use of computer graphics for visualising results of scientific and engineering applications has spread widely in the last few years. Many fields of study now employ specialised hardware and software for graphical display and simulation of scientific problems e.g. Computer Aided Design, Fluid Dynamics, Material Science, Civil Engineering, Physics, Biology, Biochemistry and Chemistry. The applications although varied in each field, really have the common goal of providing a graphical readily understandable image of scientific data from experiment, theory or simulation. Moreover the techniques provide methods for interacting with the model, simulation or experiment in real time to alter parameters and variables to see the outcome in a readily understandable form without recourse to expensive practical experimentation. A key ingredient towards achieving useful tools is the need to develop software and hardware tools. In particular, software must be efficient, user-friendly and as much as possible, programmed to take advantage of the interactive aspects of the man-machine interface. Therefore optimal programming of highly interactive computer graphics applications, with, if possible, a standard graphics interface, remains as one of the major tasks in advancing the use of visualisation in understanding scientific and engineering research.

12.2 Requirements for Molecular Graphics Applications

For many years experimental and theoretical chemists have used mechanical models to study and understand the structure of molecules particularly larger biological molecules. These models, can be time consuming to build, very bulky (particularly so for proteins & peptides) and can offer no insights into electronic or molecular properties of molecules. The computer offers the advantage of easily buildable structures with the interface to electronic and geometric calculations and simulations.

Moreover recent advances in hardware technology have meant that the sort of representations and displays available are becoming increasingly realistic.(in the sense of looking like solid mechanical models) and easy to view and manipulate offering all the advantages of computational techniques and the clarity of real physical models.

The use of computer graphics in chemistry has been concentrated in several main areas: molecular model building, edit and comparison, investigation of interactions between molecules, protein design and engineering and analysis of chemical and physical properties.\textsuperscript{48-62}

The basic requirement to all computational chemistry applications is to display
and manipulate three dimensional representations of molecular structures and properties associated with these structures. Some of the key issues in designing graphics programs for computational chemistry is that they should be interactive user-friendly and provide a great variety of functions that increase productivity and understanding of underlying scientific problems.

12.3 The User Interface

The most common and logical interface to a graphics programming environment is through the graphics display itself. This in turn points to the necessity of having a user-friendly interface, driven by the use of peripheral input devices such as the mouse, dials and function keys.

12.4 Display of Molecular Models

A common representation of a molecular model is the use of spheres centred at each atomic position and cylinders connecting two atoms joined through the presence of a chemical bond. Other typical representations are space-filling or CPK models and Drieding or stick model representations. Each type of representation has its advantages and the most useful representation is often dependent upon the type of data and the properties being investigated.

12.5 Stick and Ball and Stick Models

The typical ball and stick models are drawn using spheres with a radius of one half the covalent radii of the atoms and a cylinder to represent any bond between two atoms. This type of representation most clearly depicts both the locations of the atomic centres as well as the bonding pattern of the molecule. CPK or space filling models are drawn using spheres centred at the atomic positions, however drawn using the van der Waals radii for the atoms. This model best describes the overall spatial representation of the molecule. Drieding or stick models are constructed by drawing sticks to represent inter-atomic bonds, each half of the bond coloured according to the atom type involved in the bond. This latter model is the simplest graphical representation, and offers the fastest response to interactive manipulations from the user because only polylines are used as output primitives. Rotations and translations of these structures can then be accomplished by inserting modelling transformations within the structure.

Other types of models have also been shown to be quite useful for investigations of chemical structure. Among the models currently used are molecular surfaces or envelopes, electron densities, and more particularly for polypeptides, alpha carbon traces and ribbon models.
12.6 Molecular Surfaces

Molecular surfaces are commonly used in chemistry to illustrate either the spatial representation of a molecule (analogous to the space filling model), or a particular surface property of a molecule. The two most typical surfaces used in chemical applications are the van der Waals surface and the solvent accessible surface. As its name implies, the van der Waals surface is the surface of the 3D object constructed from van der Waals sized spheres at each atomic centre. The solvent accessible surface represents the surface of the molecule that is accessible to solvent molecules that may be surrounding the molecule. This surface is generated by "rolling" a probe molecule and computing the surface around the solute which is accessible to the solvent molecule. It is based on the ideas of Lee and Richards, Richards and Greer and Bush and has been analytically implemented by Connolly. The method computes sample points on the surface which can be subsequently displayed on any display device by displaying each point as a marker on the surface. It should be noted here that in the computation of the Connolly solvent accessible surface, the Van der Waals surface is computed by simply setting the radius of the probe molecule to zero.

Because it is also sometimes desirable to modify molecular structures from their original conformation, neither surface is automatically computed, but should be selected for computation after the proper orientation has been chosen. Once the computation has been performed, the resultant surface can then be displayed or hidden as desired. Molecular surfaces can also be coloured according to particular properties or characteristics, such as atom type, electrostatic potential, hydrophobicity etc..

12.7 Polypeptides

Additional representations are often used in the display of proteins and polypeptides that are very useful in understanding their structure and function. A good deal of structural information can be gained by looking only at the backbone of the polypeptide. This not only simplifies the structure by reducing the number of atoms, but it also focuses attention to the secondary structures such as helices and beta sheets that are commonly found in polypeptide structures.

Among the polypeptide representations that have been popularised are the alpha carbon trace and the ribbon model. The alpha carbon trace is simply a series of lines connecting the alpha carbon of each amino acid in the polypeptide. The ribbon model popularised by Richardson depicts slightly more detailed information on the
backbone structure. The polyline ribbon model is created by tracing a number of line segments along the backbone of each amino acid residue as proposed by Carson.\textsuperscript{59,60} The endpoints of the line segments are computed to be parallel to the C=O bond, and bisecting the C-N bond. In this fashion, when these points are connected a set of parallel lines tracing the positions of the backbone atoms of the amino acids is generated. Any number of segments can be drawn, but a good choice is usually three to five segments separated by about 0.2 Å. Depending upon the number of segments used, usually an equal number of segments are drawn above and below the C-N bond. In order to smooth the ribbon, a spline curve can also be drawn through the points composing the ribbon.\textsuperscript{61}

The second ribbon representation utilizes a filled polygon for each amino acid, the colour being chosen according to the residue type. Each polygon is constructed from the four corners of a rectangle defined by the endpoints of the two outer line segments composing the amino acid. The polygon, like the ribbon, is flat; however, because the polygon is coloured it is much easier to identify amino acid types. With the advent of high-resolution high quality graphics machines this representation may also be of a solid filled tube type, also computed using splines for smoothing effects.

A main advantage of the ribbon model over the alpha carbon trace is that the ribbon offers more information on the twists and turns within the backbone structure of the polypeptide. For example, in regions of alpha helical secondary structure the ribbon clearly depicts the twisting and turning of the helix, while in regions of beta sheets the ribbons are extended and flat.

12.8 Colour Coding

The use of colour in all of the above representations also play an important role in elucidating information about molecular structure. In general, most representations use a simple colouring scheme which assigns a different colour to each type of atom. However, it is also possible to use different colours to highlight different regions of a molecule, or to highlight particular properties.

One example is when considering polypeptides; instead of colouring each atom according to its atom type, it is possible to colour each amino acid residue by a different colour, or even each amino acid residue a different colour according to a hydrophobicity scale. Another example is the colouring of a van der Waals surface
according to the value of the molecular electrostatic potential computed at each point on the surface.

Different users have different preferences for choosing colours, and therefore, while default colouring schemes are generally practical for most applications, it is also useful to have a scheme for allowing users to interactively modify and choose colours.

12.9 Display of Electron Densities and Molecular Orbitals

Two- and three-dimensional contour maps are used quite frequently in all areas of science and engineering. As examples, in the field of computational chemistry they are commonly used to represent iso-energy contours, Ramachandran maps for polypeptides, electron density maps or molecular orbitals. In some of these cases the data can be represented with two-dimensional contours, while in other cases both two- and three-dimensional contours must be used.

12.10 Creation and Manipulation of Molecules

One of the advantages in using an interactive graphics system is the ability to manipulate molecular structures, either in the building process, or by altering already built molecules. By combining computer graphics techniques with geometrical model building, the user is provided with a much simpler and clearer understanding of the relationship between standard geometrical measurements such as bond distances, valence angles and torsions with resulting structures. It is imperative to have a simple method for viewing a molecule from different angles and orientations, to alter parameters and see the resulting structures, or even to build molecules without any prior knowledge of existing coordinates.

The global translation, rotation, scaling and zooming of a molecule is important towards understanding its three-dimensional structure. For many years chemists have been trying to understand the spatial three-dimensional relationships in a molecule by building real models of molecules using plastic balls and sticks. Holding a model and looking at it from different angles and views creates a succession of pictures that creates an image of the true shape of the molecule. A computer graphics program that lets a user rotate and translate a molecule in real time offers these same capabilities, without the problems of the models degrading with time, and with the advantage that the models can usually be easily changed through the use of programming techniques.
Other added advantages to the use of a computer graphics program is that it is relatively simple to program both global transformations for rotation or translation on all molecules on the graphics screen, or to limit the transformations to only one or several particular molecules, keeping the remaining ones fixed. This is useful for investigating chemical phenomena such as interactions of two molecules or docking of molecules. In addition, the data within a computer graphics program can always be made available to the user in numerical format, offering the possibility to change parameters as desired to the accuracy of the program.

In general most molecular structures are expressed either as a set of cartesian coordinates for each atom in the molecule, or as a set of internal coordinates. The internal coordinates of a molecule are usually assumed to consist of bond distances, bond angles and torsion angles. Bond distances are defined as the distance between two bonded atoms. A bond angle is defined as the angle between three consecutively bonded atoms, while the torsion angle is the angle between the two planes defined by four bonded atoms, while an out-of-plane wagging torsion consists of three consecutively bonded atoms with the fourth atom attached to the middle atom of the three.

Internal coordinates offer a more chemically intuitive representation of molecular structure, and have been adopted for use in many programs dealing with computations of molecules. However, the definition of a complete set of internal coordinates to describe the orientations of the atoms within a molecule is not unique and it is possible to introduce redundant coordinates, or to over-specify the definition of the geometry.

Graphics applications require cartesian coordinates to display the molecule. All computational methods require the use of cartesian coordinates, though the input may be specified in internal coordinates. Cartesian coordinates are also the type of data that is usually derived from X-ray crystallography, one of the major experimental techniques for investigating molecular structure.

The transformation between internal and cartesian coordinates is most commonly expressed in terms of a Z matrix. The Z matrix formalism, or similar methods are quite common in many computational chemistry programs. Basically, the Z matrix requires a length, an angle and a torsion value to specify the position of an atom in space. These parameters must be specified in terms of atoms whose positions are already defined. Naturally, this leads to the fact that the first three atoms in the
molecule are defined in a slightly different fashion from the remaining atoms. The first atom is usually chosen to be at the origin, and no parameters are necessary to specify its position. The second atom defines the positive X-axis, and its position is defined with just a distance from the origin, or the first atom. The third atom defines the X-Y plane; a distance from either of the two previous atoms, as well as the angle between all three atoms are needed to define its position.

Using these definitions for the positions of the first three atoms, the remaining atoms can be defined in terms of their relative position to already defined atoms; hence the term internal coordinates. For the purposes of nomenclature we assign the labels I, J, K and L to the four indices defining a Z matrix element. Label I refers to the atom whose position is being defined. The distance from atom I to J is then denoted by the first entry in the Z matrix. The angle between atoms I-J-K is the second entry, and the torsion between atoms I-J-K-L is the third entry.

Given a Z matrix specification, it is possible to alter the internal coordinates of a molecule by changing the proper Z matrix elements. For modification of molecular geometries, this procedure is more chemically intuitive than cartesian coordinates. For example, it is more common for a chemist to understand that a particular bond distance for a C-C bond is too long, than to be able to determine the same information from cartesian coordinates. Interactive manipulation of Z matrix elements, with the corresponding geometrical changes reflected on the graphics screen is one method to manipulate molecular structures, and to immediately determine if the resultant structure is consistent with the expected results.

Z matrix elements to describe the internal geometry of a molecule can be entered by the user in one of two ways. The first way is in a straightforward input file. The second way is more complicated and involves a transformation from user defined cartesian coordinate input to Z matrix format. As previously mentioned, this transformation is not unique, and therefore any general algorithm to build the transformation must rely on its ability to make certain assumptions. The major assumptions that must be made involve the ordering of the atoms and the choice of atoms to define a given atoms length, angle and torsion value, or J, K and L values for a particular I.

The algorithm used makes these assumptions based on the internal connectivity
table that is determined by the program. The first atom in the Z matrix is always assumed to be the first atom in the user input file that is not having its Z matrix elements individually specified. If we then assign each remaining atom in the entry in the Z matrix the second atom will be that atom which has the lowest number and is bonded to the first atom. The third Z matrix entry will then become the lowest numbered atom bound to either the first or second Z matrix entry.

For the remaining N-3 atoms the atom numbers defining the bond length, angle and torsion (J,K and L values) to be used in their Z matrix definition are determined by sequentially searching the connectivity table in ascending order. In this fashion, the majority of Z matrix elements are defined in terms of atoms who appear prior to themselves in the input file. Of course this will not always be the case (it is dependent upon the ordering of atoms in the input file), and in fact, it is possible that a particular Z matrix entry will refer to atom numbers that have not yet been defined in previous Z matrix entries. By the definition of the Z matrix (each atom must be defined in terms of atoms already defined), this particular atom's position cannot be determined. Therefore, this particular element must be placed in a lower entry in the Z matrix, and another element which utilises only previously defined atoms moved into its place. It should be additionally noted here that given the method adopted for assigning the (I,J,K,L) definitions, it may not be possible to replace this element with one whose (I,J,K,L) entries are all defined. In this case it then becomes necessary to modify the (I,J,K,L) entries for a particular atom to ensure that it can be defined. This is done automatically by the program, but as discussed later, can also be interactively specified by the user.

Once a Z matrix transformation has been properly defined, either in an input file or according to the above procedure, this information is passed to an interactive graphics screen which provides the user with the ability to modify Z matrix elements. The resultant changes in geometry are then computed and displayed on the screen along with the original geometry. The user may modify as many or as few of the Z matrix elements as desired. Each modification is reflected on the graphics screen, each time comparing it to the original structure, or simply quit and return to the original structure.

Quite often the manipulation of the internal geometry of a molecule is limited to a small region of the molecule. In this case it is usually preferable to be able to specify a particular definition for some of the (I,J,K,L) values for some of the Z matrix
definitions for certain atoms within the molecule. Therefore, interactive modification and fixing of particular Z matrix elements is allowed. This is accomplished by giving the user the option to specify the definition of the Z matrix elements for a particular atom in terms of the J, K and L values. This definition for this particular atom, I, is then fixed, and the resulting Z matrix elements are computed assuming these fixed value(s).

A modification on the Z matrix technique involves the rotation or stretching of bonds. This type of structure modification is quite common and ideally suited to an interactive environment. The user must choose the bond to rotate around or stretch by clicking on its two constituent atoms with the mouse. A transformation is then applied to define the chosen bond to be along the X-axis. This is a two-step transformation; first a rotation around the Z-axis to align the bond parallel to the X-axis, and then a rotation around the Y-axis to project the bond parallel to the X-axis, and then a rotation around the Y-axis to project the bond into the X-Y plane. A tree search on the connectivity table for the molecule is then done to determine which atoms are connected to the bond and will be modified as a result of the rotation and/or translation. The portion of the molecule to be transformed is then highlighted on the graphics screen and compared to the original structure. A dial is programmed to perform a rotation and/or translation around the new X-axis, thereby effecting a transformation of a portion of the molecule while the remainder stays fixed. If the rotation and/or translation is done around the X-axis, then the corresponding motion is a pure bond rotation or stretching. If the rotation is done around one of the remaining orthogonal axes, either angle-bending (Z-axis) or out-of-plane wagging (Y-axis) is performed.

12.11 Molecule building from templates

The interactive building of molecules from a set of pre-defined templates is one method of constructing molecules without any a priori knowledge of bond distances, valence angles or torsions. It should be emphasised here that the geometries of molecules built in this fashion are by no means definitive, but should only be used to develop a preliminary model for further refinement and study.

A set of templates are predefined, though it is possible for a user to modify, add or delete entries according to their particular applicational requirements.

The construction of molecules using templates is a four-step process that can be
briefly summarised as

1. Choice of template.
2. Choice of position on screen to add template.
3. Joining templates in appropriate positions.
4. Updating all the necessary information, such as coordinates, bond connectivity, etc., to reflect the addition of the template to the molecule.

Templates are presented to the user in a choice of atom, fragment or group libraries. Upon choosing a menu entry the template can be added to the display window by clicking with the mouse. The template can then be joined to an existing structure or manipulated using the appropriate functions.

The transformation necessary to place the template in the proper orientation for the addition to the molecule is analogous to the transformation for rotation about a bond. The local coordinate system of the templates has been defined such that they have one bond directed along the X-axis. This is the bond in the template which is used to join the molecule. Then the transformation to align the rotation about a bond in the molecule along the X-axis is computed in the same fashion as for rotation about a bond. The template can then be added to the molecule by updating all necessary information regarding the atoms in the molecule.

12.12 Enquiry of Geometrical Parameters

A computer graphics environment is ideal for making enquiries towards geometrical parameters of molecular structures. The use of the graphics screen displaying the structure and the immediate feedback of the values of distances and angles provides a user with a complete understanding of the molecular geometry. Interactive selective enquiry is possible by choosing atoms from the screen with the mouse.

12.13 Display of Molecular Vibrations

Molecules are quite often characterised by their spectra, and in particular the infrared and Raman spectra of molecules are commonly used to help identify the presence of chemical groups within a molecule. It has also become a common practice to compute infrared and Raman spectra of molecules using computational methods. A spectrum is usually characterised by a set of intensities and frequencies for the normal
modes of vibration. However, it is also possible to obtain the vectors characterising the displacements of each atom during the vibrational modes. Given this characterisation, it is useful to be able to visualise the normal modes of vibration and identify their principal components (for example C-H stretching, ring breathing or C-H bending).

12.14 Animations

Animation of molecular vibrations or molecular dynamics simulations provides a clear and concise tool for the interpretation of large amounts of numerical data. Each frame in the animation sequence is constructed from a snapshot in time of the positions of the atoms in the molecule. The frames can then be displayed in a time ordered sequence, one after another, to create a movie. The movie is able to illustrate quite clearly the spatial relationships among the atoms in the simulations as they move and respond to the forces acting upon them. However, in addition to providing information on the spatial relationships, the movies can also be used to display the formation and breaking of hydrogen bonds or other weak interatomic forces. For example, it is interesting to show the formation of hydrogen bonded networks or the importance of water as a solvent. The frames can also be displayed superimposed on top of each other. In this way one can easily visualise the amount of displacement that each atom undergoes during the simulation.

The animations discussed here are meant to be interactive, and used during the scientific discovery phase (as opposed to a post-processing or presentation phase). In this sense, the animations need to be generated quickly and easily so that they may be viewed as the simulation is running. Because the time to generate the images, as well as the amount of storage to hold the images, needs to be kept to a minimum, these animations are usually constructed from relatively simple graphics representations such as stick figures. This means that it is possible to have the opportunity to view results and change parameters or input to the simulation as it progresses, which is the main emphasis of these types of movies.

On the other hand, it is possible to render more realistic three-dimensional representations of molecules and to animate these images, but these images can be quite time consuming to generate and the procedure is usually reserved for a post-processing mode of operation.
12.15 Files Manipulation

Another important feature of most graphics packages, which is not specifically designed within the package but is more of a by-product of the nature of the program, is the ability to manage and manipulate a variety of file formats. The great variety of computational chemistry programs, as well as the different number of ways to provide input to these programs makes the creation, manipulation and maintenance of input and output files an arduous task. However, the need to display and manipulate data graphically, both before and after processing by a simulation package, is found in all areas of computational chemistry and in many ways provides a unifying medium across all disciplines. Since most users of graphics packages will also be users of several different simulation programs, it is necessary to design flexible input and output (in the form of saving data that has been graphically manipulated and modified). Standard input and output format from the most commonly used programs within a particular research environment should be accepted as input to a computer graphics program without the need for modification by a user. Output from the graphics program should be in the input format for the simulation programs, requiring little if any intervention by the user to run the input. While some of the issues may seem somewhat trivial, they are nevertheless important towards providing a user-friendly programming and research environment.

12.16 Towards More Realistic Images

To this point, most of our discussions have been centred on providing a highly interactive user-friendly environment for computer graphics. At a certain point it also becomes necessary to think about how to provide realistic three-dimensional computer graphics representations of three-dimensional objects on the two-dimensional screen. Several techniques are used to provide as much realism to these two-dimensional images as possible. Among them are:

1. Hidden line removal
2. Hidden surface removal.
3. Stereoscopic images.
4. Depth cueing.
5. Ray tracing.

Several of these techniques can be implemented in an interactive fashion through the use of specialised hardware or through the use of efficient software algorithms. Of course, the former provides higher performance, but the latter has the
advantage of being more general, flexible, and applicable to a large variety of both 'intelligent' and 'non-intelligent' display devices.

Hidden surfaces and depth cueing can be efficiently implemented using a depth sorting algorithm. A depth sorting algorithm works both in the object space as well as the image space, and consists of two parts:

1. Surfaces or objects must be ordered in decreasing depth.
2. Surfaces or objects are scan converted, starting with the surface of greatest depth and working forward.

The first step is always carried out in the object space, or the actual coordinate system of the objects in question. The second step is carried out in image space and involves the display of pixels on the graphics screen surface.

The method for hidden surface removal is sometimes referred to as the painters algorithm, as it is similar to the method that an artist uses to paint a picture. In the first pass, the background is painted on the screen. In the subsequent steps, the surface with the greatest depth is compared with the other surfaces in the list to determine if there are overlaps with closer surfaces. If no overlaps are present, then the surface can be painted on the screen. If there are overlaps, then additional comparisons must be made to determine whether any of the surfaces must become hidden.

Of course, the algorithm cannot deal with only entire surfaces, but is complicated by the fact that it must be able to handle portions of surfaces. However, the algorithm can be modified for molecular structures and made to be quite efficient. If each atom is treated as a sphere and if the atoms are drawn in order of increasing depth value, then the hidden surfaces are automatically removed, and no surface comparisons are necessary. It should be noted here that using this technique, as rotational transformations are performed, the atoms must be reordered and redrawn for each transformation. The key components of this algorithm then become a fast sorting algorithm coupled with an efficient procedure for drawing spheres or circles at atomic positions.

This algorithm can be easily modified for an efficient implementation of intensity or depth cueing. Again, a two-step procedure is necessary:
1. Objects must be sorted in order of decreasing depth.
2. Objects are drawn from the greatest depth and moving forward, assigning colour to each object based on its depth value.

The use of stereoscopic images is also quite common in molecular graphics applications. It involves the display of two slightly rotated views simultaneously, where the views are rotated between 5-8 degrees to account for the separation between the two eyes of a person. By focusing each eye only on its corresponding view and then superimposing the two views on top of each other, a view with three-dimensional perspective is created. Often, it is useful to use stereoscopic viewers to decouple the eyes and to force each eye only to see its corresponding view.

Ray tracing techniques\(^{64}\) are more time-consuming than any of the techniques previously described, but these images are by far the most realistic and include the effects of shading, shadows, transparency and reflections. The realistic appearance of these images can be a valuable aid in the perception of the three-dimensional shape and the spatial distribution of the objects represented on a two-dimensional graphics screen. This is especially true for large, complicated molecular systems, or in molecular dynamics simulations composed of many independent molecules where the molecules can appear to float in space without any apparent connections to each other or other points in space. In these cases, the shadows and reflections of the molecules on one another can provide visual clues to the three-dimensional relationship between molecules.

The ray tracing technique can be briefly summarized\(^{64}\) by recalling that a graphics image can be considered to be composed of a large number of dots, each assigned a particular colour based on the surface characteristics of the object, lighting characteristics of the screen, etc. Of course, the greater the number of dots (or pixels) the greater the resolution of the image. In principle an infinite number of pixels could be generated over the various surfaces in an image; in practice this number is usually limited to around 1,000,000 pixels. In the ray tracing technique, the colour of each pixel is determined by tracing rays backwards from the viewing position through the pixel and back to the light source.

Starting from the viewing position a ray is passed through each pixel in the viewing plane and then traced back into the objects defined in the image. The ray can then either strike one of the objects in the scene or pass through the scene. If the ray
strikes an object, this ray can then be reflected and refracted, generating several new child rays. The child rays are then traced until the rays end at a light source or pass out of the scene. The intensity of the corresponding pixel is then set by determining all of the contributions from the parent and all the child rays. Usually the number of generations of child rays that can be generated are limited because after several generations the contributions become negligible. However, the amount of computation is still quite large and requires considerable time.

12.17 Definition of the Image

An image is defined by three types of information:

1. The geometry of the scene.
2. The optical characteristics of the objects in the scene.
3. How the scene is illuminated and viewed.

The data required to specify this information are described in the following sections.

Geometry

The geometry of the scene is defined by the size, shape, and position of each object in the scene. The shape of an object is determined by one or more basic geometric entities called primitives. The ray tracing program used in this work, which is specialised for molecular modelling applications, recognises only one type of primitive: the sphere.

The geometry of the scene is specified by defining each object within the scene. Each object is defined by specifying the type of primitive (i.e., sphere), the characteristic parameters (radius and centre), and the optical characteristics of the object.

12.18 Optical Characteristics

The optical characteristics of each object are defined by the ability of the object to transmit and reflect light. These characteristics are determined by a number of factors including:

1. Transparency coefficients.
2. Coefficients of diffuse reflection.
3. Coefficients of specular reflection.

The effects of these factors are modelled by a set of corresponding parameters which are defined below in the section on the evaluation of picture elements. Each of these factors depends on the wavelength of the transmitted or reflected light, and the wavelength dependence is responsible for the observed colours associated with each object.

If a surface is opaque, the transparency coefficients are all zero, and no refracted rays are produced by the surface. Otherwise, the surface may produce both reflected and refracted rays. In principle, the index of refraction also depends on the wavelength of the refracted light. However, for simplicity we assume that the index of refraction does not depend on wavelength.

12.19 Viewing and Illumination

The image produced for a particular scene depends on a number of conditions beyond the definitions of the objects in the scene. These include the location of the observer, the direction of viewing, the size of the field of view, the number of lights, and the location, colour, intensity and concentration of each of the lights.

The image also depends on the number of distinct picture elements (pixels) used to cover the field of view, and the treatment of aliasing. Aliasing includes a number of artifacts, including stair steps on diagonal lines, caused by using a single point to determine the colour of a pixel. Measures used to reduce or eliminate aliasing are called anti-aliasing. The anti-aliasing measures used in the current algorithm are described below:

12.20 Calculation of Ray Trajectories

Ray trajectories are calculated by following lines from the observer to the objects in the scene and from the objects to the sources of illumination. Each ray is defined by a point of origin and a unit vector which specifies the direction of the ray. A set of one or more rays is calculated for each element (pixel) of the image being rendered. These pixels form a rectangular grid on a plane perpendicular to a line from the eye point of the viewer to the "gaze point" which defines the direction of view.

The first ray calculated for each pixel always has its origin at the eye point of
the viewer. The direction of this ray is defined by the line from the eye point through
the centre of the pixel. Extension of this line through the scene leads to one of the
following results:

1. The ray misses the entire scene.
2. The ray passes through the scene, but does not intersect any object within the
   scene.
3. The ray intersects an opaque object.
4. The ray intersects a transparent object.

If the ray misses the scene (ray 'a' in figure 12.1), it is marked as a "background ray,
and the ray trajectory is finished. If the ray passes through the scene, it must be tested
to determine whether it hits any objects within the scene. If the ray passes through the
scene without hitting any objects(ray 'b' in figure 12.1), it is also marked as a
"background ray," and the ray trajectory is finished.

If the ray hits one or more objects, the intersection point closest to the origin of
the ray is recorded, and one or two new rays are created. The new rays each originate
at the intersection point marking the end of the previous ray. If the object is not
opaque, one of the new rays will be either a refracted ray or a total internal reflection
ray, depending on the index of refraction and the angle at which the ray intersects the
surface. If the object is opaque , or the first ray is a refracted ray, a reflected ray is
created. Each of these rays is then analysed in the same manner until the maximum of
generations of rays have been generated. The first three generations of rays arising
from one initial ray are shown in Figure 12.2
12.21 Evaluation of Picture Elements

The contribution of each pixel to an image is defined by the red, green, and blue intensity values of the pixel. These values are determined by the intensities of the corresponding components of one or more rays starting at the viewer and passing through the pixel. In the absence of anti-aliasing, the values for each pixel are determined by a single ray passing through the centre of the pixel. With anti-aliasing, each pixel is divided into sub-pixels, as indicated in the following section.

If a ray starts from the viewer and passes through the scene without striking any objects, then the red, blue, and green intensity values of the ray are determined by the background values of the scene. Otherwise, the intensity values of a ray are determined by an illumination model similar to those proposed by Whitted and Phong. In this case, the intensity of each component (red, green, and blue) is given by:
\[ I(v) = I_d(v) + I_r(v) + I_a \]

where

- \( I_d \) is the contribution from the direct illumination.
- \( I_r \) is the contribution from indirect (reflected or refracted) illumination.
- \( I_a \) is the contribution from ambient illumination.
- \( v \) is a unit vector in the direction of the ray.

**Fig. 12.2** Generation of rays from an initial ray. The three generations of rays include ray (0) as the initial ray, ray (1) as a reflection of ray (0), rays (2a) and (2b) as reflections and refractions of ray (1), and rays (3a), (3b) and (3c) as reflections of rays (2a) and (2b), and a refraction of ray (2b), respectively.
12.22 Direct Illumination

The contribution of direct illumination is given by:

\[ I_d(v) = \sum [k_d(n \cdot L_i) + k_s(v \cdot L'_i)^{n_s}] \frac{I_i}{d_i^2} \]

where:

- \( n_i \) is the number of sources of illumination for which there is a direct path to the point where this ray strikes the scene.
- \( I_i \) is the intensity of the \( i \)-th source.
- \( d_i \) is the distance from the \( i \)-th source to the point where this ray strikes the scene.
- \( K_d \) is the coefficient of diffuse reflection.
- \( k_s \) is the coefficient of specular reflection.
- \( n_s \) is the specular exponent for the surface of the object where the ray strikes the scene.
- \( v \) is a unit vector in the direction of the ray.
- \( L_i \) is a unit vector pointing from the \( i \)-th source to the point where this ray strikes the scene.
- \( L'_i \) is the (reverse) reflection of \( L_i \) on \( n \cdot [2n(n \cdot L_i) - L_i] \). The relationships between the vectors \( v, L_i, \) and \( L'_i \) are shown in fig 9. The values of \( k_d, k_s \), and \( I_i \) depend on the colour (red, green, and blue) of the ray. The value of \( n_s \) is determined by the texture of the surface, and does not depend on the colour of the ray.
Light which may travel from a source of illumination to the surface of an object by a path through a transparent surface is ignored because it is very difficult to determine the path(s) of such illumination. Consequently, all transparent objects cast shadows, and any surface in the interior of a transparent object will be dark unless a source of light is placed inside the transparent object.

12.23 Indirect Illumination

The contribution of indirect illumination includes the effects of reflection and refraction. If a ray strikes an opaque object, then this contribution is given by:

\[ I_1(v) = k_s I(v') \]

where:
- \( k_s \) is the coefficient of specular reflection.
- \( v \) is a unit vector in the direction of the ray.
- \( v' \) is the reflection of \( v \) on \( n \), that is \( v = 2n(n \cdot v) \).
- \( n \) is a unit vector perpendicular to the surface where the ray strikes the scene.

The relationships between these vectors are shown in figure 12.3.

![Figure 12.3](image)

Fig 12.3 Geometry of the intersection of a ray with a surface with normal vector. The vector \( v \) points away from the viewer or the previous surface, \( v' \) indicates the reflected ray, and \( v'' \) indicates the refracted ray.
If the object is not opaque, and total internal reflection does not occur, then the intensity of indirect illumination is given by

\[ I_r(v) = k_t I(v') + K_t I(v'') \]

where:
- \( k_t \) is the transmission coefficient.
- \( v'' \) is a unit vector in the direction of a refracted ray.

If total internal reflection occurs, then the intensity of indirect illumination is given by:

\[ I_r(v) = 0.98I(v') \]

Each of these contributions is determined by generating one or two new rays propagating in directions \( v' \) and/or \( v'' \) determined by optical physics and the geometry of the intersection of the parent ray with the scene. If either of these rays strikes another object, the intensity \( I(v') \) or \( I(v'') \) is calculated in the same manner used to determine the intensity of the parent ray, \( I(v) \). This process is repeated recursively until either the reflected and refracted rays all leave the scene, or the maximum number of generations of new rays are generated.

If a reflected ray does not intersect an object within the scene, it does not contribute to the colour of the parent ray. If a refracted ray does not intersect an object in the scene, it is assigned the background colour. This allows the background to be seen through transparent objects, but prevents the background from being reflected on the surfaces of objects.

### 12.24 Ambient Illumination

Ambient illumination is a type of indirect illumination represented by a soft glow uniformly pervading all objects in the scene. The contribution of ambient illumination to each colour (red, green, and blue) is determined by the product of the ambient light intensity and the coefficient of reflection for ambient light. The ambient light intensity is a property of the entire scene, and the coefficient of reflection for ambient light is a property of the surface at the point of intersection. This value is independent of the orientation of the surface and the direction of the ray.
The principal effects of ambient illumination are to soften shadows and to provide a minimum level of illumination to all visible portions of the scene. Excessive levels of ambient illumination tend to reduce the contrast between light and dark portions of each object, making curved surfaces appear flat. The level of ambient illumination should be kept very low to preserve the three-dimensional appearance of curved surfaces such as spheres.

12.25 Adaptive Anti-Aliasing

Anti-aliasing is the process used to eliminate the jagged or stair-step appearance of diagonal lines and the edges of curved surfaces. In the absence of anti-aliasing, one ray passes through the centre of each pixel, and the colour determined for this ray is assigned to the entire pixel. This causes abrupt changes in colour near the edges of objects, and produces the stair-step effect seen at diagonal boundaries.

The stair-step effect (aliasing) may be minimised by dividing each picture element (pixel) into sub-pixels. The colour of each sub-pixel is calculated by sending a ray through the centre of the sub-pixel. The average of the colours determined for the sub-pixels is then assigned to the entire pixel. As a result, the colour of each pixel represents a composite of the various objects which can be seen within the pixel. Anti-aliasing is most important at the edges of objects and other places where the colours of adjacent pixels change abruptly or rapidly. Anti-aliasing is not required in portions of an image where the colours of adjacent pixels are constant (as in the background), or slowly varying.

The anti-aliasing algorithm can be briefly summarised as follows. Each pixel is compared to its eight adjacent pixels. These nine pixels form a 3x3 array with the current pixel at its centre. If the red, green, or blue colour values of the current pixel differ from the corresponding values of any of the neighbouring pixels by more than a fixed threshold, then the current pixel is divided into four sub-pixels and the colour of each sub-pixel is calculated. If the colour of any of the four sub-pixels differs from the original colour of the current pixel by more than a fixed threshold, then each sub-pixel is subdivided into four smaller sub-pixels, and the colour of each sub-pixel is replaced by the average of the four smaller sub-pixels. The colour of the current pixel is then replaced by the average of the first four sub-pixels.
12.26 Animations of Realistic Images

The animation of realistic three-dimensional images provides the advantage over the interactive animations previously described in that the higher quality images enable one to more easily interpret the complex spatial motions of the atoms and molecules. In addition, these animations, if recorded on videotape or some other media, provide a medium in which it is possible to exchange and present results at conferences and symposia. However, the production of computer animated videos requires special hardware and often special software.
CHAPTER THIRTEEN

GRAPHICS PROGRAMING USING DORÉ
Graphics programming using Dore'

13.1 Introduction

Dore' is a powerful graphics library that enables the production of dynamic image sequences and near-photographic quality images. With the Dore' library, full colour, high-resolution, three-dimensional images can be combined with computationally intensive supercomputer applications.

Dore', which stands for Dynamic Object Rendering Environment, provides a comprehensive set of tools for creating graphics, including:

i) primitives, such as polygons and patches, for representing objects. Advanced primitives including polygonal meshes, closed cubic surfaces, and non-uniform rational beta-spline surfaces.

ii) Surface properties, such as ambient, diffuse, and specular light reflectance. Other surface properties available in Dore' include transparency, shadows, and environmental reflection.

iii) features that enable the description of a graphics "scene" that includes the objects, as well as the lights that illuminate them and the cameras used to view them.

iv) rendering representations, including points, wireframe faceted, and smooth-shaded surface types, and combinations of styles in the same scene.

v) A wide array of functions that enable the editing of the graphics database.

Dore' programming is object-based, an approach which leads to programs that are both modular and easily extensible.

13.2 Basic steps in Programming with the Dore' library.

The programmer;

1) Creates the objects being modeled using primitives for points, lines, polygons, patches, and surfaces.
2) Chooses attributes to describe the appearance of the objects— their colour, shininess, whether they have shadows on their surfaces, and so on.

3) Positions and sizes scene objects relative to one another.

4) Positions various kinds of lights in relation to the scene.

5) Selects a camera lens and positions the camera in relation to the scene.

6) Asks for the scene to be rendered.

13.3 Objects and Methods.

The Dore' library consists of objects and methods. All data is kept in the form of objects. Each geometric primitive, attribute, group, camera, light, device, frame, and view is a separate object. Each object has a set of methods which are a set of internal functions that operate on the object. When a method is invoked, the Dore' database is traversed, and the objects in the database are executed using that method. The most commonly used method is rendering. Other methods involve picking, computing bounding volumes, and printing.

Objects are the basic building blocks of the Dore' library. An object is a collection of data and a set of methods (functions) that operate on the data. Certain kinds of objects, such as primitive objects, represent three-dimensional shapes (cylinders, spheres, toruses, polygons) that can be displayed. Other objects, called primitive attribute objects, affect how primitive objects look—for example, their colouring, how they respond to light, whether they can have shadows and so on. Geometric transformation objects are a type of attribute object that affect the shape and positioning of primitive objects in three-dimensional space. Primitive objects and primitive attribute objects all fall into a general category called display objects. As their name suggests, display objects expect to be rendered and displayed. If certain conditions are met (they are not invisible, they are not obscured by other objects), the primitive objects will be able to be viewed in some form, and the effects of their associated attribute objects will be visible on some form.
13.4 Groups and In-line groups.

A group is a Dore’ object that contains an ordered list of object handles. A group itself is also an object. Once an object or a group of objects has been defined, it can be referred to any number of times in a particular scene database. For example, a bolt shape could be modeled using a group containing polygons and a cylinder and then that bolt could be used a number of times in a single car wheel group. Then it would be possible to reference the basic wheel group four times in a complete car object.

An example of a simple group is shown in fig 13.1, a solid magenta sphere.

![fig 13.1 Sphere Group](image)

A more complicated group for a car wheel is shown in fig 13.2.

![fig 13.2 Basic wheel group](image)
Left and right car wheels can now be easily created using this "wheel primitive" group, with modifications for the placement and rotation of each wheel as shown in fig 13.3. To make the wheels turn together, an identical rotate object could be put above each instance of wheel group, but it would be better to build into the axle definition the fact that the left and right wheels move in unison; that is, we want to move both wheels with one edit. One way to do that is dynamically change the rotate object at the top of the wheel group. The problem with that approach is that it changes the definition of what a wheel is, thereby affecting all wheels in the database.

fig 13.3 Left and right wheel group

The correct solution involves the use of an in-line group. An in-line group acts as if its elements were actually part of its parent's group. Attribute values set in an in-line group affect subsequent objects in the parent group. A useful way to think about
the two types of groups is that a regular group is like a subroutine, whereas an in-line group is like a macro. Usually in-line groups contain only attribute objects.

Fig 13.4 shows an in-line group that is referenced by both the left and right wheel groups. The in-line group contains a rotation object, which causes the wheels to move as a pair, and a scale object, which widens and sizes the wheels. An axle group has been added to the example to connect the two wheels.

fig 13.4 Using an in-line group for the scale and rotation objects

13.5 Callback groups.

There are certain conditional elements that can be used to affect execution of the Dore' database. DoCallback<DOCB> causes Dore' to call out to user-written functions during execution and to execute the current method on all elements referenced in the callback function. (via DsExecuteObj<DSEO>).

155
The DoCallback<DOCB> function allows the shape of the definition or display tree into which it is added to be affected. DoCallback<DOCB> takes a pointer to a user-written function and a pointer to user data. When DoCallback<DOCB> is executed, the user function is called and is passed the data. If the callback is used as a conditional, the user function can call DsExecuteObj<DSEO> with an object. Whatever method was executed on the callback function is executed on any such objects. For example, if the rendering method were executed on the callback function, the rendering method will be executed on the objects executed by the callback function.

13.6 Display groups & studio groups.

The display group in the Dore' database is used to contain all the models to be displayed. The studio-group is used to contain all elements which describe the scene such as cameras, lights and other related objects.

13.7 Producer-group.

The producer-group was devised as an addition to the standard Dore' functions. This was necessary to contain more than one studio-group for the multi storage databank function.

13.8 The picking functionality in Dore'

Picking is a method that identifies the drawable primitive objects that are contained in a specific volume (the pick aperture) of a Dore' device. In molecular graphics applications, atoms need to be picked and identified using the mouse, for particular functions.

The picking method is triggered by the Dore' function DdPickObjs, which returns information about what it finds in the pick aperture. During picking, geometric attribute objects are executed just as they are during rendering. Other attribute objects, such as those relating to colour, highlights and shadows, have no effect during picking. All coordinate transformations are performed, and primitive objects are decomposed into points, lines, and triangles, which are then transformed into their final values in x, y and z device coordinates.

If any part of the object is contained within the pick aperture, and if the current pickability switch attribute is on, the primitive is considered to be hit.
For this application, the pick aperture was taken to be the entire graphics window, and pickability was restricted to the closest hit for sphere and label primitives.
CHAPTER FOURTEEN

CRYSTAL - SOLID STATE MODELLING SOFTWARE
FOR THE STARDENT TITAN GRAPHICS SUPERCOMPUTER
14.1 Introduction

Crystal is a software package developed to run on a Stardent Titan Graphics Supercomputer and is intended to be the interactive editing, building and visualising part of the solid-state design package proposed in chapter ten. It is a very large piece of software (currently 15,245 lines of C code) and has been designed on a modular basis so that new functions can be added when required and so that it can eventually be interfaced to the artificial intelligence system currently being developed.
Table 14.1 General Features

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-friendly interface.</td>
</tr>
<tr>
<td>2</td>
<td>Interface to Cambridge Crystallographic Database.</td>
</tr>
<tr>
<td>3</td>
<td>Rotation, translation and lighting of models via keyboard or dial-box.</td>
</tr>
<tr>
<td>4</td>
<td>Files manipulation I/O - supports the following formats</td>
</tr>
<tr>
<td></td>
<td>a) Biograf</td>
</tr>
<tr>
<td></td>
<td>b) Cambridge FDAT</td>
</tr>
<tr>
<td></td>
<td>c) COSMIC &amp; CHEMX .XR files</td>
</tr>
<tr>
<td>5</td>
<td>Model Representations</td>
</tr>
<tr>
<td></td>
<td>a) wireframe</td>
</tr>
<tr>
<td></td>
<td>b) ball &amp; wire</td>
</tr>
<tr>
<td></td>
<td>c) solid space-filled</td>
</tr>
<tr>
<td></td>
<td>d) solid ball &amp; stick</td>
</tr>
<tr>
<td></td>
<td>e) red/green stereo</td>
</tr>
<tr>
<td></td>
<td>f) dot surfaces</td>
</tr>
<tr>
<td></td>
<td>g) ribbons for protein</td>
</tr>
<tr>
<td></td>
<td>&amp; peptide display.</td>
</tr>
<tr>
<td>6</td>
<td>Display parameters</td>
</tr>
<tr>
<td></td>
<td>a) levelled depth cueing</td>
</tr>
<tr>
<td></td>
<td>b) levelled light intensity</td>
</tr>
<tr>
<td></td>
<td>c) three levels of resolution</td>
</tr>
<tr>
<td></td>
<td>d) two types of surface shading</td>
</tr>
<tr>
<td></td>
<td>e) wireframe, dot, surface display types</td>
</tr>
<tr>
<td></td>
<td>f) switchable backface-culling</td>
</tr>
<tr>
<td></td>
<td>g) switchable Highlights</td>
</tr>
<tr>
<td></td>
<td>h) variable background colours</td>
</tr>
<tr>
<td>7</td>
<td>Generation of unit cells from Cambridge data</td>
</tr>
<tr>
<td>8</td>
<td>Generation of extended lattices from Cambridge data</td>
</tr>
<tr>
<td>9</td>
<td>Geometry Calculations</td>
</tr>
<tr>
<td></td>
<td>a) angles</td>
</tr>
<tr>
<td></td>
<td>b) bonds</td>
</tr>
<tr>
<td></td>
<td>c) torsions</td>
</tr>
<tr>
<td>10</td>
<td>Energy calculations</td>
</tr>
<tr>
<td>11</td>
<td>Interactive manipulation and modification of structures</td>
</tr>
<tr>
<td>12</td>
<td>Adjustment of unit cell parameters</td>
</tr>
<tr>
<td>13</td>
<td>Molecular Editor</td>
</tr>
<tr>
<td>14</td>
<td>Interface to Molecular Mechanics packages</td>
</tr>
<tr>
<td></td>
<td>a) MM2</td>
</tr>
<tr>
<td></td>
<td>b) MOLMECH</td>
</tr>
<tr>
<td>15</td>
<td>Interface to QM packages</td>
</tr>
<tr>
<td></td>
<td>a) MOPAC 5.0 e.s.p.</td>
</tr>
</tbody>
</table>

159
Databank utility for multiple on-screen images
**GENERAL FLOW DIAGRAM**

1. **Reader.c**
   - Reads input arguments passed to the program by the user.
   - Initializes Dore and sets any input arguments passed to the program.
   - Reads the file containing the dialog and initialization data.
   - Initializes the display window. Set-up the devices from which input is expected. Initializes the Dore environment.

2. **Clear studio-group, model-group & text-groups**
   - **make-models**
     - Creates the displayable object model and places it in a group called `model_group` for better recall.
   
3. **geom-spec**
   - Reads necessary data files and specifies the geometry of the models to be displayed. Each model is placed in a separate group which is referenced as `models[i]`. Creates a name label `text-group`.
   - **readatomdata**
     - Reads atoms and atomcolour data from atomdata file.
   - **displaymolecules**
   - **generates models**
   - **readmoleculecam**
     - Reads molecule data from the Cambridge FDAT FILE.
   - **ORTHOG**
     - Orthogonalizes coordinate data.
   - **typeconvert**
     - Converts FDAT atom types to 'CRYSTAL' atom types.
   - **autobond**
     - Calculates atom distances and determines connectivity.

4. **do-lateract**
   - Accept and process interactive input from the keyboard, dialog, or whatever, to dynamically manipulate the model. This is done using the Unix "poll" function to determine which of the devices has input waiting. Do some last minute initialization.
   - Post the initial scene to the display device. Enter an infinite loop, waiting for input from any of the devices declared in do-late.
   - **do-dial(keyboard)**
     - Check to see if a pick has occurred in the main DORE window.
     - If update flag=1 exit else update the current display.
   - If not exiting continue at **while not exit loop**
   - Exit the Dore environment cleanly.
<table>
<thead>
<tr>
<th>MODULE</th>
<th>PURPOSE</th>
<th>PARENT FILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>addhyd</td>
<td>Add hydrogen atoms to unfilled valencies</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>assatom</td>
<td>Assign selected atomtypes to chosen atoms</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>anglec</td>
<td>Do angle calculation</td>
<td>geomdist.c</td>
</tr>
<tr>
<td>atomcon</td>
<td>Stores the atom picked from which to proceed with the next connection.</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>atominsert</td>
<td>Do void volume search calculation. Steps through the unit cell layer by</td>
<td>Render.c</td>
</tr>
<tr>
<td></td>
<td>layer checking to see if probe atom clashes with another atom. If it</td>
<td></td>
</tr>
<tr>
<td></td>
<td>doesn't it has found a void where it can fit. Writes this void to file</td>
<td></td>
</tr>
<tr>
<td>autobond</td>
<td>Calculate distances between atoms and assume connectivity</td>
<td>autobond.c</td>
</tr>
<tr>
<td>bgfcopy</td>
<td>Copies the temporary storage biograf file to a named permanent biograf</td>
<td>bgfcopy.c</td>
</tr>
<tr>
<td></td>
<td>file</td>
<td></td>
</tr>
<tr>
<td>biowrite</td>
<td>Writes a Biograf format output file</td>
<td>biowrite.c</td>
</tr>
<tr>
<td>butt xevent</td>
<td>Find out which button pressed and redraw button in new state</td>
<td>butt.c</td>
</tr>
<tr>
<td>create colour</td>
<td>Create the colour lookup table</td>
<td>Render.c</td>
</tr>
<tr>
<td>table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>copiesurf</td>
<td>Copies the temporary storage CONTACT.SURF file to a named permanent</td>
<td>bgfcopy.c</td>
</tr>
<tr>
<td></td>
<td>.dot file</td>
<td></td>
</tr>
<tr>
<td>copytofrag</td>
<td>Copies data from temporary bgf files to fragment arrays.</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>databank</td>
<td>Create the 'databank' callback function to execute the appropriate</td>
<td>Render.c</td>
</tr>
<tr>
<td>callback</td>
<td>model group in the display group.</td>
<td></td>
</tr>
<tr>
<td>default_parse</td>
<td>Parse input commands not already parsed by the application specific</td>
<td>Render.c</td>
</tr>
<tr>
<td></td>
<td>user_parse function</td>
<td></td>
</tr>
<tr>
<td>delatom</td>
<td>Delete picked atom</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>delfrag</td>
<td>Delete picked fragment</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>displaymolecules</td>
<td>generates the model types</td>
<td>geom_spec.c</td>
</tr>
<tr>
<td>distcalc</td>
<td>Calculate distance between atom1 and atom2</td>
<td>geomdist.c</td>
</tr>
<tr>
<td>do_calpha</td>
<td>Generate model of alpha carbon backbone in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>do_dials</td>
<td>Read incoming data from the dialbox using the passed function (read_dils)</td>
<td>render.c</td>
</tr>
<tr>
<td></td>
<td>and propagate it via a call to update_dial</td>
<td></td>
</tr>
<tr>
<td>do_init</td>
<td>Initialise Dore and get any input arguments passed to this program by the</td>
<td>render.c</td>
</tr>
<tr>
<td></td>
<td>user</td>
<td></td>
</tr>
<tr>
<td>do_interact</td>
<td>Accept and process interactive input from the keyboard, dialbox etc to</td>
<td>render.c</td>
</tr>
<tr>
<td></td>
<td>&quot;dynamically&quot; manipulate the model. This is done using the Unix &quot;poll&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>function to determine which of the devices has input waiting to be</td>
<td></td>
</tr>
<tr>
<td></td>
<td>processed.</td>
<td></td>
</tr>
<tr>
<td>do_main</td>
<td>Generates the main chain atoms in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>do_metals</td>
<td>Generates metals in model in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>do_sidechains</td>
<td>Generates a model of only the sidechain atoms in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>do_stick</td>
<td>Generates the stick model in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>do_tf_rotx</td>
<td>Set the x global rotation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>do_tf_roty</td>
<td>Set the y global rotation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>do_tf_rotx</td>
<td>Set the z global rotation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>do_transx</td>
<td>Set the x translation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>do_trany</td>
<td>Set the y translation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>do_tranz</td>
<td>Set the z translation value for the model object</td>
<td>render.c</td>
</tr>
<tr>
<td>draw_all_buttons</td>
<td>Fill the entire window with a solid colour, then draw the buttons with</td>
<td>Butt.c</td>
</tr>
<tr>
<td></td>
<td>their associated text</td>
<td></td>
</tr>
<tr>
<td>draw_button</td>
<td>Place a button icon in an Xwindow and label it with a string</td>
<td>Butt.c</td>
</tr>
<tr>
<td>draw_knob</td>
<td>Read the dial icon data from a disk file and display it at the</td>
<td>Knobs.c</td>
</tr>
<tr>
<td></td>
<td>appropriate location in the specified Xwindow.</td>
<td></td>
</tr>
<tr>
<td>draw_mode_init</td>
<td>Initialises molecular editor mode</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>File</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>draw_mode_main</td>
<td>Directs program flow to the selected molecular editor function.</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>draw_text</td>
<td>Draw a single text string in an X window</td>
<td>Butt.c</td>
</tr>
<tr>
<td>energy</td>
<td>Energy calculation module</td>
<td>energy.c</td>
</tr>
<tr>
<td>fforward</td>
<td>Reads through the Cambridge Fdat file until the current record number, held in the variable recno is reached.</td>
<td>camread.c</td>
</tr>
<tr>
<td>flat_ribbon</td>
<td>Generates a flat ribbon model in proteins</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>fragbiowrite</td>
<td>Writes a biograf format file for fragments</td>
<td>fragbiowrite.c</td>
</tr>
<tr>
<td>fragwrite</td>
<td>Writes .frg file from the fragments array</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>gen_splines</td>
<td>Function to generate the guide points for the splines. This then invokes CubicSpline to create the splines.</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>geomdist</td>
<td>Calculate molecule geometries and orientations</td>
<td>geomdist.c</td>
</tr>
<tr>
<td>geom_spec</td>
<td>Reads any necessary data files and specifies the geometry of any models to be displayed. Each distinct model is placed in a separate group referenced as 'models[i]'</td>
<td>geom_spec.c</td>
</tr>
<tr>
<td>geprot_spec</td>
<td>Specifies the geometry of the protein/peptide models.</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>getcamatms</td>
<td>Gets the coordinates and atom_types from a Cambridge Fdat entry</td>
<td>camread.c</td>
</tr>
<tr>
<td>getcamcell</td>
<td>Gets the unit_cell data from a Cambridge Fdat entry</td>
<td>camread.c</td>
</tr>
<tr>
<td>getcamname</td>
<td>Gets the name entry in a Cambridge Fdat file</td>
<td>camread.c</td>
</tr>
<tr>
<td>getcamsgrp</td>
<td>Gets the space_group data from a Cambridge Fdat entry</td>
<td>camread.c</td>
</tr>
<tr>
<td>getcon</td>
<td>Gets all the connectivities for an atom id.</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>getfrag</td>
<td>Stores the identity and number of atoms in a fragment containing atom i.d. This data may then be used for further operations, e.g. deletion of the fragment, rotation or translation of the fragment.</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>graphics_expansion</td>
<td>Checks the graphics Sbus status register to determine whether or not there is a graphics expansion board installed in the system.</td>
<td>Render.c</td>
</tr>
<tr>
<td>hydropathy_score</td>
<td>Calculate the hydropathy index score for amino acids</td>
<td>geprot_spec.c</td>
</tr>
<tr>
<td>index</td>
<td>Given a character string, find the location of a substring within it</td>
<td>Render.c</td>
</tr>
<tr>
<td>init_buttons</td>
<td>Initialises the buttons and draws them into a window</td>
<td>Butt.c</td>
</tr>
<tr>
<td>init_knobs</td>
<td>Draw the dials with their corresponding value on the user interface screen</td>
<td>Knobs.c</td>
</tr>
<tr>
<td>init_window</td>
<td>Initialise and display an X window of the specified type &amp; size</td>
<td>Ui.c</td>
</tr>
<tr>
<td>join</td>
<td>Joins two atoms</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>knob_xevent</td>
<td>Gets the location of the mouse and determines which knob it corresponds to</td>
<td>Knobs.c</td>
</tr>
<tr>
<td>licintens_callback</td>
<td>Creates the 'licintens' callback function to change intensity of lights in the scene</td>
<td>Render.c</td>
</tr>
<tr>
<td>main</td>
<td>This is the main routine. It initialises the Dore environment, creates the model and studio objects, displays the objects, conditionally regenerates model objects, allows user interaction with the objects, exits the Dore/X-windows environment and the program cleanly.</td>
<td>Render.c</td>
</tr>
<tr>
<td>make_cookies</td>
<td>Create the mouse cursors to be used by the user interface</td>
<td>Ui.c</td>
</tr>
<tr>
<td>make_models</td>
<td>Creates the displayable object model and places it in a group object for later recall. Also, sets up the viewing parameters for the model.</td>
<td>MkObjects.c</td>
</tr>
<tr>
<td>make_model_group</td>
<td>Define the makeup of the model itself and places them all in one group for easy reference later</td>
<td>MkModels.c</td>
</tr>
<tr>
<td>make_studios</td>
<td>Define the &quot;studio&quot; environment (lights, cameras,...) and place it in a group object for later recall</td>
<td>MkStudios.c</td>
</tr>
<tr>
<td>model_callback</td>
<td>Create the 'model' callback function to execute the appropriate Dore model object</td>
<td>Render.c</td>
</tr>
<tr>
<td>modelsize</td>
<td>Compute the bounding volume &amp; origin of the model and set global viewing parameters based on that</td>
<td>MkModels.c</td>
</tr>
<tr>
<td>mvas_init</td>
<td>Initialise the Lyon-Lamb mvas for VTR recording</td>
<td>Minivas.c</td>
</tr>
<tr>
<td>nextcomp</td>
<td>Finds the next compound entry in a Cambridge Fdat file by reading in records until a #COMPOUND NAME is found</td>
<td>camread.c</td>
</tr>
<tr>
<td>open_dials</td>
<td>Open the dial box device line (for reading and writing) and initialise the dial box attached to that line</td>
<td>dials.c</td>
</tr>
<tr>
<td>orthog</td>
<td>Orthogonalise coordinates from Cambridge Fdat file</td>
<td>crystal.c</td>
</tr>
<tr>
<td>PaintInit</td>
<td>Draw the explanation window at the bottom of the user-interface screen</td>
<td>ui_init.c</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>File</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>parent-window</td>
<td>Get the id of the parent X window from which this program was started</td>
<td>Ui.c</td>
</tr>
<tr>
<td>parse geometry</td>
<td>Extracts the location and width of the window to be opened</td>
<td>Render.c</td>
</tr>
<tr>
<td>parse_input</td>
<td>Parse the string passed on starting the program</td>
<td>Render.c</td>
</tr>
<tr>
<td>picking</td>
<td>Set up the pickable aperture in the graphics window</td>
<td>picking.c</td>
</tr>
<tr>
<td>printhelp</td>
<td>Lists the default keyboard commands to the xterm window that the program is being run from</td>
<td>render.c</td>
</tr>
<tr>
<td>producer_callback</td>
<td>Creates the 'producer' callback function to execute the appropriate studio_group in the producer_group</td>
<td>Render.c</td>
</tr>
<tr>
<td>prsarg_get_keyword_int</td>
<td>Search the command line for a keyword field followed by an integer</td>
<td>ParseArg.c</td>
</tr>
<tr>
<td>prsarg_get_keyword_real</td>
<td>Search the command line for a keyword field followed by a floating point number</td>
<td>ParseArg.c</td>
</tr>
<tr>
<td>prsarg_get_keyword_short</td>
<td>Search the command line for a keyword field followed by a short integer</td>
<td>ParseArg.c</td>
</tr>
<tr>
<td>prsarg_get_keyword_string</td>
<td>Search the command line for a keyword field followed by a character string</td>
<td>ParseArg.c</td>
</tr>
<tr>
<td>prsarg_get_keyword_switch</td>
<td>Search the command line for a keyword field followed by a 0/1 type switch</td>
<td>ParseArg.c</td>
</tr>
<tr>
<td>qnwrite</td>
<td>Generates an input file for MOPAC</td>
<td>Render.c</td>
</tr>
<tr>
<td>raytrace</td>
<td>Generates a ray-traced view of the current scene to a file</td>
<td>Render.c</td>
</tr>
<tr>
<td>readatm</td>
<td>Reads in atoms fragments and groups from .atm .grp &amp; .frg files</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>readatomdata</td>
<td>Reads the atom descriptions from the file 'atomdata'.</td>
<td>geom_spec.c</td>
</tr>
<tr>
<td>read_dials</td>
<td>Reads from the dial box device until all of the queued up data has been read. Returns the result of the read</td>
<td>dials.c</td>
</tr>
<tr>
<td>read_dialfile</td>
<td>Reads the dialbox data file which describes how to interpret all subsequent dial movement</td>
<td>Render.c</td>
</tr>
<tr>
<td>read_icon</td>
<td>Read in icon pixmap data from disk &amp; link it with an Xwindow 'image' for later display in an Xwindow.</td>
<td>Butt.c</td>
</tr>
<tr>
<td>read_input</td>
<td>Read incoming \n terminated command from an input device, calling parse_input with the result.</td>
<td>Render.c</td>
</tr>
<tr>
<td>readmoleculecam</td>
<td>Read in compound data from Cambridge Fdat file at next compound position</td>
<td>geom_spec.c</td>
</tr>
<tr>
<td>redo_knobs</td>
<td>Re-display knobs when active knob set is changed</td>
<td>Knobs.c</td>
</tr>
<tr>
<td>rmhvd</td>
<td>Remove hydrogen atoms and replace with unfilled valencies</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>rotate</td>
<td>Rotates a fragment</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>rotx_callback</td>
<td>Sets the x-rotation for object in 'modelgroup'</td>
<td>Render.c</td>
</tr>
<tr>
<td>roty_callback</td>
<td>Sets the y-rotation for object in 'modelgroup'</td>
<td>Render.c</td>
</tr>
<tr>
<td>rotz_callback</td>
<td>Sets the z-rotation for object in 'modelgroup'</td>
<td>Render.c</td>
</tr>
<tr>
<td>stereo_callback</td>
<td>Creates the 'stereo' callback function to turn stereo on/off and set the eye separation</td>
<td>Render.c</td>
</tr>
<tr>
<td>stop</td>
<td>Stop running the program</td>
<td>Stop.c</td>
</tr>
<tr>
<td>symm</td>
<td>Calculates the symmetry matrices for unit-cell calculations</td>
<td>unit_cell.c</td>
</tr>
<tr>
<td>tord</td>
<td>Do torsion angle calculation</td>
<td>geomdist.c</td>
</tr>
<tr>
<td>translate</td>
<td>Translate asymmetric atoms in unit cell, then regenerate unit cell</td>
<td>drawmode.c</td>
</tr>
<tr>
<td>tube_ribbon</td>
<td>Generates a tube ribbon model for proteins.</td>
<td>geoprot_spec.c</td>
</tr>
<tr>
<td>typeconvert</td>
<td>Converts Cambridge atomtypes to 'Crystal' atomtypes</td>
<td>camread.c</td>
</tr>
<tr>
<td>ul_exit</td>
<td>Clean up the user interface before exiting the program</td>
<td>Ui.c</td>
</tr>
<tr>
<td>Ul_init</td>
<td>Initialise the user-interface. If the user-interface cannot be successfully initialised this routine will do an exit(1)</td>
<td>Ul.c</td>
</tr>
<tr>
<td>ui_read_X</td>
<td>Parcel out the handling of X events based on what they are and what window they occurred in</td>
<td>Ul.c</td>
</tr>
<tr>
<td>ul_update_dial</td>
<td>Update the screen representation of the dial that has just been moved, including the red dial position marker and the text string indicating the value associated with the dial.</td>
<td>Knobs.c</td>
</tr>
<tr>
<td>update_callback</td>
<td>Keep track of which update is in progress</td>
<td>Render.c</td>
</tr>
<tr>
<td>update_dials</td>
<td>Update the dials</td>
<td>Render.c</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td>File</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>update_display</td>
<td>Update the current view</td>
<td>Render.c</td>
</tr>
<tr>
<td>user_parse</td>
<td>Give the user first crack at parsing the string that was passed to the</td>
<td>user_parse.c</td>
</tr>
<tr>
<td></td>
<td>program from the keyboard, or as a result of input from the mouse buttons</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or the dial box</td>
<td></td>
</tr>
<tr>
<td>user_xevent</td>
<td>Give the user first crack at parsing X window events from the Dore window.</td>
<td>user_parse.c</td>
</tr>
<tr>
<td>X_init</td>
<td>Grab the Xwindows display device</td>
<td>Ui.c</td>
</tr>
<tr>
<td>zoom_callback</td>
<td>Create the 'zoom' callback function to change the field-of-view angle to</td>
<td>render.c</td>
</tr>
<tr>
<td></td>
<td>effect zooming</td>
<td></td>
</tr>
</tbody>
</table>
14.1.1 User-friendly interface

Crystal has an iconised menu and dial box display system, in order to make interaction with the programs functions easy and efficient. Plate 14.1 shows the interface. The top right shows the hierarchical menu display. The user friendliness of the interface is greatly enhanced by the ability to use a graphical environment consisting of a number of text and graphics windows. Titan uses a version of windows called X-windows, which acts as a way of organising the input and output to an application via user defined viewports or windows. With X-windows it is easy to work with software applications that have both textual and graphical input and output by choosing the window that you require to interact with. Windows may be moved or resized as required. The menu elements are picked with the mouse, calling either the required function or a sub menu if needed. The top level menu can be returned to at any time by clicking on the required button icon. Menus or sub-menus are split into sets of ten functions. If a menu or sub-menu has more than ten functions the next set of ten functions can be displayed by clicking on the right hand arrow of the <--> button set A--> icon, thus installing button set B. Then clicking on the right hand arrow of the <--> button set B--> icon will install button set C, (The next ten functions). Clicking on the left hand side of the <--> button set B--> icon will reinstall button set A.

The dial icons correspond to the functions altered by altering the dial settings. These functions are x,y & z rotation, x,y & z translation, zoom, and degree of depth cue. If the depth cue option is switched off, a new set of icons is installed, identical to the previous set, except for the Depth icon being replaced by a light intensity icon.

Textual data is input and displayed via the blue text window, shown in the bottom centre of plate 14.1. All the program functions can be operated by textual commands, as a short-cut alternative to using the menus. Typing ? or Help produces a list of available commands in the textual window.

The rest of the display is used for the graphics window for model displays. Several editing functions allowing the picking of atoms and fragments in the graphics window with the mouse.

The stop icon is selected, with the mouse, for finishing a CRYSTAL session.
14.1.2 Interface to the Cambridge database.

The software can search a Cambridge FDAT format data file displaying asymmetric unit data in the graphics window. The Cambridge REF-code is also displayed. see Plate 14.2

The software fully decodes all the information contained in a database entry. Typical textual output is shown in fig 14.1

Plate 14.2 Tetrakis-mu-anthracene-9-carboxylato-bis(1,2-dimethoxyethane)-chromium(ii) Cambridge database entry.
fig 14.1 Sample decoded textual output for the Tetrakis-μ-anthracene-9-carboxylato-
bis(1,2-dimethoxyethane)-chromium(II) Cambridge database entry.

Refcode = #ANTXCR
The crystal system is monoclinic
Structural category = 3
Accession date = 780615
No. of symmetry operators = 2
No. of atoms = 38
No. of symmetry-generated atoms = 38
No. of reported bond lengths = 0
No. of crystallographic connectivity integers = 106
Cell record present
Intensity data measurement = diffractometer
Centre of symmetry at origin
Errors in entry still not resolved!
No problem referral to author
Entry is disordered, a few coordinates may have been removed
Year of publication = 1978
a = 17.500 esd = 0.004
b = 16.362 esd = 0.003
\( c = 8.835 \) esd = 0.010
alpha = 90.00 esd = 0.000
beta = 94.44 esd = 0.010
gamma = 90.0 esd = 0.000
\( D_m = 0 \) \( D_x = 142 \)
Space group number = 14
Space group = P21/n
\( Z \)-value = \( Z \) formula units per unit cell.
Tolerance value = 0.400
R = 0.0505
no. of atoms = 35
no. of non-symmetry atoms = 16

14.1.3 Rotation, translation and lighting of models via keyboard or dial-box

The dial box can be used to control x, y & z rotation, x,y & z translation, zoom
and degree of depth cue. If depth cue is deselected the dial box can control lighting
intensity. All of these functions can, if desired, also be controlled via the keyboard
using textual input.
E.g.

\[
\text{Drx value} \quad \text{rotates the model about the x axis by value degrees.}
\]

\[
\text{Dlvalue} \quad \text{alters the light intensity by the amount value.}
\]

A complete list of keyboard commands is given in Appendix 1 in Volume II (User guide, programmers guide, technical reference and software listing)

14.1.4 Files manipulation I/O

An important feature of graphics packages as mentioned in Chapter 13.15, is the ability to manage and manipulate a variety of file formats.

i) Biograf format - Biograf is a commercial bio-polymer modelling package produced by Biosym Inc.  

ii) Cambridge Database FDAT format.

iii) COSMIC .XR format - Cosmic is an in house package developed by Smith-Kline-Beecham Ltd.

iv) CHEMX - Chemx is a modelling package developed by Chemical Design Ltd.

Examples of these file formats are shown in Appendix 1 Vol II

It is possible to generate unit-cells and lattices from data contained within a Biograf format file by the use of additional keywords, space-group and unit cell information appended to the biograf file.

i.e.

\[
\text{SPAC <space-group no.><space-group>}
\]

\[
\text{UCELL <no.of symmetry operators,a,b,c,alpha,beta,gamma>}
\]

(Note: this is an extension to the Biograf file format)
14.1.5 Model Representations

Eight model representations have been implemented:

i) wireframe; Plate 14.3

This is a split bond wireframe model, the colour is split along the bond according to atom type at a position determined by the Van der Waals radius of the atom.

Plate 14.3 Wire-frame representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)

ii) ball & wire; Plate 14.4

This model type is used whenever atom picking functions are used, due to the balance struck between clarity of atom position and speed of rotation and translation.

iii) solid space-filled; Plate 14.5

This model provides a solid Van der Waals representation.
iv) solid ball and stick; Plate 14.6
This model comprises a solid model with solid filled cylinders for the sticks.

v) red/green stereo; Plate 14.7
Provides a three-dimensional stereo model when viewed through red/green stereo glasses.

vi) dot-surfaces; Plates 14.8
Two dot surface representations are implemented
a) Van der Waals - Michael Connolys surface program is used to generate Van der Waals surfaces from which a dot surface model is constructed.
b) Electro-static potential - The Mopac 5.0 e.s.p. program can be used to generate an electrostatic potential map from which a dot surface model is constructed.

Plate 14.4 Small ball & wire representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)
Plate 14.5 Solid space-filled representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)

Plate 14.6 Solid ball & stick representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)
Plate 14.7 Red/green stereo representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)

Plate 14.8 Van der Waals dot surface representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)
vii) Ribbons for protein/peptide display; plates 14.9-14.14
A number of protein/peptide and ribbon models are implemented.

a) alpha-carbon trace; Plate 14.9
Consists of a coloured line trace of the alpha-carbon backbone.

b) backbone trace; Plate 14.10
Consists of a grey line trace of the carbon backbone.

c) mainchain
Consists of a wireframe model of the protein mainchain.

d) sidechains
Consists of a wireframe model of the protein side-chains.

e) stick model; Plates 14.12,14.14
Consists of a wireframe model of the entire protein/peptide

f) Backbone ribbon; Plate 14.11
Consists of a smooth shaded flat ribbon for the protein backbone.

g) Residue ribbon; Plate 14.12
Consists of a solid filled tubular ribbon colour code according to amino-acid residue types.

h) Hydropathy ribbon; Plates 14.13,14.14 & 2e
Consists of a solid-filled tubular ribbon colour coded according to a hydropathy index for each amino-acid.

iix) label models; Plates 14.15-14.17
Three label models have been implemented for atomtypes, atomnumbers and charge values.
Plate 14.9 Alpha carbon trace of Human Carbonic Anhydrase (II)

Plate 14.10 Backbone trace of Human Carbonic Anhydrase (II)
Plate 14.11 Backbone ribbon representation of the peptide Crambin

Plate 14.12 Residue ribbon representation of the peptide Crambin
Plate 14.13 Hydropathy ribbon representation of Human Carbonic Anhydrase(II)

Plate 14.14 Hydropathy ribbon & stick representation of Human Carbonic Anhydrase(II)
Plate 14.15 Atom label representation of a unit cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)

Plate 14.16 Atom number representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)
Plate 14.17 Charge label representation of a unit-cell of bis(Dihydro-bis(1-pyrazolyl)borato) chromium (ii)

5.1.6 Global display parameters

Depth cueing, described in section 12.16, can be altered via the dial box, as can the global light intensity.

Surfaces can be viewed in low, medium or high resolution. Whilst the high-resolution mode is obviously the most visually impressive, it can be slow in updating for translation and rotation operations on very large structures. Medium and low resolutions improve this update speed.

Two types of surface shading can be selected, and three types of global representation types are available. If dot is selected all surfaces are represented by dots. Wireframe produces wireframe surfaces and the surface option produces filled surfaces.

Backface culling (12.16) can be selected or deselected as can surface highlights which are very effective for space-filled models. The background colour to the display can be chosen from a number of colours.
14.1.7 Generation of unit-cells from Cambridge data; Plates 14.18, 14.19

The program uses Unit-cell data and space-group information to generate Unit cells.

Plate 14.18 Tetrakis-mu-anthracene-9-carboxylato-bis(1,2-dimethoxyethane)-chromium(ii). A generated unit cell

Plate 14.19 A solid ball & stick representation of YBaCu Oxide Superconductor
14.1.8 Generation of extended lattices from Cambridge data: Plate 14.20, 14.21

The program also uses unit-cell data and space-group information to generate extended lattices. The lattice may be generated by any amount, measured in unit-cells in any axis direction. Plate 14.20 shows a lattice of tetraakis-mu-anthracene-9-carboxylato-bis(1,2-dimethoxyethane)-chromium(II), consisting of two unit-cells, the basic unit-cell having been extended by an extra unit cell in the x-positive direction. Plate 14.21 shows the same unit-cell having been extended by 1/2 a unit-cell in every direction.


14.1.9 Geometry calculations

Bond lengths, angles and torsions can be calculated and displayed by picking the appropriate menu buttons and picking the required atoms, in the graphics window, with the mouse.

14.1.10 Energy calculations

This module employs the force-field equation of the program MOLMECH (See P51) to provide an energy estimate. This module is to be extended to calculate crystal lattice energies.

14.1.11 Interactive manipulation and modification of structures

Structures can be built, modified and edited with the functions within the molecular editor (14.1.3). Plate 14.23 shows the manipulation and rotation of part of the unit-cell. The package also allows unit-cell parameters to be interactively modified and manipulated (14.1.12).

14.1.12 Adjustment of unit-cell parameters

Unit-cells can be stretched or shrunk in the a, b or c cell direction and alpha, beta and gamma angles can be changed, to see the effect on the unit-cell. An energy monitor is used to show the energetic effect of these operations (see 14.2.3).

14.1.13 The molecular editor

The package incorporates a fully functional molecular editor of the template building type (12.11). Template fragments can be selected from a library of fragments, groups, atoms or ligands, and can be joined and edited as required. A full description of the molecular editor functions is given in Appendix 1 Volume II. See Plate 14.24
14.1.14 Interface to Molecular Mechanics packages

Interfaces to molecular mechanics programs is implemented for structural refinements, both to the MM2 package and to the program MOLMECH described in chapter 5.

Plate 14.24 The molecular editor

14.1.15 Interface to Quantum Mechanics packages

Access to the semi-empirical quantum mechanical package MOPAC 5.0 E.S.P is implemented with an easy to use menu-driven interface for the many MOPAC options available. Output may be printed, and a dot surface model can be generated to show the electrostatic potential.
14.1.16 Databank facility; Plate 14.25

Multiple models may be displayed simultaneously on the screen for comparison using the packages database functions. Different models can be stored in each databank and can be called back to the screen to be displayed simultaneously or separately as required, or can be combined into one databank. This is extremely useful for superimposing images of related or modified structures for comparison.

14.1.17 Interface to Michael Connolly’s surfacing program

The program is interfaced to Michael Connolly’s program for generating molecular surfaces.
14.2 New functions

14.2.1 Fragment library

Using the molecular editor, ligands from the Cambridge database can be extracted and stored in a template library for use in future designs.

14.2.2 Cell slicing

In order to provide a tool to visualise packing arrangements in complex unit-cells, a cell-slicing utility was developed. This allows unit cells to be sliced by a plane allowing separate layers to be more closely studied. This enables a clearer picture of the packing arrangements and the free space within the lattice. See Plates 14.26 & 14.27.

Plate 14.26 The cell-slicing utility
14.2.3 Interactive modification of unit-cells

This utility allows the changing of unit-cell parameters in order to see the effect on the structure. A fragment in the unit-cell can be picked, rotated or translated and the symmetry related positions are updated. A simple energy monitor consisting of a Van der Waals term and an electrostatic term is used to give a guide to the energetic effect of these modification. Unit-cell a, b, c, alpha, beta and gamma parameters can be modified to see the effects on the structure.

In Plate 14.28 the propane-di-ammonium cation in the chlorochromate structure has been replaced by a tri-ethane-triammonium cation. The unit cell was then drawn out in the b-direction and the correct structure for the di-ethane-triammonium chlorochromate(II) is generated when the energy monitor reaches a minimum. Thus it is possible to make simple predictions on closely related structures just by inspection, therefore this is a useful tool to have.
14.2.4 X-ray diffraction simulation

This module, currently being developed, will enable modifications to be matched to practical physical data. The X-ray diffraction pattern of a structure will be calculated and displayed. Any subsequent modification to the structure, interactively will generate a new diffraction pattern. Thus linking theoretical speculations with practical knowledge.
14.2.5 Void volume searching

This utility searches for volumes where a 'probe atom' of specified radius could fit. It systematically scans the unit-cell and places a pink 'atom' at every position that the probe atom could fit without clashing with any other atom. These pink 'atoms' can then be displayed as a separate model, enabling 'empty' regions of a certain volume to be easily displayed see Plate 14.29. This is an important tool for packing considerations.
CONCLUSIONS

CRYSTAL (PC)

An inorganic visualisation package was designed and implemented for a PC environment. The package makes an excellent teaching tool for the exploration of crystal chemistry and symmetry in crystals. Future developments could include the graphical representation of symmetry operators, such as mirror planes, in the display. The software provides the framework for the development for more powerful and interactive software for enhanced PC platforms, i.e. those containing subsidiary processors with greatly enhanced graphics and performance. With the rapid advance of computer technology such systems will soon be able to supply the quality of visualisation, at the moment only achievable by graphics supercomputers such as the TITAN.

Molmech

Molmech is a complete molecular mechanics system, which includes functions that give it the ability to model systems containing metals, such as metalloproteins and metal complexes. It has been integrated into the TITAN CRYSTAL system, which also provides a platform on which to visualise both metalloproteins and complex inorganic systems. It is amenable to modification allowing, in future, the use of new potential functions to be explored. Future work could include the optimising of the code to run in PC environments and therefore the integration of the code into the CRYSTAL PC system.

Molecular Mechanics and crystal structure prediction

Molecular mechanics is an important and useful technique for molecular simulation. For proteins it is the method of choice for energy investigations. The use of an empirical
potential function for metal centres allows the inclusion of metals in metalloprotein investigations, a situation obviously preferable to omitting the metal from the calculation. The empirical function is not defined on any theoretical basis, but is chosen to reproduce empirically based evidence of preferred geometries. The molecular mechanics technique itself is based on empirical force constant data, and so the inclusion of such a function is reasonably justified. The molecular mechanics program MOLMECH has been successfully used in metalloprotein studies, and the code is amenable to modification as required. In turning the technique of molecular mechanics to inorganic compounds, the problem of general all encompassing parameters is encountered. The technique is useful for conformational studies in metal complexes if parameters are correctly derived. However, serious limitations are encountered in trying to use the technique as a tool for structural predictions in solid state systems.

The prediction of the crystal structure of a compound given no more than its molecular structure has become one of the major goals of structural chemistry. The two major strategies for structure prediction and design, namely computation and statistical inference are now firmly established. The number of structures now available in the Cambridge database is sufficiently large for statistical techniques to be meaningful. In addition to hydrogen bonding, much is now known concerning interactions such as C-H...O, C-H...N, halogen...halogen, halogen...O, halogen...S, S...S, S...N, and S...O. These weak interactions, however, usually define only the secondary structure, which manifests as molecular motifs with particular geometrical properties: rods, ribbons, sheets and helixes. The extrapolation of these secondary motifs into tertiary structures is not predictable in a general sense. It is the tertiary rather than the secondary structure which usually governs many physical or chemical properties sought after in crystal engineering.

The molecular basis of structure

The general assumption in structural chemistry is that, ultimately all chemical information concerning an organic or inorganic compound is encoded in its molecular...
structure, which can be used to predict its crystal structure. Quantum mechanics is able to describe in detail the electronic orbitals and the energy states of the simplest of all chemical systems, the hydrogen atom. With more complicated atoms and with even the simplest chemical molecules its methods are no longer so precise; the complexity of the calculations becomes formidable, and only approximate methods can be used. For complex molecules and crystals detailed calculations are impossible, at least in practice.

The structures of the molecules and the atomic arrangements within crystals can be found out empirically, by chemical and crystallographic methods; these structures may be more or less predictable on the basis of empirical laws. But this is a very different matter from providing a fundamental explanation of chemical structures by means of the Schrödinger wave equation.

It is important to realise this severe limitation of quantum mechanics. Certainly it helps to provide a qualitative or semi-qualitative understanding of chemical bonds and of certain aspects of crystals, such as the difference between insulators and electrical conductors. But it has not enabled the forms of even simple molecules and crystals to be predicted from first principles. In the words of Linus Pauling:

"We may believe the theoretical physicist who tells us that all the properties of substances should be calculable by known methods—the solution of the Schrödinger equation. In fact, however, we have seen that during the 30 years since the Schrödinger equation was discovered only a few accurate non-empirical quantum-mechanical calculations of the properties of substances in which the chemist is interested have been made. The chemist must still rely upon experiment for most of his information about the properties of substances."

Although a further 20 years have passed since this passage was published, and although there have been important improvements in the approximate methods of calculation available to quantum chemists, the situation remains essentially the same today.
The reasons for the difficulty, if not impossibility, of predicting the form of a complex chemical structure on the basis of the properties of its constituent atoms can perhaps be understood more clearly by means of the following illustration. Consider elementary building blocks which can be added to each other one at a time either endways or sideways (fig 15.1). With two building blocks there are \(2^2 = 4\) possible combinations; with three, \(2^3 = 8\); with four, \(2^4 = 16\); with five, \(2^5 = 32\); with ten, \(2^{10} = 1,024\); with twenty \(2^{20} = 1,048,576\); with thirty, \(2^{30} = 1,073,741,824\) and so on. The number of possibilities soon becomes enormous. The situation for a complex chemical system is described in chapter 9. What seems remarkable is that for a complex crystal system the number of potential energy minima with very similar energies would be expected to be quite large, however very definite arrangements are preferred. Polymorphs are indeed encountered but the common number of them for a crystal structure is rarely more than five.

Although we cannot carry out the detailed calculations necessary to predict the minimum energy structures or structures of a system a priori, we are able to use various approximate methods in combination with empirical data on the structures of similar substances. In general, these calculations do not permit unique structures to be predicted (except for the simplest of systems), but only a range of possible structures with more or less similar energies. Thus we can only assume that, if the structure is formed purely on energetic grounds, the approximate methods of calculation are not taking into account subtle energy effects. It is unlikely, therefore, that approximate techniques such as molecular mechanics or atom-atom potentials can predict the preferred minimum energy structures unambiguously.
Figure 16.1 Possible combinations of different numbers of building blocks capable of being joined together either endways or sideways.
Visualisation

Visualisation of results of scientific work is a broad field and the project encompassed such activities as interactive computer graphics, algorithms for realistic rendering of three dimensional computer images, animation of time dependent phenomena, manipulation and creation of models and data for computation, and enhancing the friendliness of the user-to-machine interface. All of these components are important towards advancing the scientific discovery process, and are invaluable tools in performing computer simulation. Visualisation and the ability to interactively manipulate and view complex crystal systems is an important step towards the formation of empirical rules for structure predictions.

CRYSTAL (TITAN)

The program crystal is a fully functional visualisation system allowing the real-time viewing and manipulation of both organic and complex inorganic structures. The system was coded so as to allow maximum interaction with the user, by interfacing the application to the mouse, keyboard and dial-box, and through a software interface incorporating both text and graphics windows. A molecular editor was created incorporating a large fragment and ligand library, allowing interactive editing and manipulation of structures. A number of visualisation enhancements and tools were implemented and previously designed molecular mechanics code was integrated into the system.

The programming was done in such a way as to be highly modular (The modules are described on P161-164 and complete program listings and programers guide are included in Vol II) and, as such, provides a solid framework on which to build further tools and code to aid in materials design and provides a significant move towards the system proposed on P106.

Computer programming is an extremely time intensive process. The provision of
software tools such as the Dore' graphics library greatly assists the developer in producing final applications, however the programming commitment is still vast. Software companies have attempted to help overcome this situation by providing a number of visualisation application development tools such as Stardent computers AVS (Advanced Visualisation System). Such programs eliminate the need for programming by allowing applications to be built from pre-existing modules that can be linked together by selecting a group of modules and establishing connections between them. The hope is that in many cases, users can construct an entire visualisation application, using standard modules and without resorting to traditional procedural programming. On investigation, however, it was found that these applications do not offer the flexibility needed to produce a system that provides the required degree of interaction and accessibility to specific graphics functions. The future may well see a step further towards better fifth generation (application generation programs that reduce the need for programming) tools that remove the need for such intensive programming. Until then much time and very much effort is still needed in the design and coding of useful software systems.
REFERENCES

26. G.W.Harris, J.C.A.Boeyens, N.J.Coville, Organometallics, 4, 1985, 914
35b. M.A.Barber, Ph.D. Thesis, University of Surrey, 1979
37. Cridle & Ellis, Spectral & Chemical characterisation of organic compounds.
41. Cambridge Crystallographic Data Centre, University Chemical Laboratory, Lensfield Road, Cambridge CB2 1EW.
44. Brown et al.
47. J. Maddox, editorial in *Nature*, 325, 201, 1988
52. S. Chin, D. P. Vercauteren, Proc of the 4th Int. Conf. on Supercomputing, C.A., 1989
62. P. Quarenden, Proc of the 4th Int. Conf. on Supercomputing, C.A., 1989
76. P. Wright, Brookhaven Protein Databank, 1ZNF
77. Summes, South, Kim, Hare, Brookhaven Protein Databank, 2ZNF