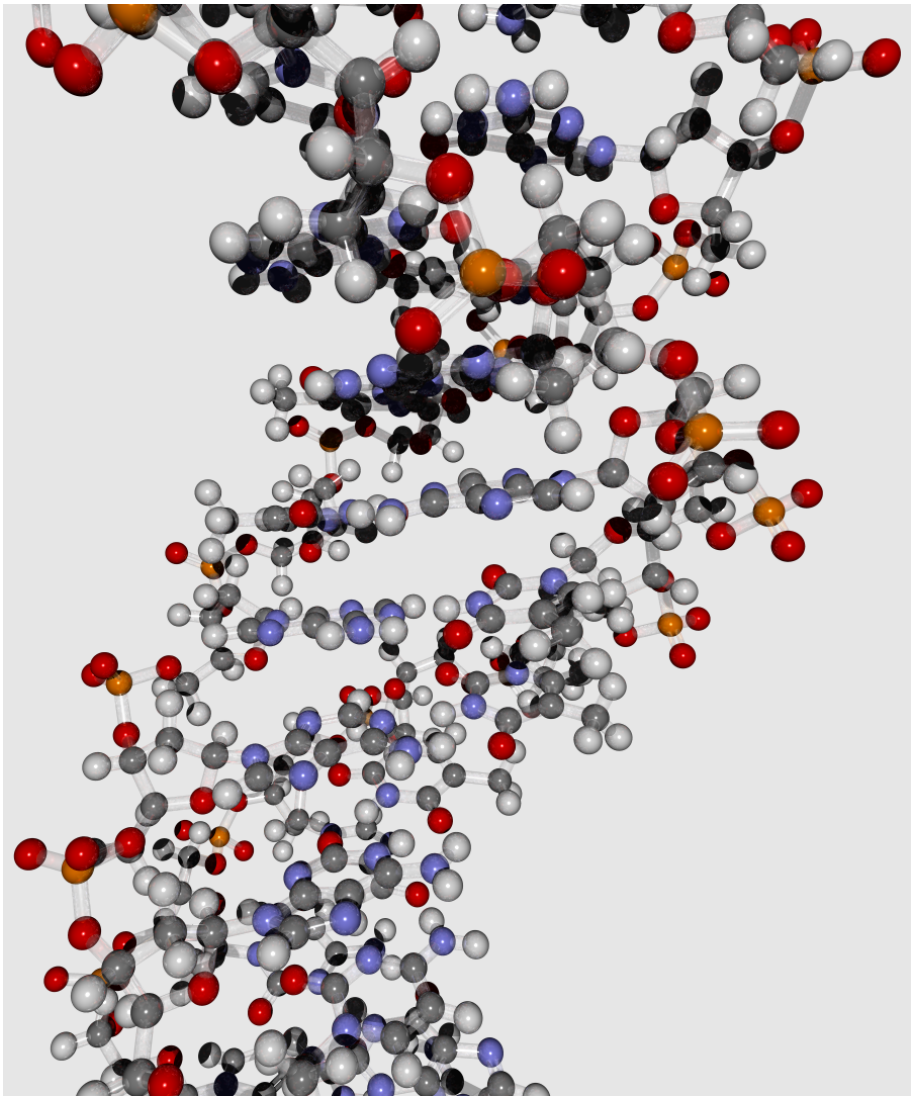
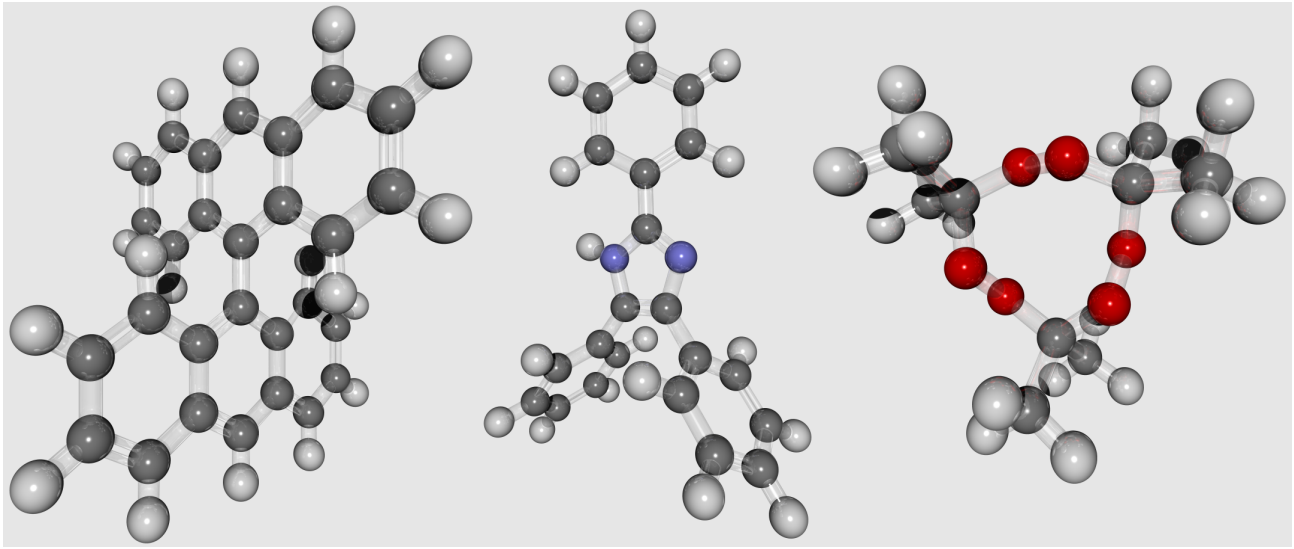
Molekülanimationen mit POV-Ray



DNA-Abschnitt



links: 9,9'-Bianthryl :: mitte: 2,4,5-Triphenylimidazol (Lophin) :: rechts: Acetonperoxid (trimer)

Inhaltsverzeichnis

1. Benötigte Software 3
2. BallView Einstellungen 3
3. Jetzt geht’s los - Eine kleine Übung mit Methan 4
4. Das erste Rendering mit POV-Ray 7
5. Let‘s talk POV-Script 8
6. Glanz und Gloria 10
7. Bindungen mit Glasoptik 11
8. Das INI-File für die Animation 14
9. Danksagung 17

1. Benötigte Software

BallView 1.4.X

POV-Ray 3.7

Texteditor (Notepad++, Phase5 etc.)

Installieren Sie die Software mit einem Administrator-Konto auf ihrem PC.

2. BallView Einstellungen

Starten Sie BallView. Kontrollieren Sie unter »Display → Display Properties« (Ctrl+I) ob die Einstellungen wie in Abb. 1 vorliegen, ggf. müssen Sie diese selbst vornehmen.

Lassen Sie das Fenster geöffnet und kontrollieren Sie unter »Model Options → Ball and Stick«, ob die Werte mit Abb. 2 übereinstimmen. Bestätigen Sie die Einstellungen über »OK« oder »Apply«. Das Fenster kann jetzt geschlossen werden. BallView ist nun konfiguriert und kann wieder beendet werden.

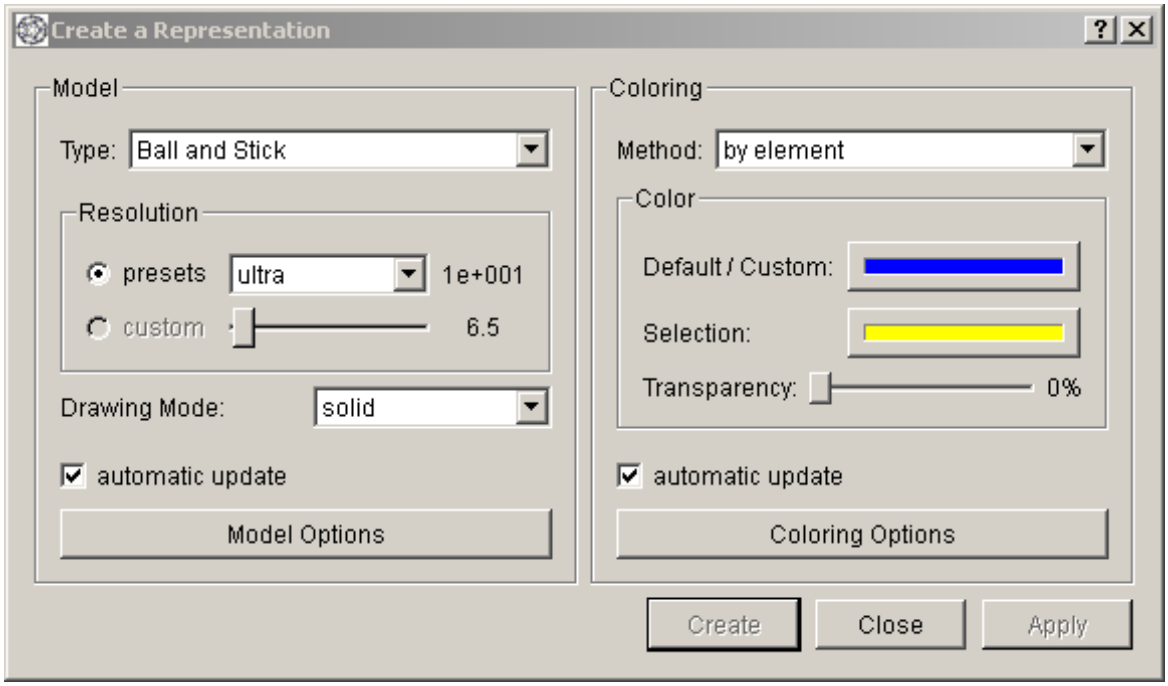


Abb. 1 - Display Properties

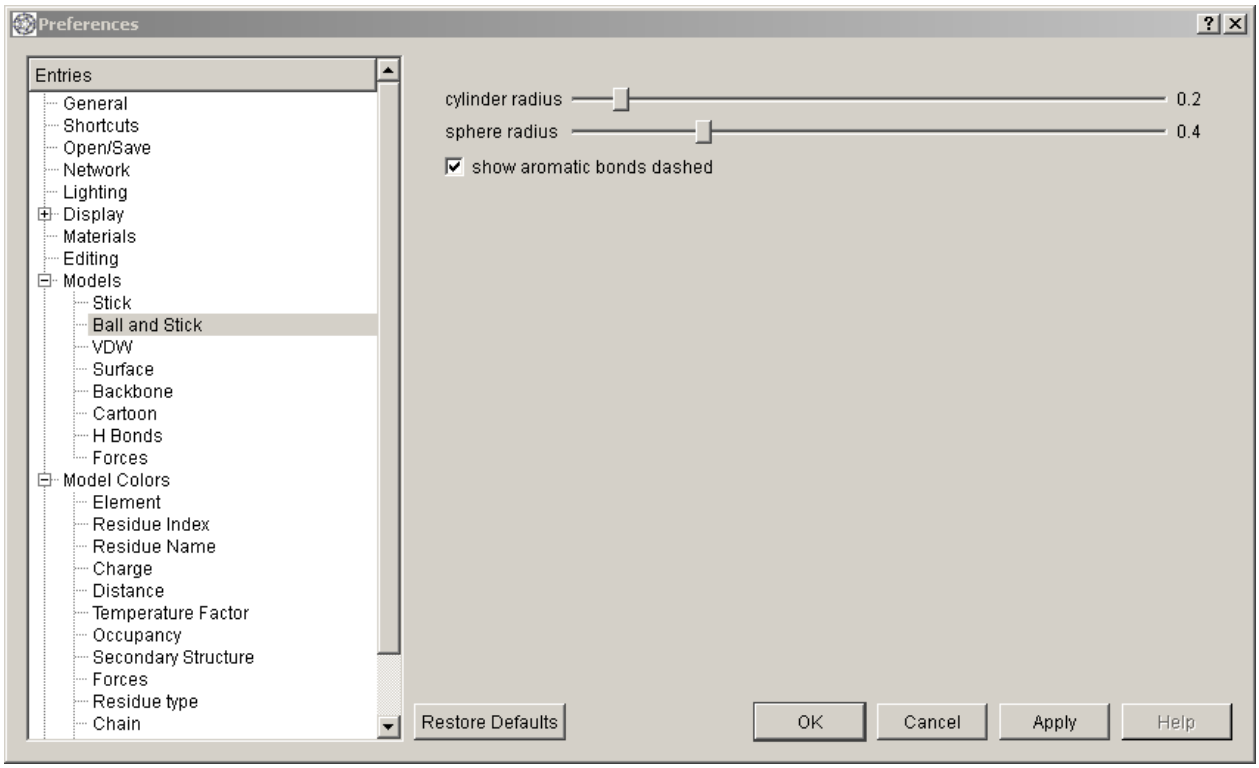


Abb. 2 - Model Options

3. Jetzt geht's los - Eine kleine Übung mit Methan

Rufen Sie die staatliche Webseite des PubChem Projekts (USA) (http://pubchem.ncbi.nlm.nih.gov/) auf (Abb. 3), diese dient als Quelle der Moleküle.

Abb. 3 - PubChem-Startseite

In der Suchmaske können Sie nach dem Molekül Ihrer Wünsche suchen lassen.
Sie können den Namen, die CAS-Nr, molare Masse oder andere Ihnen bekannte Angaben eingeben. PubChem speaks auch a little Bit Deutsch, sodass Sie auch Moleküle in Deutsch eingeben können und trotzdem das gesuchte Molekül finden werden. Die schnellste Methode ist jedoch die Suche über die CAS-Nummer. Für Methan wäre das: 74-82-8
Geben Sie die Nummer ein und klicken mit der linken Maustaste auf »Go« oder drücken Sie »Return«.

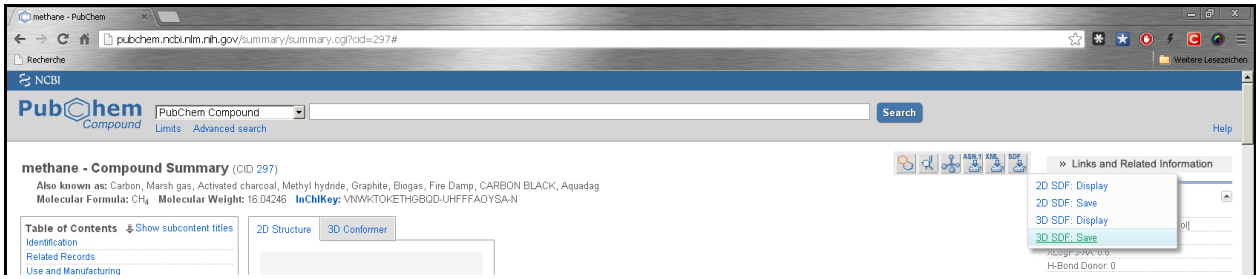


Abb. 4 - Methan als 3D-SDF-Datei speichern

Nun sollten die Informationen über »methane« zu sehen sein. Klicken Sie rechts oben auf das Icon »SDF« und wählen wie in Abb. 4 zusehen »3D SDF: Save«. Es wird jetzt eine Datei mit dem Namen »CID_297.sdf« gespeichert. Es empfiehlt sich, die Datei umzubenennen in »Methan.sdf«, dadurch behalten Sie den Überblick, wenn Sie mehrere Moleküle speichern wollen.

Schließen Sie den Browser und starten Sie BallView. Schließen Sie alles, was nicht benötigt wird in diesem Fenster, siehe Abb. 5.

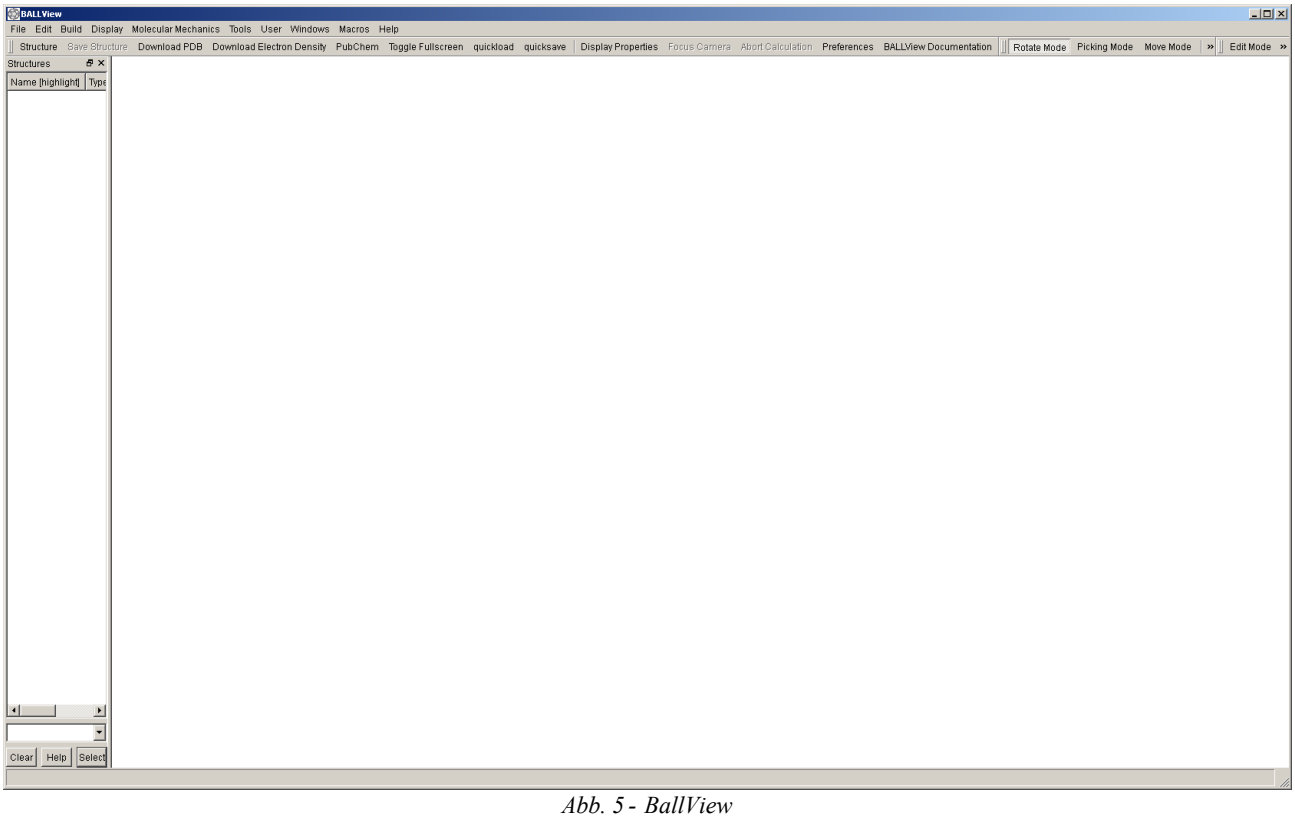


Abb. 5 - BallView

Öffnen Sie die Datei »Methan.sdf« über »File → Open → Structure« oder drücken Sie »Ctrl+O«. Das Molekül sollte nun wie in Abb. 6 dargestellt werden.

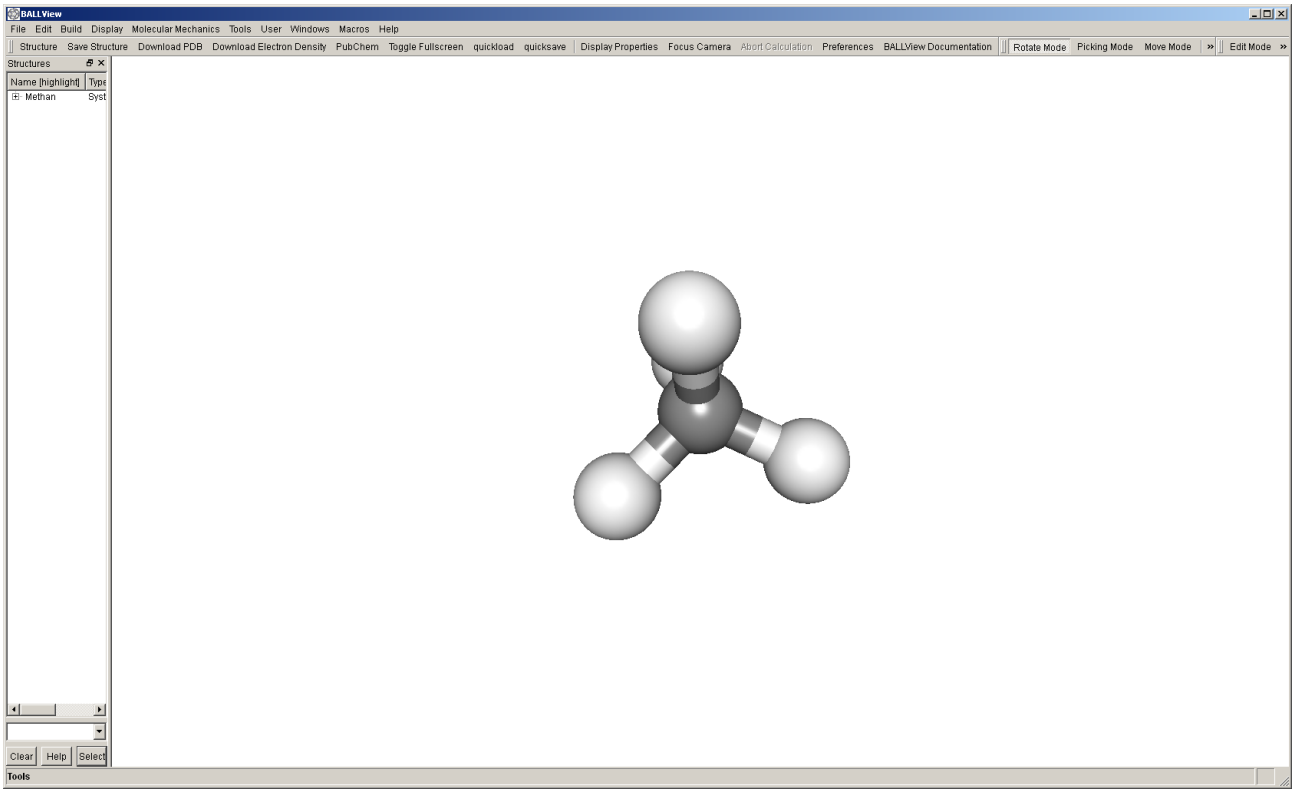


Abb. 6 - Das Rohmaterial

Das Molekül kann mittels Mausrad vergrößert bzw. verkleinert werden.

Aber Vorsicht: In den SDF-Dateien sind die Moleküle irgendwo im Raum platziert, d. h., das Kohlenstoffatom befindet sich nicht unbedingt auf der Koordinate 0.0.0! Das Molekül soll am Ende um die X-Achse rotiert werden, wenn Sie das Molekül verdrehen, kann das Endprodukt evtl. nicht wie gewünscht aussehen. Vergrößern Sie es auch nicht zu stark, sonst laufen die Wasserstoffatome beim Drehen aus dem Bild!

Exportieren Sie jetzt das Molekül als POVRay Scene über »File → Export → POVRay Scene« (Ctrl+Y), als »Methan.pov«. BallView kann wieder geschlossen werden.

4. Das erste Rendering mit POV-Ray

Starten Sie POV-Ray 3.7 und öffnen Sie über »Open« die Datei »Methan.pov«. Kopieren Sie die Werte wie in Abb. 7 in das rotmarkierte Feld und klicken Sie auf »Run«. Das Molekül wird jetzt berechnet, das Endergebnis sehen Sie in Abb. 8.

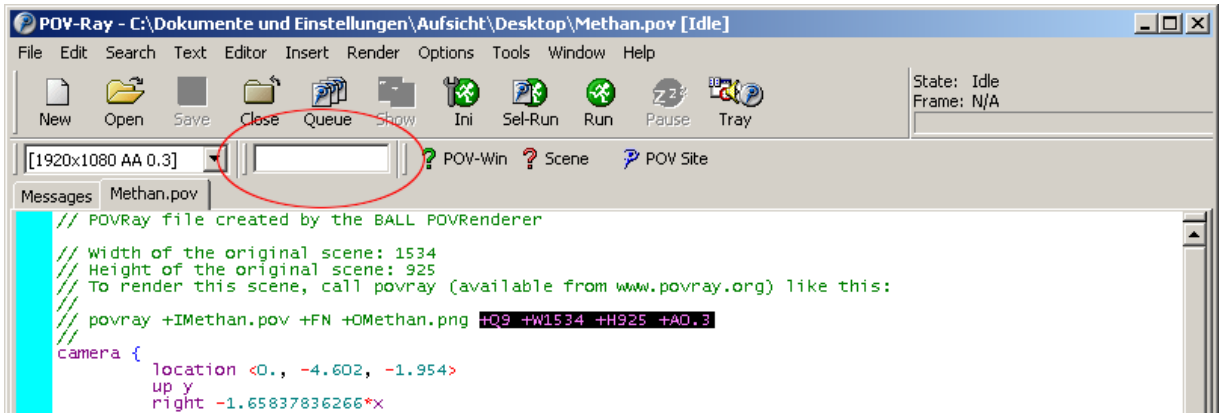


Abb. 7 - Die markierten Werte kopieren

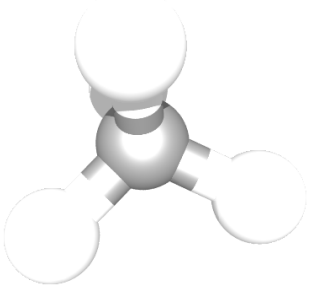


Abb. 8 - Ergebnis des Renderings

Sieht aus wie »weißer Adler auf weißem Grund«. Außerdem hat POV-Ray zwei Warnungen ausgegeben:

1. Es fehlt die Angabe zur Gamma-Korrektur: »Parse Warning: assumed_gamma not specified, so gamma_correction is turned off...«
2. Keine Definition, für welche Version das Script geschrieben wurde: »Parse Warning: This scene did not contain a #version directive...«

Diese Warnungen und den »weißen Adler auf weißem Grund« wollen wir jetzt loswerden.

5. Let‘s talk POV-Script

Sie können die Datei in POV-Ray oder in einem Editor Ihrer Wahl ändern, was wesentlich komfortabler ist. Ich verwende z. B. mirabyte Web Architect. Dieses Programm kann nämlich Textpassagen in allen geöffneten Dokumenten oder in Dokumenten eines Verzeichnisses ändern (Globales Suchen & Ersetzen). Natürlich kann man wählen, welche Dateien geändert werden sollen. Dies geschieht über die Dateiendung. Diese Funktion ist goldwert, wenn Sie mehrere Dateien gleichzeitig bearbeiten möchten. Zurück zum Thema.
Der obere Teil der Datei wimmelt nur so von »//«. Mit »//« werden Kommentare eingeleitet, POV-Ray beachtet die anschließenden Zeichen in dieser Zeile nicht mehr.
Wir legen jetzt fest, für welche Version von POV-Ray das Script gedacht ist. Gehen Sie mit dem Cursor in die erste Zeile »// POVRay file created by the BALL POVRenderer« und schreiben »#version 3.7;« vor den »//« und drücken Sie anschließend »Return«. Das Script sollte nun so aussehen:

```
#version 3.7;
// POVRay file created by the BALL POVRenderer
...
```

Die von BallView exportierten POV-Ray-Scripte haben alle einen kleinen »Fehler«. Benutzt man die Scripte, ohne Änderungen im Quelltext, für Animationen, dann erscheinen auf den Molekülen seltsame viereckige oder streifenartige »Blitzer« bzw. Artefakte. Folgender Quellcode verursacht das Problem:
global_settings { max_trace_level 99 radiosity { brightness 0.6 } }

Um dies zu verhindern, wird die Zeile geändert zu:
global_settings { max_trace_level 99 assumed_gamma x.x }

Das x.x wird bei Windows-Betriebssystemen mit 2.2, bei Mac OS X <10.6 mit 1.8 ersetzt. Für Mac OS X ab 10.6 gilt auch der Wert für Windows-Systeme. Durch diese Änderung wurden zwei Probleme beseitigt:

1. Die Meldung (Warning) des nicht definierten Gamma-Wertes.
2. Die »Blitzer« und Artefakte tauchen auch nicht mehr auf.

Nun zum Hintergrund. Dieser wird durch die Anweisung »background« definiert.
Durch »background { <1., 1., 1., 0., 0.> }« ist definiert, dass der Hintergrund weiß sein soll. Einen schwarzen Hintergrund erhält man durch »background { <0., 0., 0., 0., 0.> }« oder Sie kommentieren die Anweisung einfach aus »// background { <1., 1., 1., 0., 0.> }«, so habe ich es gemacht. Meine Animationen habe ich durch PNG-Bilder, mit transparentem Hintergrund, erstellt. Dies wird durch das INI-File erledigt, wie das geht, erkläre ich später.

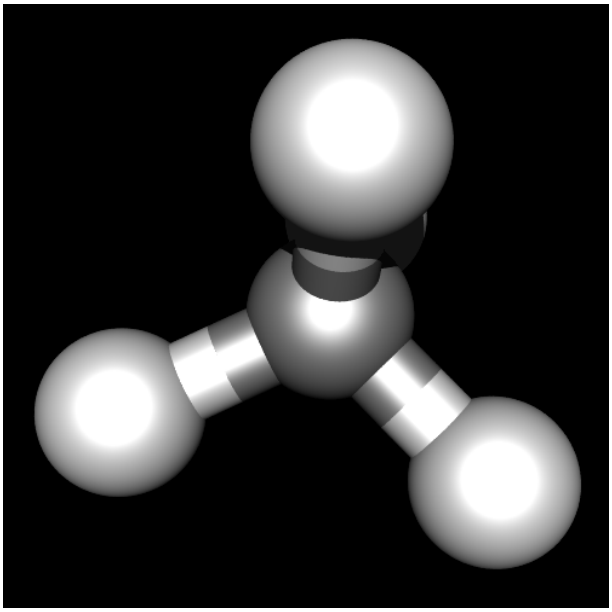


Abb. 9 - Endergebnis

Nun müssen wir nur noch den Befehl für die Rotation einfügen. Gehen Sie dazu in die letzte Zeile des POV-Script. Fügen Sie vor das »}« »rotate 360*clock*x« für eine Rotation um die x-Achse ein. Wollen Sie eine Rotation um die y- oder z-Achse, dann ersetzen Sie das »x« durch »y« oder »z«. Für eine Rotation um mehrere Achsen müssen die Rotate-Anweisungen hintereinanderstehen, z. B. Rotation um x- und y-Achse:

```
rotate 360*clock*x
rotate 360*clock*y}
```

Das Script würde jetzt eigentlich schon für eine Animation funktionieren, aber gut aussehen würde die Animation nur bei einer Renderauflösung, in meinem Beispiel, von 1534 × 925 Pixel. Bei einer anderen Auflösung würde das Bild gestreckt oder gestaucht werden. Um dies zu verhindern, ändern Sie im oberen Teil die Anweisung »right Wert*x« in »right x*image_width/image_height // Wert*x«. Nun ist es egal, in welcher Auflösung das Bild erstellt werden soll. Bei größeren Molekülen kann es jetzt aber passieren, dass Teile des Moleküls aus dem Bild herauslaufen! Hier muss man dann selbst etwas mit der Kameraposition spielen. Der gesamte Code für die Kamera sieht so aus:

```
camera {
        location <0., -4.602, -1.954>
        up y
        right x*image_width/image_height //-1.65837836266*x
        angle 95.7414398193
        sky <1., 0., 0.>
        look_at <0., 0., 0.>
}
```

Die Kameraposition wird durch »location <x, y, z>« definiert. Läuft das Molekül stark aus dem Bild, ändern Sie den Wert für z um 1 oder mehr, also »location <0., -4.602, -1.954>« würde dann zu »location <0., -4.602, -2.954>«. Wenn es nicht so schlimm ist, um 0.5 »location <0., -4.602, -2.454>«. Die Werte können positiv oder negativ sein, dies hängt von der verwendeten 3D-SDF-Datei ab.

6. Glanz und Gloria

Jetzt verändern wir noch das Aussehen der Moleküle und Bindungen, damit diese einen schönen Glanz und Spiegelungen erhalten.
Gehen Sie in die Zeile, die folgenden Code enthält:
#declare BALLFinish = finish { specular 1 roughness 0.0674312 diffuse 1 ambient 0.1 }
BallView verwendet immer den oben aufgeführten Code, bei einem POV-Ray-Export, also ein hervorragender Kandidat für Globales suchen & ersetzen.

Ändern Sie den Code in:

```
#declare BALLFinish = finish {
                      ambient 0.1
                      diffuse 0.9
                      brilliance 2.0
                      specular 0.8
                      reflection 0.6
                      roughness 0.0003
                      phong 1 phong_size 120
                     }
```

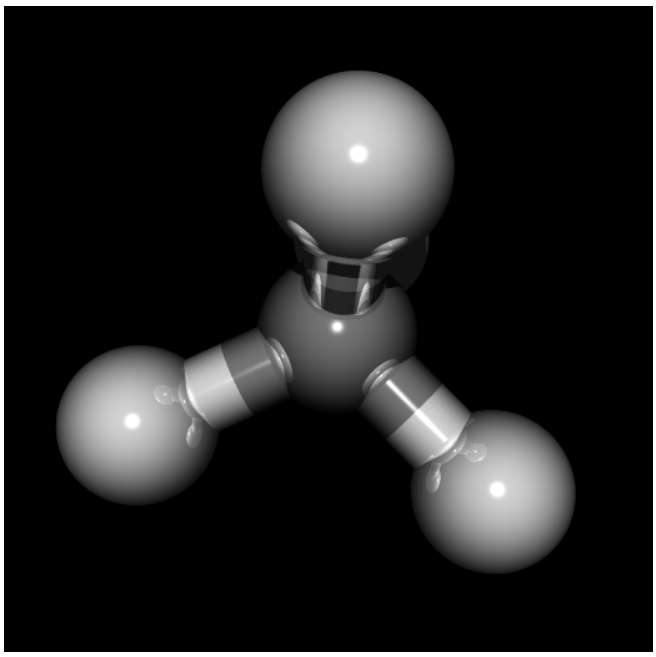


Abb. 10 - Glanz und Spiegelungen

7. Bindungen mit Glasoptik

Das Methan sieht ja schon ganz gut aus, aber ich wollte die Bindungen der Atome gerne als durchgehende »Glasröhren« dargestellt haben. Momentan ist eine Hälfte der Bindung weiß, die andere grau.

Suchen Sie in der POV-Datei folgenden Abschnitt:

```
#declare c1 = <0.564, 0.564, 0.564, 0., 0.>;
#declare c0 = <1., 1., 1., 0., 0.>;
// Representation Ball and Stick Methan
```

Ändern Sie den Code in:

```
#declare c1 = <0.364, 0.364, 0.364, 0., 0.>;
#declare c0 = <1., 1., 1., 0., 0.>;
#declare c10 = <1., 1., 1., 0.8, 0.>;
// Representation Ball and Stick Methan
```

Was macht diese Anweisung?

Nun »#declare« ist eine Deklaration, in diesem Fall legt sie die Farbe und Transparenz fest. c1 ist unser Kohlenstoff, der 1. Wert für Rot, 2. Wert für Grün, 3. Wert für Blau, der 4. Wert ist die Transparenz. Bei c0 (Wasserstoff) ist es ebenso. Mit c10 habe ich eine neue Deklaration eingefügt, Farbe weiß und Transparenz von 80 %. Diese soll auf die Bindungen angewendet werden.

Der Code für das Molekül sieht momentan so aus:

```
union {
Sphere(<0., 0., 0.>, 0.4, c1)
Sphere(<0.554, 0.799, 0.496>, 0.4, c0)
Sphere(<0.683, -0.813, -0.253>, 0.4, c0)
Sphere(<-0.778, -0.373, 0.669>, 0.4, c0)
Sphere(<-0.459, 0.387, -0.912>, 0.4, c0)
Tube(<0.162, 0.234, 0.145>, <0.277, 0.399, 0.248>, 0.2, c1)
Tube(<0.277, 0.399, 0.248>, <0.391, 0.565, 0.351>, 0.2, c0)
Tube(<0.2, -0.238, -0.074>, <0.341, -0.406, -0.126>, 0.2, c1)
Tube(<0.341, -0.406, -0.126>, <0.483, -0.575, -0.179>, 0.2, c0)
Tube(<-0.228, -0.109, 0.196>, <-0.389, -0.186, 0.334>, 0.2, c1)
Tube(<-0.389, -0.186, 0.334>, <-0.55, -0.264, 0.473>, 0.2, c0)
Tube(<-0.134, 0.113, -0.267>, <-0.229, 0.193, -0.456>, 0.2, c1)
Tube(<-0.229, 0.193, -0.456>, <-0.324, 0.273, -0.644>, 0.2, c0)
rotate 360*clock*x}
```

Sphere beschreibt die Atome (Kugeln). Tube (Röhre) die Bindungen.

Ändern Sie den Code für Tube von c0 bzw. c1 auf c10 ab:

```
union {
Sphere(<0., 0., 0.>, 0.4, c1)
Sphere(<0.554, 0.799, 0.496>, 0.4, c0)
Sphere(<0.683, -0.813, -0.253>, 0.4, c0)
Sphere(<-0.778, -0.373, 0.669>, 0.4, c0)
Sphere(<-0.459, 0.387, -0.912>, 0.4, c0)
Tube(<0.162, 0.234, 0.145>, <0.277, 0.399, 0.248>, 0.2, c10)
Tube(<0.277, 0.399, 0.248>, <0.391, 0.565, 0.351>, 0.2, c10)
Tube(<0.2, -0.238, -0.074>, <0.341, -0.406, -0.126>, 0.2, c10)
Tube(<0.341, -0.406, -0.126>, <0.483, -0.575, -0.179>, 0.2, c10)
Tube(<-0.228, -0.109, 0.196>, <-0.389, -0.186, 0.334>, 0.2, c10)
Tube(<-0.389, -0.186, 0.334>, <-0.55, -0.264, 0.473>, 0.2, c10)
Tube(<-0.134, 0.113, -0.267>, <-0.229, 0.193, -0.456>, 0.2, c10)
Tube(<-0.229, 0.193, -0.456>, <-0.324, 0.273, -0.644>, 0.2, c10)
rotate 360*clock*x}
```

Das Ergebnis sehen Sie in Abb. 11, aber da sind Berührungsflächen (rote Markierung), die wollen wir nicht haben.

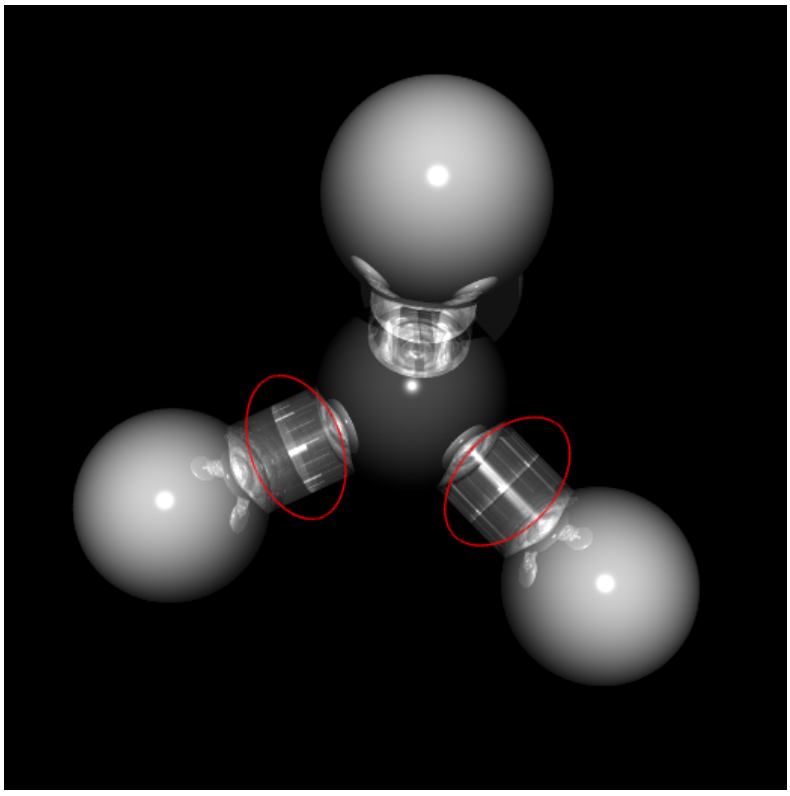


Abb. 11 - Das Ziel ist nah

Das Methanmolekül hat vier Bindungen, die in unserem Script mit acht Tubes gezeichnet werden. Also lassen Sie uns aus acht Tubes nur vier machen.

Die Tube-Anweisung beschreibt Folgendes:
Tube(<von x,y,z>, <nach x,y,z>, Durchmesser, Definition)

Da die Tubes als Paar beschrieben sind, erscheint in der darunter folgenden Tube-Anweisung das »nach« in der Folgezeile als »von«! Verwirrend, oder?

Beispiel:

```
Tube(<0.162, 0.234, 0.145>, <0.277, 0.399, 0.248>, 0.2, c10)
Tube(<0.277, 0.399, 0.248>, <0.391, 0.565, 0.351>, 0.2, c10)
Tube(<0.2, -0.238, -0.074>, <0.341, -0.406, -0.126>, 0.2, c10)
Tube(<0.341, -0.406, -0.126>, <0.483, -0.575, -0.179>, 0.2, c10)
Tube(<-0.228, -0.109, 0.196>, <-0.389, -0.186, 0.334>, 0.2, c10)
Tube(<-0.389, -0.186, 0.334>, <-0.55, -0.264, 0.473>, 0.2, c10)
Tube(<-0.134, 0.113, -0.267>, <-0.229, 0.193, -0.456>, 0.2, c10)
Tube(<-0.229, 0.193, -0.456>, <-0.324, 0.273, -0.644>, 0.2, c10)
```

Wenn Sie jetzt die rot markierten Bereiche löschen, erhalten Sie am Ende die vier gewünschten Tubes:

```
Tube(<0.162, 0.234, 0.145>, <0.277, 0.399, 0.248>, 0.2, c10)
Tube(<0.277, 0.399, 0.248>, <0.391, 0.565, 0.351>, 0.2, c10)
Tube(<0.2, -0.238, -0.074>, <0.341, -0.406, -0.126>, 0.2, c10)
Tube(<0.341, -0.406, -0.126>, <0.483, -0.575, -0.179>, 0.2, c10)
Tube(<-0.228, -0.109, 0.196>, <-0.389, -0.186, 0.334>, 0.2, c10)
Tube(<-0.389, -0.186, 0.334>, <-0.55, -0.264, 0.473>, 0.2, c10)
Tube(<-0.134, 0.113, -0.267>, <-0.229, 0.193, -0.456>, 0.2, c10)
Tube(<-0.229, 0.193, -0.456>, <-0.324, 0.273, -0.644>, 0.2, c10)
```

Der fertige Code für vier Tubes:

```
Tube(<0.162, 0.234, 0.145>, <0.391, 0.565, 0.351>, 0.2, c10)
Tube(<0.2, -0.238, -0.074>, <0.483, -0.575, -0.179>, 0.2, c10)
Tube(<-0.228, -0.109, 0.196>, <-0.55, -0.264, 0.473>, 0.2, c10)
Tube(<-0.134, 0.113, -0.267>, <-0.324, 0.273, -0.644>, 0.2, c10)
```

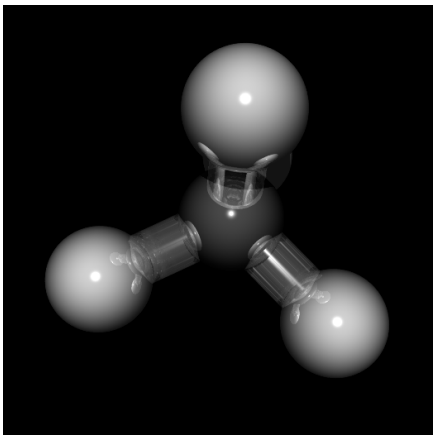


Abb. 12 - Endlich fertig

8. Das INI-File für die Animation

Die INI-Datei enthält die Einstellungen wie POV-Ray das POV-Script rendern soll.

Meine INI-Files sehen so aus:

```
Antialias=On

Sampling_Method=2
Antialias_Threshold=0.3
Antialias_Depth=4
Input_File_Name=»Filename«.pov
Width=1920
Height=1080
Output_File_Type=N
Output_Alpha=on

Initial_Frame=1
Final_Frame=360
Initial_Clock=0
Final_Clock=1

Pause_when_Done=off
Cyclic_Animation=on
Display=Off
```

Um in INI-Dateien einen Kommentar zu schreiben, muss man einen »;« verwenden, alles hinter diesem Zeichen wird in dieser Zeile nicht mehr beachtet.

Nun zu den einzelnen Anweisungen.

Antialias=On oder Off

Kantenglättung ein- oder ausgeschaltet.

Sampling_Method=n

Wertmöglichkeiten 1 oder 2. Die Unterschiede sehen Sie bei Antialias_Depth.

Antialias_Threshold=n.n

Der Wert gibt an, ab welcher Farbdifferenz das Supersampling greifen soll. Der Standardwert ist 0.3, bei 0.0 wird Supersampling für jeden Pixel ausgeführt, dies würde zu extremen Berechnungszeiten führen.

Antialias_Depth=n

Definiert die Anzahl der Supersamples der gewählten Samplingmethode. Werte zwischen 1 und 9 sind zulässig.

Wert n =	Verwendete Supersamples bei	
	Sampling_Method=1	Sampling_Method=2
1	1	9
2	4	25
3	9	81
4	16	289
5	25	1089
6	36	4225
7	49	16641
8	64	66049
9	81	263169

Input_File_Name=»Filename«.pov

Dateiname des POV-Scripts. Bei dieser Art der Angabe müssen sich POV- und INI-File im selben Verzeichnis befinden.

Width & Height

Width = Breite des zu berechnenden Bildes.
Height = Höhe des zu berechnenden Bildes.

Output_File_Type=x

Angabe, in welchem Format das Bild gespeichert werden soll.

Mögliche Optionen:
B = Universal **B**itmap image file Format
C = **C**ompressed Targa-24 Format
E = Open**E**XR High Dynamic-Range Format
H = Radiance **H**igh Dynamic-Range Format
J = **J**PEG Format (kann zu Artefakten im Bild führen)
N = P**N**G Format
P = Unix **P**PM Format
S = **S**ystem-specific Format (Bei Windows BMP)
T = Uncompressed **T**arga-24 format

Output_Alpha=on/off

Alphakanal ein oder aus. Durch diese Anweisung kann man einen transparenten Hintergrund ausgeben lassen.

Initial_Frame=n

Legt fest, mit welchem Bild die Berechnung beginnt. Bei einer Rotation um 360° ist der Wert 1.

Final_Frame=n

Legt fest, mit welchem Bild die Berechnung endet. Bei einer Rotation um 360° ist der Wert 360.

Initial_Clock=n

Dies ist eine Zeitangabe und beginnt bei 0, für Rotationen um 360°.

Final_Clock=n

Dies ist eine Zeitangabe und endet bei 1, für Rotationen um 360°.

Pause_when_Done=on/off

Ist diese eingeschaltet, dann wartet POV-Ray auf eine Usereingabe (Tastendruck), bevor die Berechnung mit dem nächsten Bild fortgesetzt wird. Ich schalte das immer aus (off).

Cyclic_Animation=on

Dies veranlasst POV-Ray, die Final_Clock automatisch zu justieren, damit die Animation flüssig erscheint.

Display=on/off

Das Renderfenster anzeigen oder ausblenden. Bei älteren Computern empfiehlt es sich, das Fenster auszublenden, also ist hier »off« die beste Wahl.

Bei meinen INI-Dateien werden die erzeugten Bilder im gleichen Verzeichnis gespeichert in dem sich die INI- und POV-Datei befindet. Mit »Output_File_Name=« kann der Ausgabepfad angegeben werden. Die Bilder werden fortlaufend durchnummeriert: Dateiname001.xxx, Dateiname002.xxx etc.

Nun können Sie die Animation rendern lassen. Die Berechnungszeit ist sehr stark abhängig von den Werten für:
Sampling_Method, Antialias_Threshold, Antialias_Depth, Width & Height

Bei einer 360°-Rotationsanimation von 12 Basenpaaren eines DNA-Abschnitts habe ich folgende Werte verwendet: Sampling_Method=2, Antialias_Threshold=0.3, Antialias_Depth=9, Width=1920, Height=1080

Gerendert wurde das Ganze auf einem Rechner mit 3 × Intel Xeon Quadcore 3GHz Prozessoren, also mit 12 realen Kernen und 128 GB RAM! Die Prozessorauslastung lag bei 100%. Die Berechnung der 360 Einzelbilder dauerte 26 Stunden und 23 Minuten!

9. Danksagung

Herrn Friedrich Lohmüller danke ich, für seine POV-Ray-Tutorials auf http://www.f-lohmueller.de und für seine persönliche Hilfe bei meinem Projekt, das ohne ihn so nicht zustande gekommen wäre.

Dem gläsernen Forscherlabor im Zentrum neue Technologien (Deutsches Museum) ein herzliches Danke, für die Benutzung des schnellsten Computers auf der Museumsinsel.

POV-Ray-Skripts für Molekülanimationen