Teil II

Methoden- und Programmentwicklungen

Kapitel 3

Grafische Verfahren

3.1 Moliso

3.1.1 Motivation

Die Entwicklung der Computergrafik wird durch Hardwarehersteller und die Spieleindustrie ständig vorangetrieben. Spiele mit einer Grafik-Technologie, die älter als ein Jahr ist, gelten bereits als veraltet. Umgekehrt laufen aktuelle Spiele teilweise auf Rechnern, die älter als zwei Jahre sind, nicht oder nur sehr eingeschränkt.

*Beispiel-Grafik aus dem aktuellen PC-Spiel „**Gothic3**[a]“. Man beachte die realistische Darstellung der abendlichen Lichtstimmung und die Fernsicht!*

[a] Gothic3 © 2006 by Piranha Bytes

Doch was in Spielen für „realistische“ Darstellung virtueller Spiele-Welten geeignet ist, kann auch zur Darstellung komplexer wissenschaftlicher Sachverhalte dienen.

„Ein Bild sagt mehr als tausend Worte“, lautet eine bekannte Formel. Sie gilt insbesondere bei der Darstellung komplexer wissenschaftlicher Sachverhalte. Es ist also naheliegend, die vorhandenen Möglichkeiten moderner Computer-Hardware auch für wissenschaftliche Zwecke zu nutzen.

Zu Beginn dieser Arbeit war kein geeignetes Programm erhältlich, welches transparente, farbkodierte Isooberflächen aus experimentellen Daten erstellen konnte. Viele grafische Programme neigen zudem dazu, wahre Alleskönner

zu sein. Dies führt meist dazu, dass die Bedienung nur über eine eigens zu erlernende Kommandosprache möglich ist. Ziel war es somit, ein kleines benutzerfreundliches Programm zu erstellen, welches transparente farbkodierte Isooberflächen aus experimentellen Rasterdaten (Grid files) erstellen kann.

3.1.2 Benutzerfreundliche Bedienelemente

Ein benutzerfreundliches Programm zeichnet sich (zumindest nach der Meinung des Autors) durch folgende Bedienelemente aus:

- Übersichtliches Benutzermenü für häufig verwendete Funktionen.
- Naheliegende Tastenkürzel für alle im Benutzermenü aufgeführten Funktionen wie beispielsweise 'b' für das Zeichnen von Bindungen.
- Selten zu verändernde Einstellungen werden in einer kurzen und übersichtlichen Steuerungsdatei vorgenommen.

Moliso, das hier entwickelte Programm, berücksichtigt diese drei Bedienelemente. Es geht sogar noch einen Schritt weiter, indem es die besagte Steuerungsdatei mit den aktuellen Einstellungen selbst schreibt. Dies spart dem Benutzer den Blick in das Handbuch für die korrekte Schreibweise der Befehle.

Abbildung 3.1 zeigt die aufgeklappte Benutzermenüstruktur. Die schwarzen Striche bedeuten, dass sich entsprechendes rechts stehendes Untermenü hinter dem Menüpunkt links verbirgt. Buchstaben in runden Klammern geben Tastenkürzel an, mit denen man dieselbe Funktion schneller erreichen kann. Schema 3.2 zeigt den typischen Inhalt einer Steuerungsdatei.

3.1.3 Farbkartierte Oberflächen

Farbkartierung oder *engl.* color-mapping kennt jeder, der schon einmal eine Landkarte benutzt hat. Dabei wird in zwei-dimensionalen Darstellungen eine weitere Information durch die Farbgebung veranschaulicht. Dieses Verfahren lässt sich natürlich auch auf dreidimensionale Isooberflächen anwenden. Mit dem Programm Moliso kann man beispielsweise auf eine Isodichteoberfläche das elektrostatische Potential (ESP) auf dieser Oberfläche darstellen. Dies hat

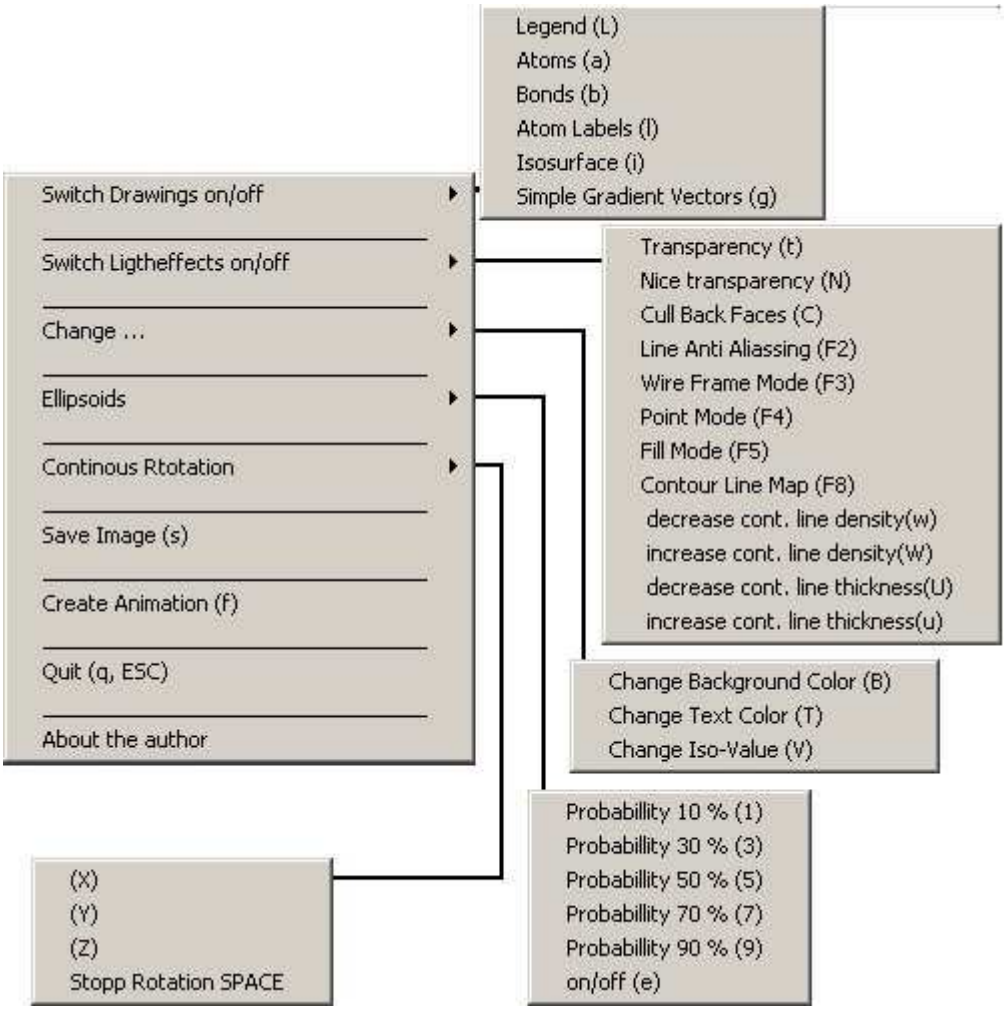

Abbildung 3.1: Menüstruktur von Moliso

```
Iso_Grid=xd_rho.grd
Map_Grid=xd_esp.grd
Grad_Atom=0.0 0.0 0.0 1.0
Refl_Texture=sph5.ppm
Iso_Value=0.300000
Gradient_Cutoff=0.000100
Sphere_Detail=2
XD_RES_PATH=xd.res
Use_Stored_Surface=0.3.face
Store_Surface=
Transparency_Mode=NICE
Enable_Ellipsoids=NO
Ellipsoid_Probability=50
Regular_Iterations=0
Background_Color=1.000000 1.000000 1.000000 1.000000
Text_Color=0.000000 0.000000 0.000000 1.000000
Mouse_Interaction=0
```

Abbildung 3.2: Typischer Inhalt einer Steuerungsdatei für Moliso

gegenüber einfarbigen Isooberflächen des elektrostatischen Potentials mehrere Vorteile: Man erkennt den räumlichen Verlauf des Potentials, was bei diskreten Isooberflächen nicht möglich ist. Da das ESP an manchen Stellen eine sich räumlich nur sehr langsam verändernde Funktion ist, resultieren in kleinen Änderungen des Wertes der Isooberfläche oft drastische Veränderungen der Form und Größe derselben. Dies führt dazu, dass bei Vergleichen des ESP verschiedener ähnlicher Verbindungen die vorhandene Ähnlichkeit des ESPs nicht ersichtlich ist. In Moliso wird standardmäßig ein Farbgradient benutzt, welcher aus den Spektralfarben des Sonnenlichts besteht (rot-orange-gelb-grün-cyan-blau-violett). Hierbei sind die Farben, um das räumliche Empfinden des Betrachters nicht zu stören, der Empfindlichkeit des menschlichen Auges angepasst. Das menschliche Auge ist viel empfindlicher für Grüntöne und Rottöne als für Blautöne: so erscheint dem Betrachter ein gelbes Objekt (Mischung aus rot und grün bei additiver Farbmischung) viel heller als ein blaues bei gleicher Farbsättigung. Der Nutzer hat jedoch natürlich die Möglichkeit, den Farbgradienten seinen Wünschen und Bedürfnissen anzupassen, wobei auch der Grad der Transparenz für jede der bis zu sieben möglichen Farbgradientenstufen beliebig wählbar ist. Um die Vergleichbarkeit zwischen mehreren Abbildungen

zu gewährleisten, ist es möglich, gemeinsame Mini- und Maximalwerte für den Farbgradienten festzulegen, wodurch ein bestimmter Wert des ESP auf allen Abbildungen demselben Farbwert entspricht. Zusätzlich zur Farbkartierung kann man die Darstellung auch mit Isokonturlinien oder besser Isokonturbändern einstellbarer Dichte und Breite versehen. Diese Isokonturbänder sind nicht an allen Stellen gleich breit, da sie durch ihre Breite auch die Steigung des ESP veranschaulichen.

3.1.4 Korrekte transparente Darstellungen

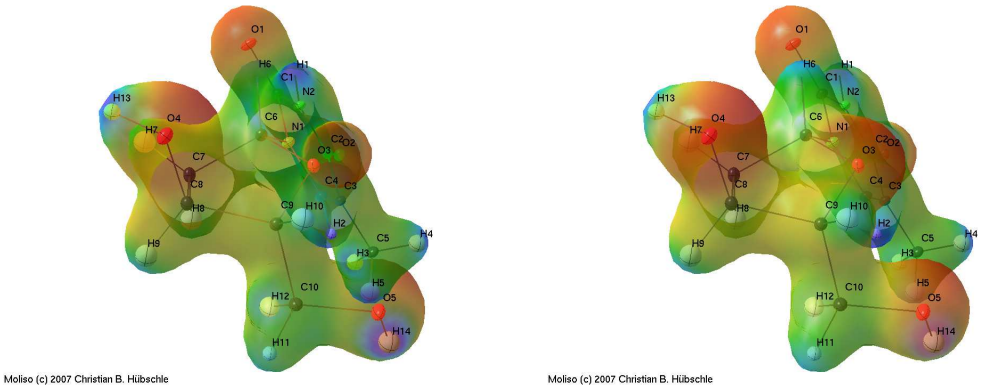

Abbildung 3.3: Nicht korrekte (links) und korrekte (rechts) transparente Darstellung.

Da Isooberflächen der Elektronendichte und anderer physikalischer Eigenschaften die Struktur des Moleküls ganz oder teilweise einhüllen, ist es notwendig, um sowohl die Struktur als auch die physikalische Eigenschaft in einem Bild zu visualisieren, dass diese Isooberflächen transparent dargestellt werden können. Bei heute gebräuchlichen Verfahren zur Visualisierung von räumlichen Objekten werden diese Objekte in Polygone zerlegt. Die Eckpunkte dieser Polygone werden durch Matrizenoperationen gestaucht, gedreht und perspektivisch verzerrt. Danach werden die sichtbaren Polygone ermittelt, deren Farbe sich aus der relativen Position zu einer virtuellen Lichtquelle errechnet. Ein einfaches Verfahren zur Ermittlung sichtbarer Flächen wird auch Z-Bufferverfahren genannt. Dabei werden alle gespeicherten Objekte nur dann gezeichnet, wenn kein anderes Objekt näher zum Betrachter steht, wenn es also nicht verdeckt wird. In welcher Reihenfolge das Programm diese Polygone zeichnet, spielt dabei keine Rolle. Sind die Objekte dagegen transparent, so kommt dieses Verfahren nicht zum Einsatz, da sich die Polygone nicht mehr

verdecken können. Die Farbe eines transparenten Polygons berechnet sich dann als Mischung aus der Farbe eines nicht transparenten Polygons und dem bisher gezeichneten (Hintergrund oder andere Polygone). Daraus ergibt sich leicht, dass die Reihenfolge, in der das Programm die Polygone zeichnet, nun nicht mehr belanglos ist. Wird zum Beispiel ein transparentes Polygon aus dem Vordergrund zuerst gezeichnet, erhält es eine Mischfarbe aus seiner eigenen und der des Hintergrunds. Wird dann später ein dahinter liegendes Polygon gezeichnet, so erhält dieses seine Farbe aus seiner eigenen und der des zuvor gezeichneten Polygons. Dies erweckt jedoch beim Betrachter den falschen Eindruck, dass das hinten liegende Polygon im Vordergrund sei. Dies ist in Abbildung 3.3 recht gut zu erkennen. Um dies zu vermeiden, müssen die transparenten Polygone immer von hinten nach vorne gezeichnet werden, das heißt, sie müssen sortiert werden. Diese Sortierung müsste aber nach jeder Drehung wiederholt werden, was aufgrund der großen Zahl an Polygonen, aus denen eine Isooberfläche besteht, und des sich daraus ergebenden hohen Rechenaufwands aber nicht praktikabel ist.

Für Moliso wurde dieses Problem auf ganz andere Weise gelöst. Das Programm berechnet die Polygone aus sechs verschiedenen Ansichten und hält diese im Speicher vor. Dreht man die Isooberfläche, so wird aus dem Speicher immer die richtige Anordnung der Polygone geladen und gezeichnet. Dabei kann die Ansicht so schnell gedreht werden, wie dies ohne diesen Trick möglich wäre. Allerdings benötigt dieses Verfahren eine etwas längere Ladezeit und einen höheren Arbeitsspeicherverbrauch.

3.1.5 Beleuchtungsmodelle und Verschönerungstechniken

Um bei dem Betrachter die Illusion von der Dreidimensionalität zu erwecken, verwendet man in der Computergrafik sogenannte Beleuchtungsmodelle. Bei diesen wird aus dem Winkel zwischen der Flächennormalen und einer virtuellen Lichtquelle der Helligkeitswert des gezeichneten Polygons berechnet. Man unterscheidet dabei zwischen Umgebungslicht, welches aus allen Richtungen kommt, diffusem Licht, welches gerichtet ist, und Glanzlicht, das nur einen eng eingegrenzten Winkelbereich aufhellt. Diese drei Beleuchtungsarten werden bei Moliso verwendet. Glanzlicht und diffuses Licht sind für den dreidimensionalen Eindruck besonders wichtig. Da aber durch Glanzlicht bei geringen Rasterauflösungen der Aufbau der Isooberflächen aus einzelnen flachen Polygonen besonders unschön hervorgehoben wird, verzichten besonders älte-

re Programme oft auf diese Beleuchtungsart. Moliso verwendet optional einen einfachen iterativen Glättungsalgorithmus, um dieses Problem zu verbessern. Daneben kann bei Moliso mit Hilfe der „Sphere-Mapping “-Technik ein Bild so auf die Oberfläche kartiert werden, dass der Eindruck entsteht, die Oberfläche reflektiere dieses Bild. Verwendet man für dieses Bild die Fotografie einer polierten Kugel, so erscheint die Isooberfläche ebenfalls besonders glatt und poliert.

3.1.6 Moleküldarstellungsmodelle

Um die auf eine Isooberfläche kartierten Informationen der Struktur richtig einordnen zu können, hilft es, die Molekülstruktur simultan mit der Isooberfläche darzustellen. Moliso bietet hierzu mehrere Darstellungsmodelle an: das Kugel-Stab-Modell, bei dem die Atome als Kugeln und die Bindungen als Zylinder dargestellt werden, oder ein Draht-Modell, bei dem nur die Bindungen als Linien dargestellt werden. Die Atome lassen sich alternativ dazu auch als ausgefüllte Verschiebungs-Ellipsoide zeichnen. Diese Schwingungsellipsoide grenzen den Bereich von mehr als 10, 30, 50, 70 oder 90% Aufenthaltswahrscheinlichkeit ein. Daneben hat der Nutzer die Möglichkeit, Farbe und Radien der Atomkugeln und der Bindungen frei zu wählen. Durch Setzen der Atomradien auf die Van-der-Waals-Radien der jeweiligen Atomsorten ist auch eine Art Kalottenmodell verfügbar.

3.1.7 Bild- und Filmerstellung

Moliso kann alle Visualisierungen, die es produziert, als Einzelbild oder auch als kurzen Videoclip einer kompletten Drehung um die Y-Achse exportieren. Beim Abspeichern eines Einzelbildes speichert das Programm auch eine Orientierungsmatrix ab, die alle Drehbewegungen, Translationen und Vergrößerungen enthält. Dadurch kann der Nutzer später durch einen einzigen Tastendruck auf der Tastatur diese Orientierung des Moleküls wiederherstellen. Zur Erstellung der Videoclips benutzt Moliso die Windowsschnittstelle „Video for Windows“. Dadurch können sämtliche installierte Videocodecs benutzt werden. Ein Videocodec beschreibt, wie die Filmdaten komprimiert und gespeichert werden. Die Qualität des Videoclips hängt vom Videocodec und der verwendeten Komprimierungsstufe ab. Außerdem ist es möglich, Vektorgrafiken abzuspeichern, bei denen die Pixelauflösung keine Rolle spielt. Da die dabei

verwendete Technik zur Ermittlung sichtbarer Flächen weniger gut funktioniert als die in Grafikkarten eingebaute, kann es bei diesen Vektorgrafiken zu Darstellungsfehlern kommen. Die erzeugten Dateien können zudem sehr groß werden, was die Zeit zum Ausdrucken erheblich verlängert.

3.1.8 Quantitative Analyse der Oberfläche

Neben dem eingeschlossenen Volumen, welches in Moliso durch Abzählen der eingeschlossenen Voxel abgeschätzt wird, bietet Moliso auch die Analyse der auf die Oberfläche kartierten Werte an. Die Berechnung der einzelnen Werde wurde bereits in Kapitel 2.4 auf Seite 19 beschrieben. Um die Ausgewogenheit von negativen und positiven Oberflächenwerten zu erhalten, wird ν berechnet:

$$\nu = \frac{\sigma_+^2 \sigma_-^2}{[\sigma_{tot}^2]^2} \tag{3.1}$$

Hat ν einen maximalen Wert von 0.25, sind negative und positive Oberflächenwerte ausgewogen verteilt. Außerdem wird der Ort des kleinsten und größten Wertes auf der Oberfläche angegeben. Durch diese Größen können die Oberflächen genauer beschrieben werden und Unterschiede besser diskutiert werden.

3.1.9 Schnittebenen

Eine sehr neue Funktion in Moliso sind Schnittebenen. Sie ermöglichen es, den Verlauf einer Funktion in einer bestimmten Ebene zu visualisieren. Die programmtechnische Implementierung einer beliebigen Schnitteben durch ein dreidimensionales Punktgitter mag hier nicht die eleganteste Methode sein, soll aber aufgrund ihrer ungewöhnlichen Herangehensweise trotzdem näher erläutert werden. Leitgedanke hinter dem verwendeten Ansatz war, vorhandene Programmresourcen möglichst wiederzuverwenden. Daher sind die in Moliso dargestellten Schnittebenen eigentlich Isooberflächen. Hierzu wird von der Abstandsgleichung einer Ebene $d = \vec{n} \cdot \vec{r}$ ausgegangen, in der $\vec{n}$ die Normale der Ebene ist, $\vec{r}$ ein Ortsvektor im Raum und d der Abstand des Vektors $\vec{r}$ von dieser Ebene. Rechnet man nun diese d-Werte für jeden Punkt eines dreidimensionalen Grids aus, so ist eine Isooberfläche bei einem Wert, der dem Abstand zur Ebene entspricht, eine Schnittebene durch dieses Punktgitter. Auf eine solche Schnittebene kann nun analog zu den Isooberflächen eine Funktion aus einem anderen Punktgitter kartiert werden. Der Nutzer kann eine Schnittebene

durch einfaches Klicken auf drei linear unabhängige Atome definieren und die Höhe anschließend nach Belieben anpassen.

3.1.10 Programmierdetails, Verfügbarkeit und Dokumentation

Moliso ist in C und C++ geschrieben. Anweisungen werden, sofern es sich nicht um sogenannte Makros handelt, in dieser Programmiersprache durch Semikola ; von einander getrennt. Der Quellkode besteht aus 6780 durch Semikola getrennten Anweisungen. 4151 davon sind im Rahmen dieser Arbeit entstanden, der Rest dient der Film- und der Vektorgrafikerstellung und ist von den Autoren Jonathan de Halleux [24] und Christophe Geuzaine [25] entlehnt. Das Programm wurde auf Linux- und Windows-Rechnern getestet. Moliso ist für jedermann frei und kostenlos von der Seite
http://userpage.chemie.fu-berlin.de/~chuebsch/moliso
als Linux- und Windows-Version herunterzuladen. Dort ist auch eine Anleitung in englischer Sprache zu finden. Eine Statistik der Downloadzugriffe ist im Anhang in Tabelle A.1 zu finden.

3.2 Molecool

3.2.1 Motivation

Es gibt sicherlich eine Menge hochpotenter Moleküldarstellungsprogramme, so dass es auf den ersten Blick nicht naheliegt, ein eigenes Programm zu diesem Zweck zu schreiben. Jedoch sprechen folgende Punkte auch dafür: Es gibt kein Programm, welches direkt Molekülkoordinaten aus XD-Verfeinerungen darstellen kann. Viele Moleküldarstellungsprogramme schreiben Logdateien, was bei nicht vorhandenen Schreibrechten (in Verzeichnissen von Kollegen) zu Programmabstürzen führt. Außerdem sind einige der notwendigen Routinen zum Darstellen von Molekülen bereits mit Moliso erstellt worden. Eine für die experimentelle Ladungsdichtebestimmung wichtige Funktion fehlt allerdings bei den anderen Moleküldarstellungsprogrammen: die Darstellung von Lokalen Koordinatensystemen, welche besonders bei der Verwendung von lokaler Symmetrie wichtig ist, da sich bei der Definition der lokalen Koordinatensysteme leicht Fehler einschleichen.

3.2.2 Eigenschaften von Molecool

Molecool ist ein kleines Programm zum Darstellen von Molekülen aus XD-, SHELX- und CIF-Dateien. Es findet selbstständig Wasserstoffbrücken und packt falls gewünscht symmetrieerzeugte Moleküle um die asymmetrische Einheit, welche einen Maximalabstand unterschreiten. Wie Moliso ist auch Molecool mit einem Benutzermenü und Tastaturkürzeln zu bedienen, jedoch wird keine Steuerungsdatei benötigt. Eine weitere Besonderheit von Molecool ist die Darstellung von lokalen Koordinatensystemen. Die Definition der lokalen Koordinatensysteme durch den XD-Nutzer dient der Anwendung von lokaler Symmetrie und ist ein Punkt, bei dem leicht Fehler unterlaufen können. Molecool kann optional die um ein Molekül gepackten Nachbar-Moleküle berechnen und darstellen, wenn sie ein bestimmtes Abstandskriterium unterschreiten. Dabei werden eventuell vorhandene Wasserstoffbrücken in tabellierter Form aufgelistet. Der Quellkode von Molecool besteht aus 2135 durch Semikola getrennten Anweisungen.

3.3 rlat4XDS

3.3.1 Motivation

Es ist bei vielen Einkristall-Beugungsexperimenten wichtig, sich einen schnellen Überblick über den reziproken Raum zu verschaffen. Dies gilt besonders auch für Messungen am Synchrotron, wo unter Zeitdruck entschieden werden muss, ob Kristalle geeignet sind für hoch präzise Einkristallmessungen. Mit einem Programm, welches den reziproken Raum darstellt, ist es auf den ersten Blick möglich, einen Einkristall von einem multikristallinen Konglomerat zu unterscheiden, selbst wenn Parameter wie Wellenlänge, Abstand, Strahlenmitte oder Beugungswinkel falsch in das Zellbestimmungsprogramm eingegeben wurden und deshalb keine Zellkonstanten bestimmt werden konnten.

3.3.2 Eigenschaften von rlat4XDS

Rlat4XDS ist ein Programm, das für den Einsatz zusammen mit dem Einkristallprozessierungsprogramm XDS entwickelt wurde. Nach dem Durchlauf des XDS-Programmteils IDXREF, welcher zur Indizierung von Reflexen dient, liegen alle notwendigen Daten für rlat4XDS vor, selbst wenn die Zellkonstan-

tenbestimmung nicht erfolgreich war. Rlat4XDS stellt die Reflexe als farbige Punkte in der Ausbreitungskugel dar. Reflexe, die indiziert wurden, sind blau, die anderen rot. Durch geeignetes Drehen des Blickwinkels auf die Kugel sind die Netzebenen des Gitters leicht zu erkennen. Eisreflexe oder Verzwillingungen liegen klar neben diesen Netzebenen und sind daher leicht zu erkennen. Bei falschen Parametern sind diese Netzebenen nicht mehr eben, sondern systematisch gekrümmt.

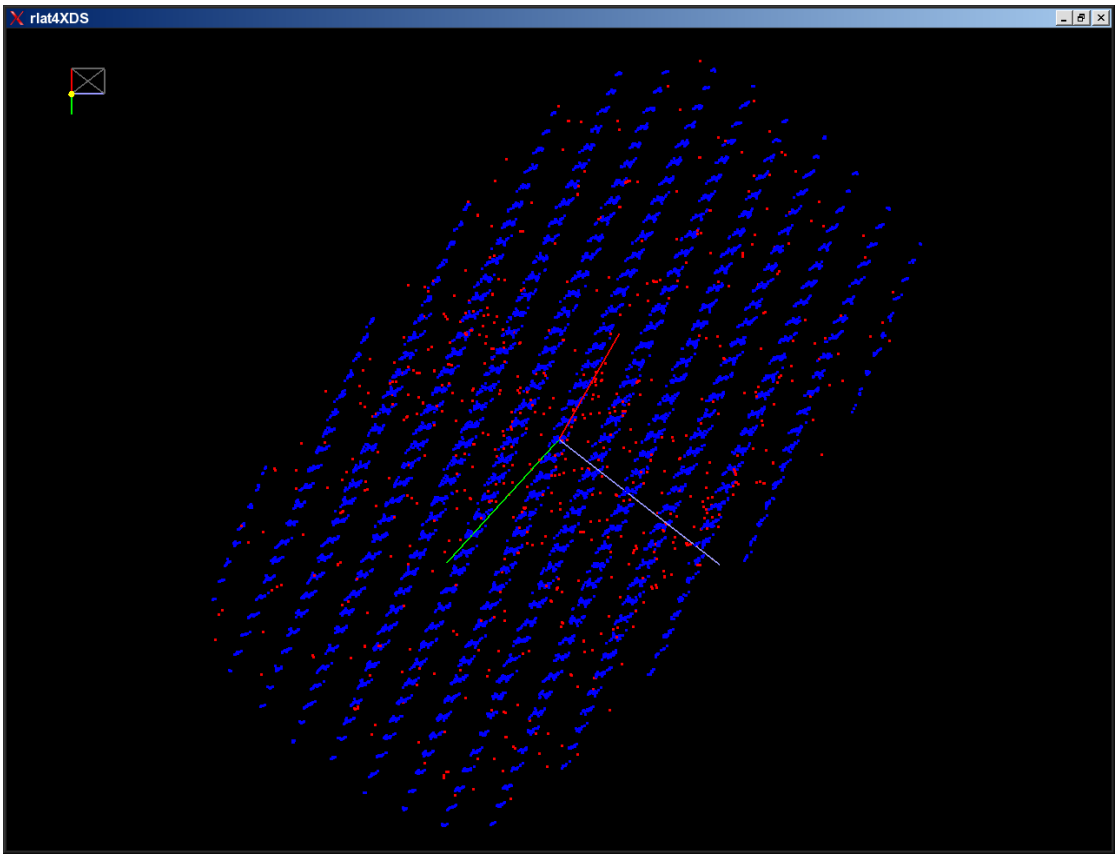

Abbildung 3.4: Screenshot von rlat4XDS. Adenosin: Gut zu erkennen sind die Netzebenen. Die Aufspaltung der Gitterlinen resultiert aus der Ungenauigkeit durch die 3°-Schrittweite.

Eisreflexe, welche durch Vereisung des Kristalls während der Messung entstehen, sind leider manchmal nicht zu vermeiden. Sie sind nur bis zu kleinen Beugungswinkeln als unvollständige Pulverringe zu beobachten und verhindern die Zellbestimmung durch IDXREF, welches leider nicht in der Lage ist, Reflexe unter einer bestimmten Auflösung von der Zellbestimmung auszuschließen. Durch rlat4XDS ist es möglich, Reflexe unterhalb oder oberhalb einer bestimmten Auflösungsgrenze von der Zellbestimmung auszuschließen.

Außerdem ist rlat4XDS dazu in der Lage, alle Reflexe nach der Integration, welche durch unterschiedliche Weglängen durch den „Phosphor“ in ihrer Intensität verfälscht wurden, zu korrigieren. Dazu wird der Wert der Absorption des Phosphors bei senkrechter Einstrahlung $T_\perp$ benötigt und die Formel von Coppens et al. [26] angewandt.

$$I_\perp = I_{obs} \frac{(1 - T_\perp)}{1 - \exp \frac{\ln(T_\perp)}{\cos \alpha}} \tag{3.2}$$

Der Einstrahlwinkel α wird von rlat4XDS aus der Beugungsgeometrie berechnet.