Appendix B

Visualisation of Atomic Scale Simulations

Originally the text in this appendix was written in HTML and the pictures and diagrams in this reports could be accessed with a computer mouse, to illustrate an atomic scale process via animations. Additionally, the references in the text in electronic format were links to either the biography or WWW (World Wide Web) sites. In this text, all references from the original report to paper documents or WWW-sites are listed in a biography. The original report is solely available in the form of a CD-ROM disc, which can be viewed on any modern personal computer with a CD-ROM drive and a HTML browser such as Netscape or Internet Explorer.

The computer programs developed and packaged for this project were supplied on the original CD-ROM disc but are now also available via the web at "http://abulafia.mt.ic.ac.uk/penicillin".

B.1 Introduction

The visualisation of atomic scale phenomena resulted from the need to understand the spatial distribution of atoms in complex molecules, such as enzymes and proteins, and in the crystal structures of inorganic solids. Initially, this resulted in the building of physical ball and stick models. However, the results of computational chemistry and physics are not always easily transformed into a physical model, particularly when the process being modelled is dynamic.

B.1.1 Graphical Modelling programs

The advent of an affordable desktop computer (PC) with a graphical display led to the development of a variety of software packages based on building a structure atom by atom. Usually these packages allowed the user to specify atom colours, radii and positions, and resulted in a 3-dimensional representation of the molecular structure. Recently, much more sophisticated commercial packages have become available, which include a structure database and interactive graphics. Perhaps the leading solid state physics/chemistry packages are those of MSI Inc., whose Cerius^2 package includes a development toolkit as well as many advanced features.

Although advanced visualisation packages such as Cerius^2 support different input formats and allow new formats to be defined, writing a module to create, for example, movies is not straightforward or automated. The programmer would have to be familiar with both the host program interface and the input and output formats. The large number of parameters involved in making movies forces us to use an automated batch method, which, therefore, has no need for an interactive interface at all. Output formats of significant interest are, for example, the POV-Ray input format or VRML output for use on the World Wide Web.

In order to create high quality graphics in a structured way, a graphics language is

necessary. The Persistence of Vision Ray tracing [143,144] team have developed a language which is highly structured and modular. Although the quality of the graphics produced depends entirely on the skill of the user, the production of animated sequences is very simple. Unfortunately, the rendering speed of POV-Ray is too slow to allow interactive manipulation of the objects created.

B.1.2 Impact of the World Wide Web

The Internet and its protocols and standards has changed computing dramatically over the last five years. Hyper Text Markup Language (HTML) [145] prompted the development of a similar standard for 3-dimensional objects: the Virtual Reality Markup Language (VRML), enabling users of networked computers (if equipped to understand VRML) to manipulate 3-dimensional objects on screen, regardless of the type of computer or operating system. VRML has not yet become as widespread as was initially anticipated, mainly because of the high demands placed on processing power when actually viewing a VRML world (a collection of objects). However, since VRML graphics are more basic than ray-traced graphics (POV-Ray), VRML can be used to manipulate models interactively.

The demand for programs which can run on any computer led to the development of JAVA by Sun. JAVA currently supports 3-dimensional graphics, but the development of applications is still in its infancy. Nevertheless, packages for visualizing simple molecules are available in Java (a basic example of the current abilities of the JAVA 3D API can be seen at e.g. http://ws05.pc.chemie.th-darmstadt.de/java/). Moreover, JAVA comes with both Netscape and Internet Explorer, the two most successful web browsers available at this time.

B.1.3 Advances in Hardware and Software

As always, the development of software and hardware go hand in hand. Powerful 3-dimensional applications rely on powerful processors and graphics sub-systems. Prompted by the computer games industry, astounding progress has been made in graphics sub-systems. 3D accelerated hardware is now as cheap as the Color Graphics Adapter (CGA) was ten years ago. Simple PC expansion boards developed for games are as fast as professional boards for high-end workstations were two years ago.

The games industry has developed a low level programming standard for these graphics boards. Microsoft supports this standard via the DirectX libraries and via Silicon Graphics' OpenGL [146, 147] libraries. Since DirectX is only available for Microsoft operating systems and OpenGL is available for virtually all modern desktop operating systems, the preferred development library in use today is OpenGL.

B.1.4 Atomistic Simulation

The examples of visualisation in this report are all based on the simulation of materials at the atomistic level, that is, we explicitly model the interactions between the constituent ions. All materials that we are concerned with are crystalline, that is, the ions occupy well defined positions in space relative to one another. This is known as a crystal lattice. Almost all materials used for engineering purposes possess defects. Some of the lattice ions are either missing (a vacancy) or occupy an unusual position (an interstitial). Alternatively, impurities may be present which may occupy either lattice sites or interstitial sites.

The aim of the type of calculations presented here is to understand how defects in a crystalline lattice behave. Consequently, it will become possible to predict how to modify a lattice in order that its properties can be tailored to a specific requirement.

B.2 Atomistic Simulation Methodology

The ionic configurations and defect enthalpies reported here were calculated using the "CASCADE" code [23]. The CASCADE code implements the Mott-Littleton procedure which divides the lattice into two regions: an Inner Region I, which incorporates the defect ions and in which the migration process takes place and an Outer Region II. In Region I the interactions between ions, as defined though inter-atomic pair potentials, are calculated explicitly, whereas in Region II forces are calculated using a continuum model. Ions in Region I are relaxed to zero force using a Newton-Raphson minimization procedure. The inter-ionic pair potentials consist of two terms: a long range Coulombic interaction and a short range parameterized interaction. The parameters are selected to ensure that the calculations reproduce experimental data [44].

B.2.1 Migration Enthalpy Calculation

We can use the static atomistic simulation code CASCADE to predict defect cluster enthalpies of intermediate steps in complex migration mechanisms. The migration path is the minimum energy pathway from one solution site to the other. In simple crystal structures, crystal symmetry can make this task very easy. In principle, to obtain a migration enthalpy profile we need only move the migrating species iteratively from one site to the other, miminizing the surrounding ions at each step.

However, impurity ions (e.g. I^- in U_3O_{8-z}) migrate via a lattice vacancy mechanism and clearly require that we introduce defect species into the lattice. A 2D schematic representation of such a mechanism is shown in Figure B.1. The large iodide ion moves between oxygen sites in this, the basal plane of the alpha-U_3O_8 structure. This is possible only when a vacant oxygen site is available for the iodide to move into. The migration of oxygen in the structure is orders of magnitude faster than that of iodide. The schematic representation of iodide migration therefore shows oxygen ions moving across the basal

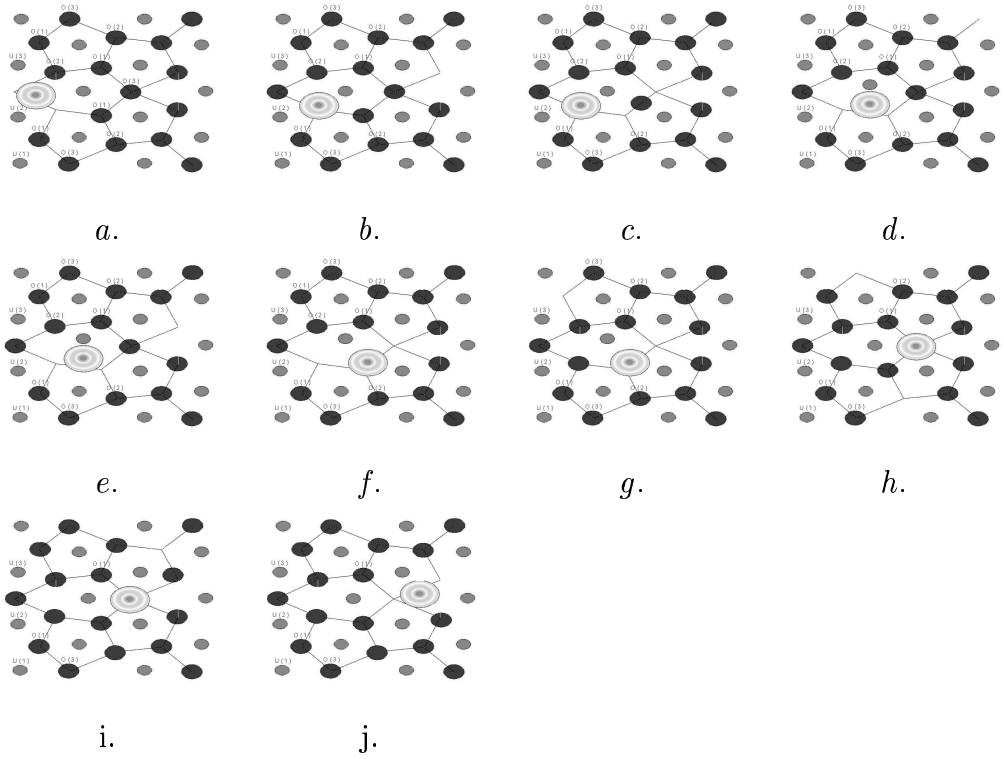


Figure B.1: From Picture *a.* to *j.*: the migration of I^- through the U_3O_8 lattice. *a.* Shows the initial position of the iodide in its equilibrium solution site. In pictures *b.* through *i* the ion moves though the lattice via vacancies which come available as a consequence of oxygen self-diffusion. Finally, in Picture *j.*, the iodide is located at an equivalent site as in Picture *a.*. It has travelled a distance of one primitive lattice vector.

plane and the iodide ion moving when a vacant site becomes available. (Note: oxygen ions that seem to disappear are simply moving out of the simulation box.) Each time an ion moves from one site to another, the difference between the solution site enthalpy and the maximum energy encountered in between solution sites is the activation energy for the diffusion step. Unfortunately this method does not yield pre-exponential terms or indeed, the absolute diffusion coefficient.

B.2.2 Molecular Dynamics Techniques (MD)

Molecular Dynamics can in principle be used to determine the necessary diffusion coefficients. Migration enthalpies can be predicted by simulating the thermal behaviour of a large number of atoms, and studying their predicted behaviour as a function of time. This involves solving Newton's equation of motion given a series of ions with a distribution of kinetic energies. Again, the forces which act between the ions are defined by both Coulombic and short range parameterized interactions.

The product of such a simulation is an extended sequence of position information. Statistical analysis of the root mean square displacement (RMSD) of the data as a function of displacement time yields the diffusion coefficient. Since diffusion follows a normal distribution, the variance of the ion positions progresses as:

$$VAR(t) = VAR(t0) + 2Dt, \tag{B.1}$$

where D is the diffusion coefficient, $VAR(t0)$ the distribution of the diffusing ions at $t = t0$ (which is zero in our case) and t is the time interval used to calculate the RMSD. Since we have a relatively small number of ions in the simulations presented here, the distribution of displacements is not exactly normal. Additionally, a large contribution to the observed RMSD will result from the thermal vibrations of ions. It is therefore important to run the simulation for long time intervals to make accurate predictions.

When the time intervals are too short, $VAR(t)$ will not show a linear behaviour with respect to t.

B.3 Visualisation Techniques

B.3.1 Images

High quality still images are produced using the Persistence Of Vision Ray tracing software [143, 144]. POV-Ray software includes a more than adequate descriptive language for three dimensional objects, colours and textures. It is simple to translate output from any type of modelling program (not restricted to atomistic modelling) to the input format required by POV-Ray. The "cas2pov" code (see Section B.7) does this conversion for CASCADE output files. After conversion, the user needs to add texture to the atoms and to position the lights and the camera. Recent versions of POV-Ray include extensive support for making movie frames automatically.

B.3.2 Movies

Animated sequences of, for example, oxygen migration were produced by running a static defect calculation for each frame in the movie. This simulation technique ensures that the behaviour of the lattice ions is realistic in every frame.

After converting the output from the defect calculations to the appropriate POV-Ray form, all configurations were ray-traced with POV-Ray using the same texture, lighting and camera conditions. The actual encoding of these frames into MPEG movies was carried out with "MPEG" version 1.2.1 by the Portable Video Research Group at Stanford (PVRG). MPEG video is a very good video compression system, which was designed for network use where bandwidth is limited. MPEG video encoding is, therefore, ideal for the Internet.

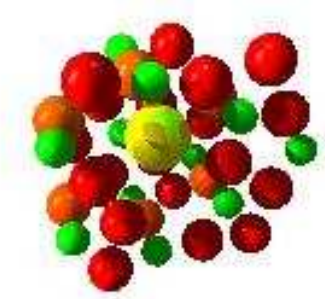


Figure B.2: A particle of U3O8-z. In the HTML report you can click the picture to manipulate the structure as a VRML object.

The process of making movies was simplified by the use of the "make" utility (e.g. Oram and Talbott [148]) in combination with the "perl" interpreted language (by Randal L. Schwartz [149]). All graphics manipulation was done with "ImageMagick" by John Cristy [150].

B.3.3 3D Interactive Objects (VRML)

The composition of movies has the disadvantage that the end user cannot manipulate the contents of the movie. To some extent this can be overcome via the Virtual Reality Markup Language (VRML) standard. VRML relies on the user to have a VRML browser installed on his system. Previously, only Silicon Graphics, Inc. systems were shipped with a VRML browser, but nowadays more and more home PC systems are able to understand VRML and contain the hardware to process complex three dimensional scenes.

Translation of simulation data to VRML is very similar to the translation performed for the POV-Ray translation and was done with the "cas2vrml" code which is very similar to the "cas2pov" code. However, translation to an image format is not necessary, as rendering is done at high speed (typically more than 5 frames per second) by the graphics

system of the person accessing the VRML code. Although the image quality of a scene rendered by POV-Ray will be superior to the result produced by the VRML browser, the VRML browser is fast enough to give the user the impression that they are manipulating the object in a literal sense. The image in Figure B.2 was produced with POV-Ray, but when it is clicked with the mouse, the VRML code is sent over and interpreted by the VRML viewer on the client (i.e. your) side. However, as VRML has limited support for dynamic scenes, such as MD simulations we need to write an application.

B.3.4 3D Interactive Graphics with OpenGL

Using the X-Windows Motif and OpenGL libraries we can produce easy-to-use applications with high speed realistic graphics. The advantages over MPEG movie encoding and VRML objects is clear. MPEG movies allow us to see a fixed sequence of frames, but do not allow any user interaction. VRML is all user interaction, but does not allow objects to be animated in time.

We have written an example application which shows the positions of the ions in a Molecular Dynamics simulation as they evolve with time and allows the user to manipulate the cluster. The customer required all applications to run on SGI systems. The code uses the standard SGI Motif libraries and the OpenGL libraries [146] which should be present on all SGI systems. However, the code compiles and runs on virtually all systems which have OpenGL and Motif libraries installed. A screen shot of the application is shown in Figure B.3.

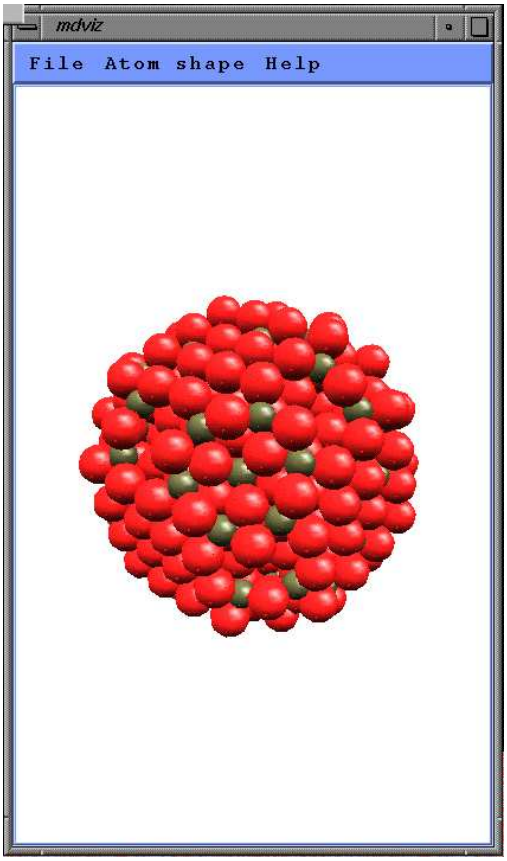


Figure B.3: A screenshot of the MDviz code in action. The user is investigating a high temperature simulation of a particle of UO_2.

B.4 The Migration of O^{2-} and U^{4+} in UO_2 and U_3O_8

B.4.1 Crystallography

UO_2 exhibits a cubic, fluorite-type structure (Wyckoff IV,a1) as shown in Figure B.4. In this structure, vacancy assisted migration of oxygen ions occurs along <100> directions between adjacent sites. In U_3O_8 (the structure is shown in Figure B.5), the oxygen transport behaviour is more complex because we may distinguish four different symmetry types for oxygen in the structure.

As both the oxidation rate for U(VI) solids and vacancy assisted diffusion mechanisms depend on the migration of oxygen, reliable predictions of the diffusion coefficient are important.

The self diffusion of uranium through the lattice is important when studying fission product behaviour; in particular, positively charged fission products often rely on uranium self diffusion which becomes the controlling mechanism in hyper-stoichiometric UO_{2+x}. Simulating the migration behaviour of uranium ions though uranium oxides using the Quasi Harmonic approach is computationally more intensive, as the migration paths are not straight.

B.4.2 Methodology

Migration though the lattice can be simulated by placing the migrating ion at consecutive sites along its migration path and relaxing the surrounding lattice. This method is known as the Quasi-Harmonic approach. The energy of the ion will change from the relaxed lattice value through a maximum and back to the relaxed value again. The difference between the relaxed enthalpy and the saddle point enthalpy is the migration enthalpy. The program used to simulate the lattice relaxation is CASCADE [23].

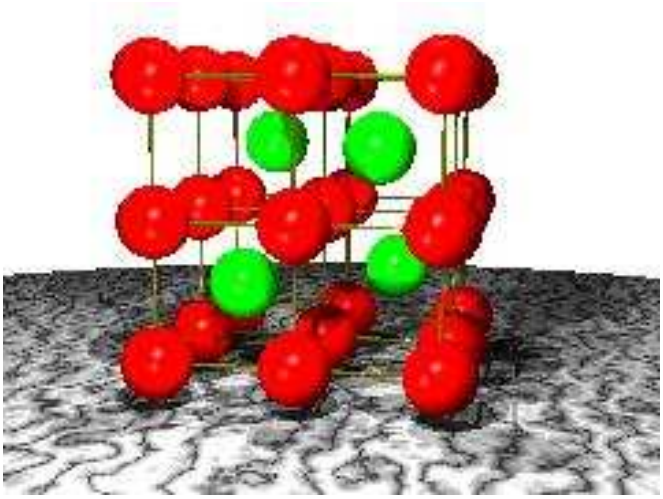


Figure B.4: The fluorite crystal structure of UO_2, created with POV-Ray.

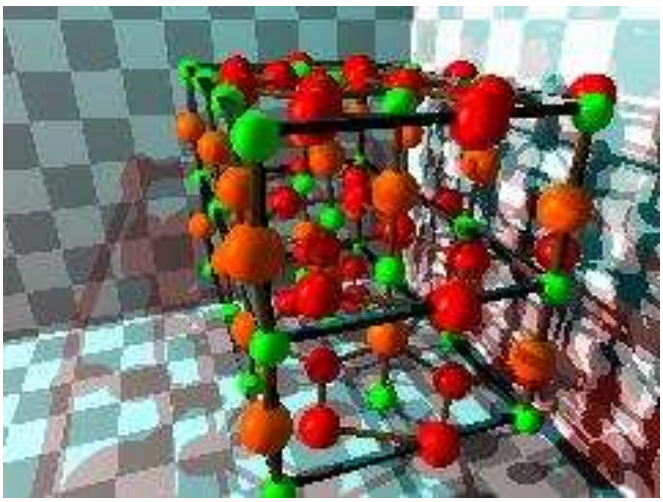


Figure B.5: The crystal structure of U_3O_8 created with POV-Ray.

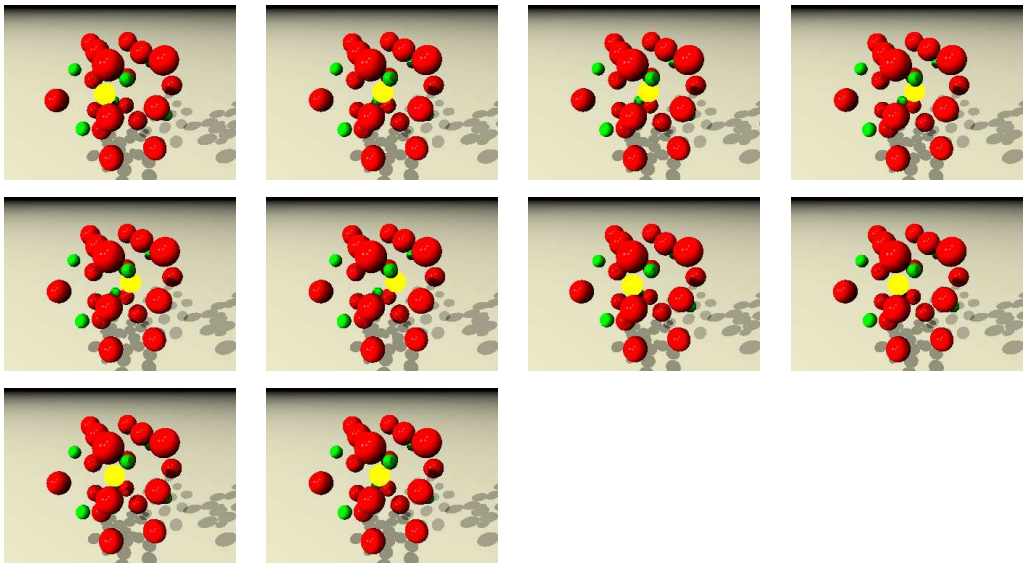


Figure B.6: A fragment of the UO_2 crystal cut from an infinite crystal in which an oxygen ion migrates. In the original report, clicking on the image would show an animation of the migration of oxygen in UO_2. Presented here are a selection of snapshots from the original animation. Note that the actual calculations on which this simulation is based consider an infinite lattice, not just the atoms shown in this animation.

If the migration path is not known, a large number of calculations need to be performed to find the path. Fortunately, in many structures symmetry predicts in which plane migration will take place and the path can be found more easily.

B.4.3 Migration of U^{4+} and O^{2-} in UO_2

Figure B.6 shows a movie of the oxygen migration process in UO_2. It is clear that the migration step does not substantially strain the lattice, that is, lattice ions are not displaced from their lattice sites to any great extent. The associated oxygen migration enthalpy is 0.8 eV.

Migration of uranium ions on the other hand (Figure B.7) shows extensive lattice

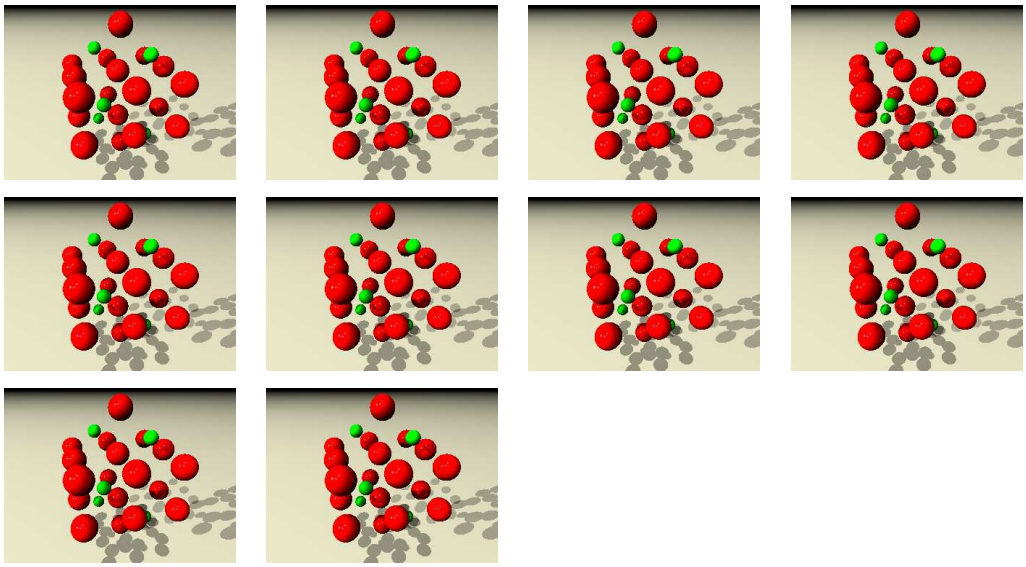


Figure B.7: The migration of uranium in UO_2, details as in Figure B.6.

distortion during the migration process. Additionally, the charge state of uranium induces extensive ionic polarisation effects (not visualized here). The migration enthalpy is more than 6 eV.

As explained earlier, as an ion moves through the lattice from one stable energy position to another (in the case of O^{2-} and U^{4+} migration in UO_2 these are near the stable lattice sites) the ion experiences an increase in potential energy up to a maximum which occurs at the saddle point. In these cases, the saddle point occurs half way along the migration pathway. The energy change is visualized by an alteration in the colour of the migrating species. In Figure B.6, the migrating oxygen ion begins the motion as a red ion. As the potential energy increases, the colour changes until it is yellow at the saddle point. The colour changes back to red as the ion completes its journey. For U^{4+} migration as shown in Figure B.7, the ion is initially green, becomes progressively pale towards the saddle point and darkens as it completes the migration process.

Therefore, colour may be used as a dynamic variable, to import technical data, in addition to its more usual function of simply differentiating between species.

B.5 Results

B.5.1 Migration of Iodide in U_3O_{8-z}

An example of a fission product migration mechanism where oxygen self-diffusion is important is the migration of iodide in U_3O_{8-z}. Iodide is a radiologically important volatile fission product. The large ionic radius and its negative charge restricts this ion to occupying substitutional oxygen ion sites in the U_3O_8 structure. Migration of iodide relies, therefore, on the mobility and availability of oxygen vacancies (as shown in Figure B.1) In U_3O_{8-z}, a large number of oxygen vacancies are available to assist iodide to migrate through the lattice. In essence, migration occurs via a number of stable iodide solution sites of different enthalpies. Iodide moves from one site to the other via interaction with oxygen vacancies.

Figure B.8 shows how this migration may take place. However, with a complex structure such as U_3O_8, the information provided by Figure B.8 alone could be confusing. Thus, conjunctive use of schematic animations such as in Figure B.1 may be very valuable in guiding the viewer.

B.5.2 O^{2-} migration in UO_2 via MD

Migration enthalpies can be predicted by simulating the thermal behaviour of a large number of ions or molecules and studying their predicted behaviour as a function of time. Here, the motion of ions is simulated by solving Newton's equations of motion for a system of ions, using a methodology known as Molecular Dynamics Simulation (MD). The interactions between the ions are the same as in the calculations described above. However, interpretation of the data is very different and is based on the statistical analysis of the ionic positions as a function of time. Owing to the computational constraints only relatively fast migration processes, such as oxygen self diffusion in UO_2 may be studied at

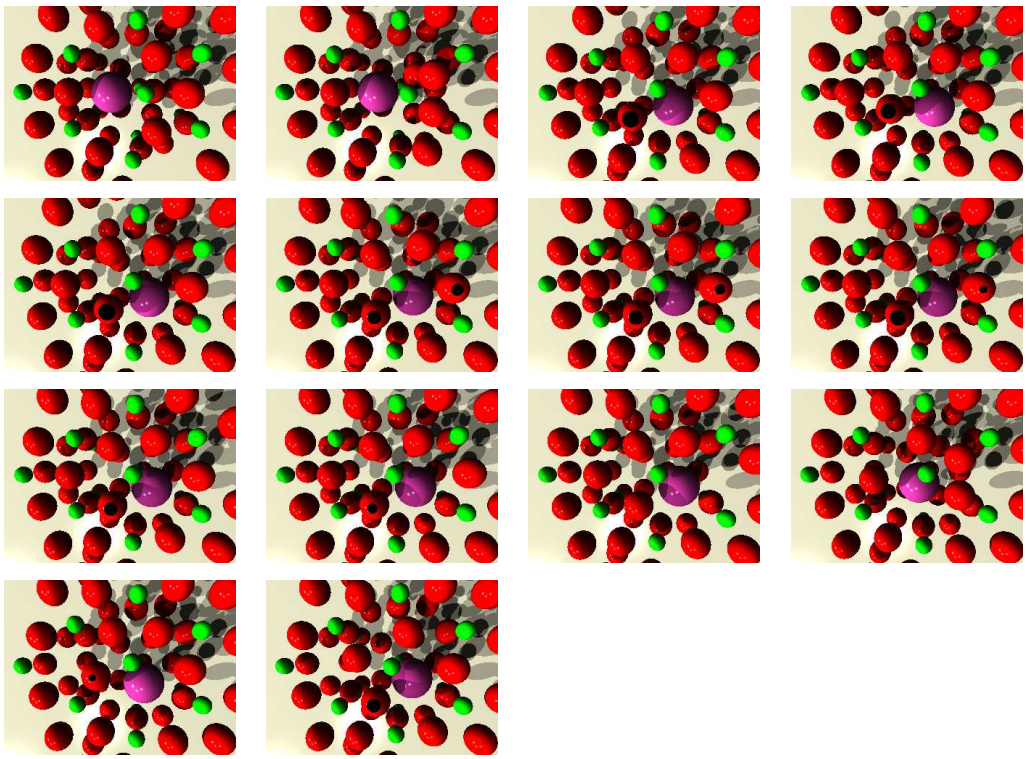


Figure B.8: Visualisation of an oxygen vacancy assisted migration process: I^- migration in U_3O_{8-z}.

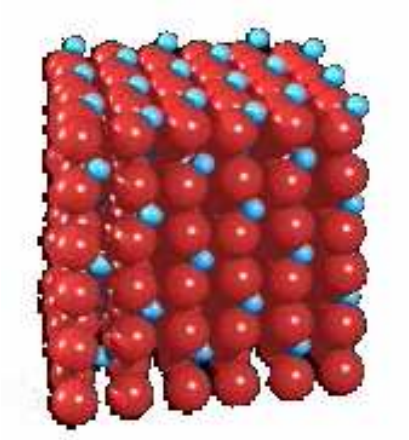


Figure B.9: A particle of UO_2 as simulated with the Molecular Dynamics technique.

present.

Figure B.9 shows a particle of UO_2, containing 324 atoms. Although we are able to simulate particles of several thousands of atoms, a small particle was chosen to reduce calculation times. The movement of the ions in the particle were simulated for three different temperatures: 500K , 700K and 900K. The particle was simulated for 150 ps at each temperature. The total simulation time is about 600 ps (including the time to heat up the particle).

Figure B.10 shows the Mean Square Displacement (MSD) of the atom positions as a function of time. A ion displacement threshold of 1Å was used to filter the effects of atom vibration from the RMSD calculation.

Figure B.11 shows the resulting diffusion coefficients. Our example calculation does not yield a straight line Arrhenius plot for three reasons:

1. Probability theory predicts that the variance (or MSD) of Gaussian distributions progress linearly with interval time. This theory fails if the diffusion length of the ions approaches the cluster radius, especially at high temperatures and long interval times where the diffusion length is, on average, shorter but of the same order of magnitude as the cluster size.

2. It is difficult to distinguish atomic vibration from migration at low temperatures, possibly affecting the 500 K result.

3. The migration activation energies for surface species will be different from these for bulk species. We have developed methodologies for differentiating between bulk and surface ions [31] although here the small size of this particle inhibits a clear distinction between the two types of species.

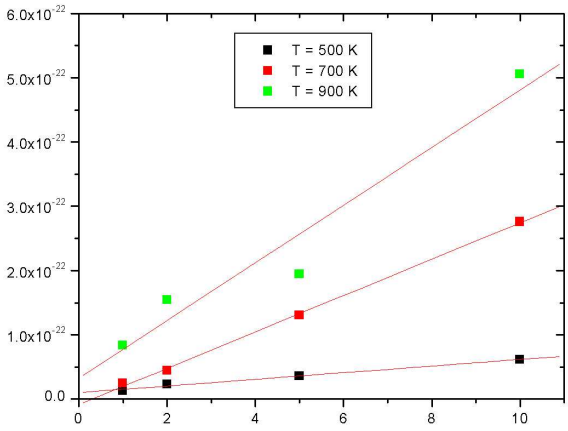


Figure B.10: The mean square displacement of lattice ions at constant temperature, which should be a linear function of time. The gradient of this function is the diffusion coefficient at that temperature.

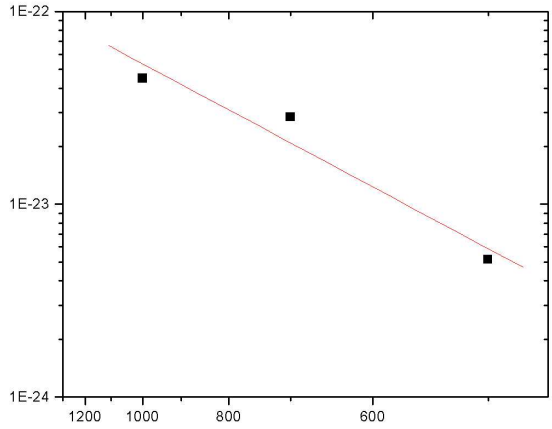


Figure B.11: In an Arrhenius plot of the diffusion coefficient normal activated processes should result in a straight line. As our calculations were only performed at three different temperatures, there is a considerable uncertainty associated with the predicted activation energy (i.e. the slope of the line).

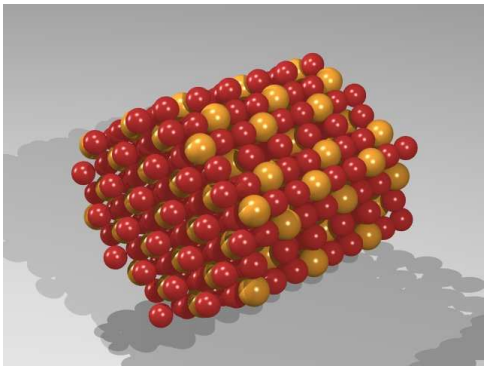


Figure B.12: A cut from an ideal crystal of LaF_3. Click the image to see a movie of the heating process from 10K to 1000K.

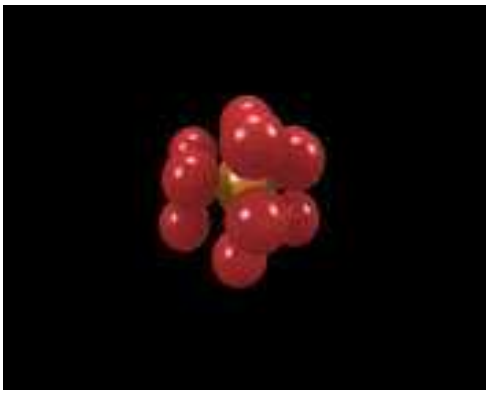


Figure B.13: The hopping behaviour of fluoride ions in LaF_3.

B.5.3 Modelling of an LaF_3 Nanocluster with MD

In the previous sections we have studied migration mechanisms in detail, by simulating individual ion migration and the behaviour of an ensemble of atoms and used statistical techniques to extract diffusion coefficients. In this section, we will study the movement of individual atoms in a molecular dynamics simulation.

We start with an ideal 552 atom cluster of LaF_3 (Figure B.12.). When the cluster is heated, ions begin to migrate. After the simulation has been performed, we can remove most of the atoms from the visual field and focus on a few atoms which move from their relaxed positions. An animation of this process is shown in Figure B.13, which illustrates the hopping motion model proposed for migration of F^- ions through the lattice. However, it also shows that at one point toward the end of the video clip, many of the ions hop at approximately the same moment; this occurs via a concerted process. Now that we are aware that this type of process occurs, we must derive the analysis tools necessary to assess its overall importance in the migration process. It is pertinent to note that without an animated sequence this complex migration process in LaF_3 would probably never have been discovered.

B.6 Concluding Comments

Transport phenomena in solids can be exceedingly complex. It is necessary, therefore, to develop visualisation techniques which are able to emphasize important aspects of such processes. Without doubt, such animated sequences are able to convey information which would otherwise require many pages of highly detailed text. Here we have provided some examples which demonstrate this point using some of the most recent facilities available. None of these examples required particularly powerful hardware or expensive software.

With the advent of 3-dimensional programming libraries and standards (such as OpenGL, Java 3D API and VRML) it is possible not only to view a static object, but to alter the

perspective of a 3-dimensional evolving object. The next step will be to specifically adjust individual elements of a simulation as the simulation takes place. For example, to introduce a molecule or ion to a surface. Such processes underly chemical vapour deposition, ion beam milling, ion implantation or radiation damage. Furthermore, as computing facilities become more powerful, it will be possible to carry out such simulations as part of the control loop for surface techniques. In all cases it is critical to provide an effective graphical interface for interpretation.

B.7 Computational Issues

In this section, some of the technical details of modelling software are discussed. In particular, the merits of various tools and libraries used for developing visualisation software are reviewed. Finally, details of how the images and animations were constructed are discussed.

B.7.1 Example Modelling Software

An example application was developed for viewing the results of a Molecular Dynamics simulation. The following tools were used for the development of the code:

- Autoconfig, Free Software Foundation, Inc.
- "make"
- "RCS" Revision Control System
- Silicon Graphic's and GNU's C++ compiler
- File_ARC class, V. Bulatov, Atomistic Simulation Group
- CascadeCluster class, G. Busker, Atomistic Simulation Group
- OpenGL (Silicon Graphics Inc.) and Mesa (Brian Paul, 1995-1997)

- Motif (Silicon Graphics Inc.) and LessTiff (Free Software Foundation Inc., 1995)
- GLUT library, Silicon Graphics, Inc.
- trackball, Gavin Bell, Silicon Graphics, Inc.

"Autoconfig" is a utility which allows developers to configure their source code to be platform independent. By using "Autoconfig" our package compiles under Digital Unix, SGI IRIX, Solaris, Linux and probably many more operating systems. The "make" utility is instrumental in this regard, as "Autoconfig" changes the "make" project settings depending on which platform is used. "RCS" was used to keep track of revisions of the source codes.

The C++ language was used, because data abstraction is transparent and modular code is easy to implement in C++. Some of the classes used in this project were written a long time ago by myself and V.L. Bulatov. The fact that no changes needed to be made to those classes is significant: problems caused by having different versions of the same module can be disastrous for further development of code.

OpenGL was used as the main graphics code. It handles such details as surface rendering, object and camera definition, translation and lighting. Motif was used for the construction of menus and dialogs.

Both "GLUT" and "trackball" are codes which were developed at Silicon Graphics Inc. to demonstrate the possibilities of OpenGL. The icosahedron drawing routine was taken from the "GLUT" library and the implementation of the object manipulation by mouse was copied from "trackball".

B.7.2 Conversions

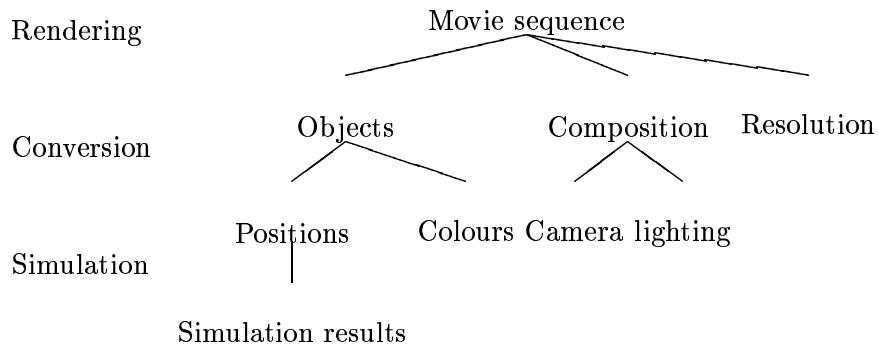
Some code was developed to cope with the output data presented by simulation software, such as "CASCADE" and "Penicillin". Codes for creating POV-Ray input files and even VRML files were developed, using the same classes which were used for the visualisation

package.

Some features of the conversion codes are the ability to specify cutting planes and the inclusion of a ionic radius and colour database. These tasks are by no means "user friendly", yet.

B.7.3 Making the Movies and Pictures

The use of "make" and "Makefiles" has greatly simplified the building of complex packages and can be used successfully for tasks such as making movies. The process of making a movie is shown here schematically:



If, for instance, we want to change the length of the movie, we will have to perform additional calculations. On the other hand, if we just want to change the resolution, we can suffice with re-rendering the movie frames. By constructing a "Makefile" project with the correct hierarchical structure, "make" will decide which calculations, renderings and conversions need to be done. Since a small 100 frame movie is constructed from up to 1000 source files, it is nearly impossible to manage this by hand, unless one is prepared to redo all the necessary calculations (which may take up to a week on an average workstation).