

Linköpings universitet | Institutionen för fysik, kemi och biologi
Examensarbete på grundnivå, 16hp | Programområde: Fysik
Vårterminen 2018 | LIU-IFM/LITH-EX-G--18/3551--SE

Design och implementering av system för interaktiv visualise- ring av elektronstrukturdata

**Anders Rehult
Andreas Kempe
Marian Brännvall
Viktor Bernholtz**

Handledare : Johan Jönsson
Examinator : Per Sandström

Design och implementering av system för interaktiv visualisering av elektronstrukturdata

Redaktör: Viktor Bernholtz

Version 0.3

Status

Granskad	VB	2018-05-25
Godkänd		

Förord

Vi vill rikta ett stort tack till vår handledare Johan Jönsson som haft stort tålmod med alla våra kursrelaterade och icke-kursrelaterade frågor under arbetets gång. Vi vill även tacka vår beställare Rickard Armiento som svarat på alla möjliga frågor och varit behjälplig under projektet.

Sammanfattning

Att kunna göra olika typer av visualiseringar baserat på elektronstrukturberäkningar är viktigt, bland annat för att kunna presentera resultat från forskning inom materialfysik på ett enklare sätt. Detta dokument är en slutrapport för kandidatprojektet i visualisering av elektronstrukturer. Slutrapporten beskriver bland annat projektets problemformulering och resultat, dessutom finns en beskrivning av genomförandet av projektet samt en teknisk beskrivning av det visualiseringssverktyg som tagits fram.

PROJEKTIDENTITET

2018/VT, Linköpings universitet
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	viii
1 Inledning	1
1.1 Parter	1
1.2 Användning	1
1.3 Begränsningar	2
1.4 Definitioner	2
2 Problemformulering	2
3 Kunskapsbas	3
4 Fasplan	3
4.1 Före-fas	3
4.2 Under-fas	3
4.3 Efter-fas	4
5 Fördjupningsarbeten	4
5.1 Fermi-ytor	4
5.2 Visualisering av volymdata	4
6 Teknisk beskrivning	4
6.1 Datakonvertering	4
6.1.1 HDF5	5
6.1.2 VASP	5
6.2 Visualisering	5
6.2.1 Utritning	6
6.2.2 Användarindata	6
6.2.3 Interaktivitet	8
6.2.4 Kristallstruktur	8
6.2.5 Elektrontäthet	8
6.2.6 ELF	8
6.2.7 Tillståndstäthet	8
6.3 Sammanlänkning av olika visualiseringar	8
7 Resultat	8
7.1 Krav på systemet	9
7.2 Användning	14

8 Slutsats	14
8.1 Tekniskt	14
8.2 Utförande	15
8.3 Framtida arbete	15
Referenser	16
Bilaga A Användarmanual	17
Bilaga B Designspecifikation	33
Bilaga C Fördjupningsarbete - Fermi-ytor	62
Bilaga D Fördjupningsarbete - Visualisering av volymsdata	80
Bilaga E Gruppkontrakt	107
Bilaga F Kravspecifikation	108
Bilaga G Projektdirektiv	128
Bilaga H Projektplan	131
Bilaga I Systemskiss	153
Bilaga J Tidplan	163
Bilaga K Teknisk dokumentation	164
Bilaga L Naturvetenskaplig undersökning	202

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-05-14	Första utkast.	Projektgruppen	PL
0.2	2018-05-22	Andra utkast.	Projektgruppen	VB
0.3	2018-05-25	En del korrigeringar efter återkoppling från beställare.	Projektgruppen	

1 Inledning

Detta dokument är en slutrappart för kandidatprojektet i visualisering av elektronstrukturer. Visualisering av elektronstrukturer är ett av projekten i kurserna TFYA75 vid Linköpings universitet.

Inom materialfysik är elektronstrukturberäkningar viktiga teoretiska verktyg. Detta för att få en så bra förståelse som möjligt av hur olika material är uppbyggda, bland annat vad gäller kristallers egenskaper sedda ur ett kvantmekaniskt perspektiv. Det kan exempelvis handla om att man vill ta reda på olika materials egenskaper vad gäller värmeförståelse, strömledningsförmåga, etc.

För att öka förståelsen samt för att enklare kunna presentera resultat från forskning inom området underlättar det att kunna göra olika typer av visualiseringar baserat på elektronstrukturberäkningar.

I många av de system som används för att utföra dessa beräkningar, exempelvis programmet VASP som utför elektronstruktur-och-molekylärdynamikberäkningar, ges ingen eller begränsad möjlighet till detta. Vidare finns begränsningar i de system som finns tillgängliga idag vad gäller effektivetet, standardisering och tillgång till visualiseringsfunktioner.

Projektgruppen har vidareutvecklat och utökat det system som togs fram av 2017 års projektgrupp. Utöver det konkreta målet med utvecklingen av mjukvara för visualisering ska även projektet ge projektmedlemmarna erfarenhet av att arbeta i projekt och utöka deras förmåga till analytiskt och fysikaliskt tänkande för att ge värdefull erfarenhet inför arbetslivet.

Kappan ger en översikt över hela projektet och består av följande delar:

- **Problemformulering**, avsnitt 2, och **Kunskapsbas**, avsnitt 3, beskriver själva förarbetet i projektet.
- **Fördjupningsarbeten**, se avsnitt 5, beskriver kortfattat de fördjupningsområden som projektgruppen behandlat.
- **Fasplan**, avsnitt 4, beskriver överskådligt hur projektet har genomförts.
- **Teknisk beskrivning**, avsnitt 6, beskriver överskådligt hur systemet är uppbyggt.
- **Resultat**, avsnitt 7, och **Slutsats**, avsnitt 8, beskriver vilka krav som är uppfyllda och hur det slutgiltiga resultatet blev samt sammanfattar projektet i sin helhet.

1.1 Parter

Rickard Armiento har beställt systemet som är beskrivet i slutrapparten. Medlemmarna i projektgruppen, som är listade under rubriken projektidentitet ovan, är mottagare av denna beställning och har haft i uppgift att implementera systemet. Projektgruppens handledare är Johan Jönsson.

1.2 Användning

Denna produkt kommer användas vid Linköpings universitet för att analysera data från elektronstrukturberäkningar.

1.3 Begränsningar

I projektet har visualiseringssverktyget Inviwo och programmeringspråken Python och C++ användts. Det har inte utretts huruvida andra verktyg hade varit lämpligare.

1.4 Definitioner

C++ är ett programmeringsspråk. [1]

I Inviwo används C++ för att skriva programkod till processorer.

Fermi-yta är, för elektroners k-punkter i reciproka rymden, isoytan där elektronernas energi är lika med Fermi-energin. [3]

HDF5 är ett filformat som kan hantera stora mängder data. [2]

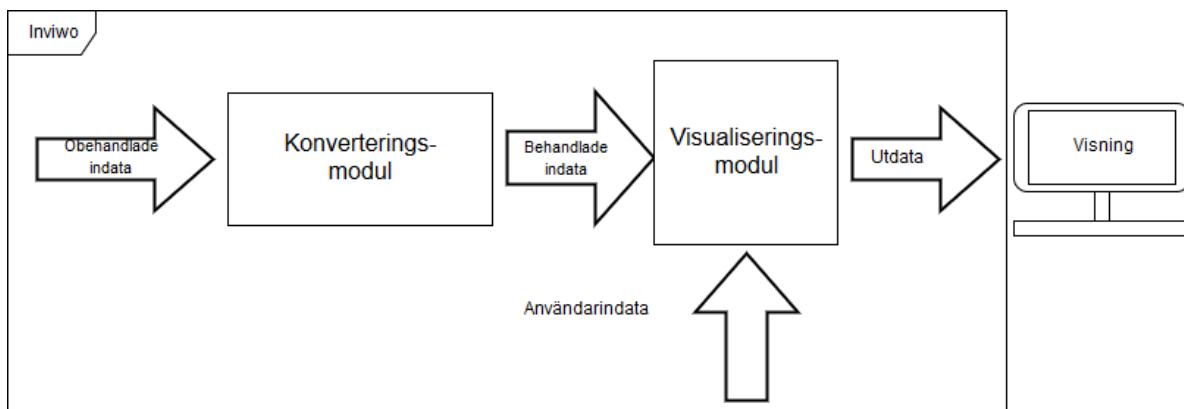
Inviwo (Interactive Visualization Workshop) är programvara för visualisering som tillhandahåller en nätverksredigerare för designen av dataflödesnätverk. Noderna i dessa dataflödesnätverk kallas processorer. Indata till nätverken behandlas i dessa processorer och utdata genereras. [4]

Python är ett programmeringsspråk [5]. I Inviwo används Python för att knyta samman processorer.

2 Problemformulering

I projektdirektivet, se appendix G, ges en översiktlig bild av de krav som kan komma att ställas på systemet. Utifrån projektdirektivet har projektgruppen skapat en kravspecifikation, se appendix F, med de krav som ställs på systemet och även de generella krav som ställs på projektet och dokumentationen.

Systemet ska vara ett verktyg för att visualisera olika egenskaper från elektronstrukturberäkningar och själva visualiseringen ska ske i programvaran Inviwo [4]. Systemet ska vara uppbyggt av två huvudsakliga delsystem: ett som hanterar konvertering av data och ett annat som visualiseras den konverterade datan, se figur 1. Vilken data som visualiseras ska kunna väljas av användaren. Systemet ska vara en vidareutveckling av det system som togs fram av 2017 års projektgrupp. Projektgruppen ska utöka och uppdatera befintlig kod för att uppfylla de krav som ställts i kravspecifikationen.



Figur 1: Grov design av systemet

3 Kunskapsbas

Projektgruppen har efter att ha tagit del av projektdirektivet, se appendix G, skrivit en kravspecifikation, se appendix F. Därefter skrevs en projektplan, se appendix H, tidplan, se appendix J och en systemskiss, se appendix I, enligt LIPS-modellen [6]. Dessa dokument har sedan fungerat som en utgångspunkt för projektets utformning. Utöver dessa har en designspecifikation skrivits, se appendix B, som är en fördjupning i systemskissen och ger en mer detaljerad beskrivning av systemet.

Projektgruppen har skrivit två fördjupningsarbeten, se appendix C och D, knutna till projektarbetet.

Projektgruppen har tagit del av ett antal föreläsningar om dokumentationen i projektet. Utöver dessa har även föreläsningar i Inviwo och Python hållts samt laborationer i beräkningsfysik och Python-programmering.

4 Fasplan

Projektet drivs enligt LIPS-modellen, som är en modell med regler, instruktioner och mallar för att bedriva projekt. Projektet delas upp i tre faser - före, under, och efter.

4.1 Före-fas

Före-fasen är ämnad att undersöka om projektet bör genomföras och, om så är fallet, definiera och konkretisera projektets mål samt organisera arbetet som krävs för att uppnå dessa. Under före-fasen skrivs bland annat ett projektdirektiv, se appendix G, en kravspecifikation, se appendix F, en systemskiss, se appendix I, och en projektplan, se appendix H. Projektdirektivet skrevs innan projektgruppen skapades och gavs till projektgruppen vid projektets start. Projektgruppen har under före-fasen skrivit ett gruppkontrakt, se appendix E, en kravspecifikation, en systemskiss samt en projektplan.

4.2 Under-fas

Under under-fasen utförs det arbete som leder till att projektets krav uppfylls. Projektgruppen skriver en designspecifikation, se appendix B, som beskriver hur systemet ska konstrueras mer detaljerat än systemskissen. Projektgruppen arbetar sedan med att, utgående från denna designspecifikation, konstruera och testa systemet. En teknisk dokumentation, se appendix K, skrivs parallellt med systemets konstruktion för att reflektera den faktiska implementationen av idéerna beskrivna i designspecifikationen.

Projektgruppen har också skrivit en naturvetenskaplig undersökning och två fördjupningsarbeten knutna till projektet. Det ena fördjupningsarbetet, se appendix C, behandlar Fermi-ytor som är ett av fördjupningsområdena i projektet och som ska visualiseras. Det andra fördjupningsarbetet, se appendix D, behandlar visualisering som är en mer generell del av projektet. Den naturvetenskapliga undersökningen är en analys av visualiseringar skapade med ENVISIoN, se appendix L.

4.3 Efter-fas

Under efter-fasen överförs projektresultatet till beställaren och projektet avslutas. Projektgruppen skriver en slutrapport som bifogas vid slutleveransen av produkten och även en användarmanual, se appendix A. Vid slutleveransen ska även en demonstration hållas som visar att produkten uppfyller kraven ställda på den. Efter produkten levereras skriver projektgruppen en efterstudie.

5 Fördjupningsarbeten

Nedan beskrivs kortfattat de fördjupningsarbeten som gjorts under projektet. Dessa finns i sin helhet i appendix C och D.

5.1 Fermi-ytor

Syftet med fördjupningsarbetet om Fermi-ytor var att besvara följande frågor: Vad är Fermi-ytor och hur kan dessa beräknas numeriskt? Detta var av intresse för projektet eftersom en del av det bestod av att visualisera Fermi-ytor. Arbetet beskriver kortfattat begrepp inom fasta tillståndets fysik som är av betydelse för att förstå vad Fermi-ytor är, för att sedan gå in på hur dessa kan beräknas numeriskt.

5.2 Visualisering av volymdata

I fördjupningsarbetet om visualisering av volymdata tittades det närmare på ett par av de metoder mjukvaran Inviwo använder sig av. Detta för att kunna göra en jämförelse av både prestanda och resultat mellan de två metoderna.

6 Teknisk beskrivning

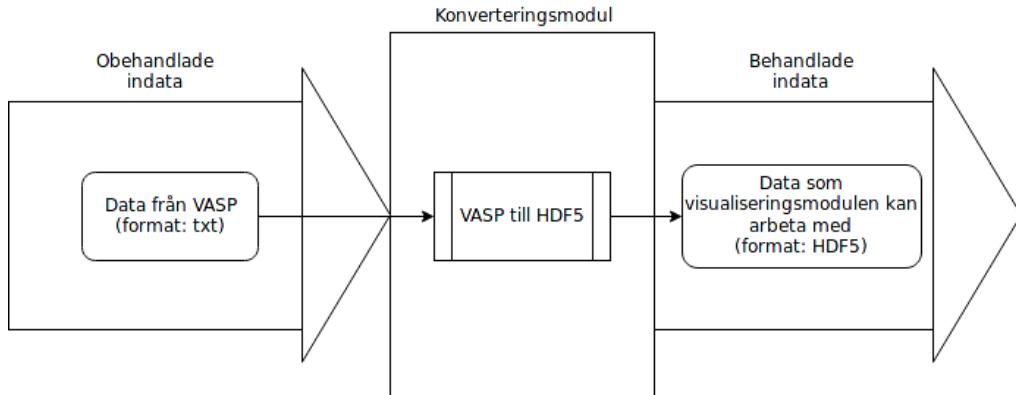
Systemet hanterar indata från VASP. Daten kommer från elektronstrukturberäkningar och konverteras till HDF5-format för att därefter visualiseras. Användaren kan välja vad som ska visualiseras, kontrollera renderingens utseende och ställa in egenskaper såsom färg, transparens, och rotation. Systemet har delats in i delsystem och dessa har utvecklats och testats enskilt i den mån det inte funnits beroenden till andra delsystem.

I avsnitt 6.1 och 6.2 nedan ges en övergripande inblick i det två huvudsakliga delsystemen och deras delsystem. För en mer detaljerad teknisk beskrivning av systemet se den tekniska dokumentationen, se appendix K.

6.1 Datakonvertering

Detta avsnitt behandlar delsystemet datakonvertering, figur 2 ger en grov översikt av delsystemet. Obehandlad data skickas in där obehandlad data är i form av textfiler med data från

beräkningar gjorda i VASP. Indata behandlas sedan i konverteringsmodulen. Från konverteringsmodulen kommer behandlad data i HDF5-format.



Figur 2: Grov design av konverteringsmodulen

6.1.1 HDF5

Systemet kan hantera olika utdatafiler från VASP. Detta görs genom att datan konverteras till HDF5-format. HDF5 är ett filformat som är designat för att hantera stora mängder data på ett flexibelt sätt [2]. HDF5 har flera olika datatyper och ett HDF5-objekt som antingen är lagrat på disk eller hålls i minnet är uppdelat i två huvudsakliga underobjekt, nämligen grupper och dataset.

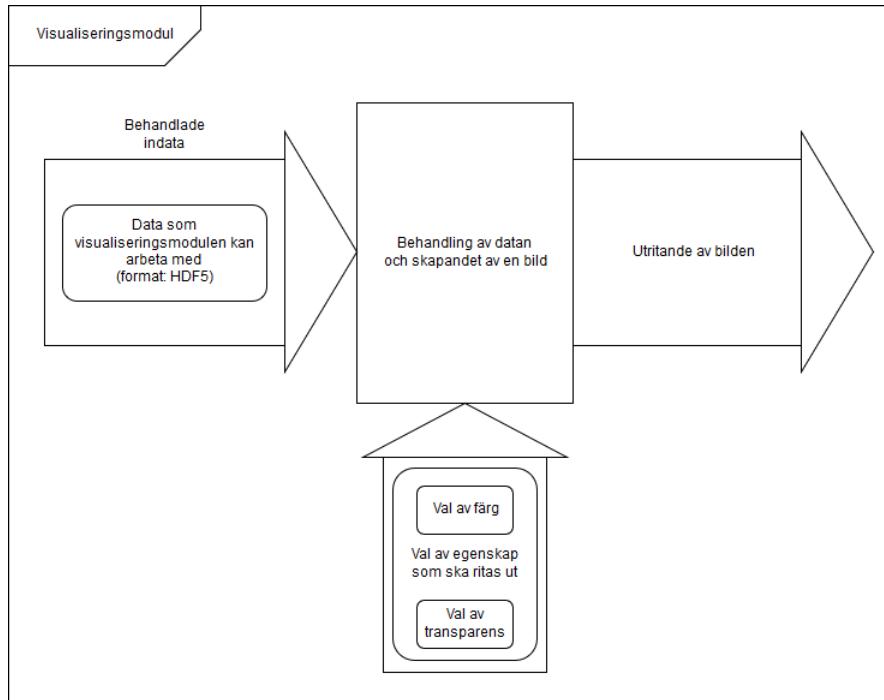
Alla HDF5-objekt har en rotgrupp som äger alla andra objekt i datastrukturen. Denna grupp innehåller i sin tur all övrig data i form av andra grupper, länkar till andra grupper, eller dataset. Dataset innehåller rådata av något slag. Rådata kan i sammanhanget vara bilder, utdata från beräkningar, programdata, etc. I designspecifikationen, se appendix B, ges en mer detaljerad beskrivning av HDF5.

6.1.2 VASP

Från beräkningsprogrammet VASP får en rad olika utdatafiler. Dessa listas och beskrivs i designspecifikationen, se appendix B. Ur dessa utdatafiler läses saker som atompositionsdata, gittervektorer, elektrontäthetsdata, tillståndstäthetsdata, Fermi-energi, och så vidare för att kunna visualisera kristallstrukturer, elektrontäthet, elektronlokaliseringsfunktionen (ELF), tillståndstäthet, och Fermi-ytor. I designspecifikationen beskrivs närmare vad som läses från vilka utdatafiler.

6.2 Visualisering

Figur 3 visar hur delsystemet för visualisering är uppbyggt. Behandlad data samt användarindata skickas in. Användarindata består av val av färg, val av egenskap som ska ritas ut, val av transparens etc. Dessa behandlas i visualiseringssmodulen som skapar en bild utifrån dem. Den utritade bilden skickas sedan ut från modulen. Detta är en allmän beskrivning av visualiseringen - mittenrektangeln i figur 3 är för de olika egenskaperna som visualiseras uppbyggd på olika sätt med olika processorer.



Figur 3: Grov design av delsystemet för visualisering

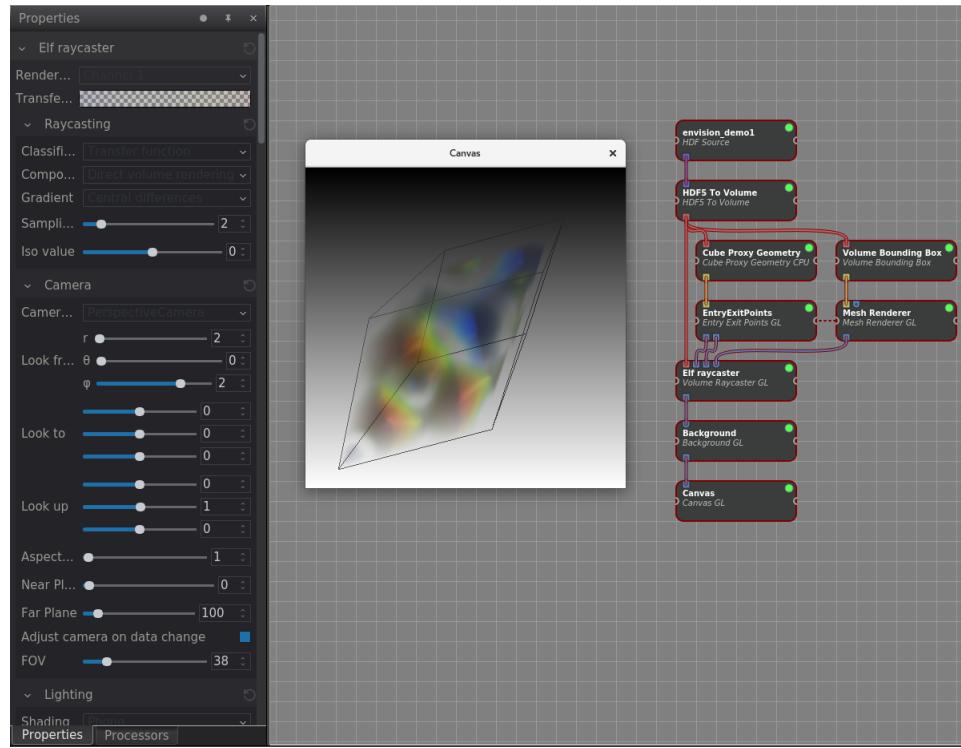
6.2.1 Utritning

Delsystemet ritar ut bilder utefter den behandlade datan. Utritningen ger en visualisering av den egenskap som modulen behandlar och kan till exempel vara en volymrendering eller en 2D-graf, beroende på vilket som är mest lättolkat.

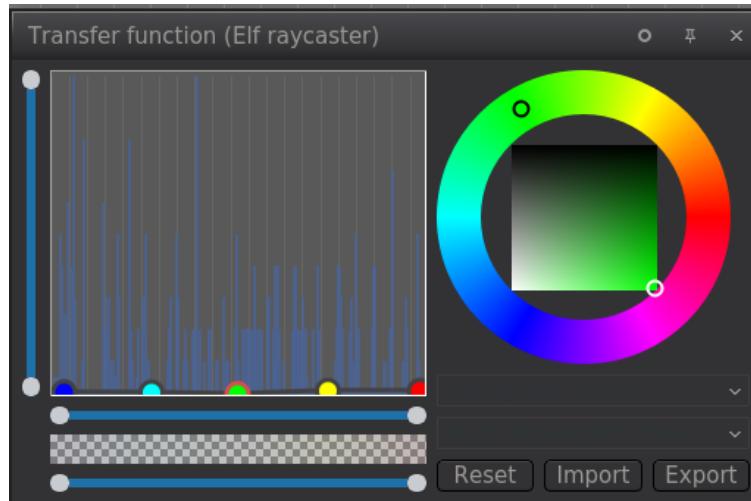
Utritningen görs via Inviwos inbyggda funktionalitet för att rendera via OpenGL som är ett API för att hantera grafik.

6.2.2 Användaridata

Användaren kan ändra inställningar som kontrollerar utseendet på visualiseringarna. Denna indata matas in i Inviwos användargränssnitt via inställningsrutan för en processor. Figur 4 visar vad användaren ser när ELF-data för diamant visualiseras. Längst till vänster i figuren ses ett fönster med Properties, i detta fall för processorn ELF raycasting som kan hittas i nätverket längst till höger i figuren. I mitten av figuren ligger den utritade bilden. Bilden kan roteras genom att klicka och dra i den. I fönstret Properties finns en egenskap vid namn Transfer function, med hjälp av vilken användaren kan ändra transparens/opacitet samt färg, se figur 5.



Figur 4: Bild av vad användaren ser när ELF visualiseras, i detta fall för diamant.



Figur 5: Transfer function property där transparens och färg kan ställas in.

Visualiseringsfunktionaliteten läggs till i form av processorer som via Inviwo tillhandahåller inställningsmöjligheter på det sätt som beskrivits i designspecifikationen, se appendix B. Således blir inte användarinställningar ett eget separat delsystem, utan kommer att bokas in i visualiseringssystemen. Detta görs genom att processorerna är skrivna att exponera de inställningar som användaren är menad att justera. Inviwo kommer då automatiskt lägga till inställningarna i rutan.

6.2.3 Interaktivitet

Användaren kan modifiera visualiseringen genom att reglera ett intervall av värden för någon egenskap där full transparens fås för alla värden inom intervallet, rotera 3D-bilder etc. Ny användaridata, som skickas in efter att den första bilden har ritats upp, skickas tillbaka till visualiseringssmodulen för att utföra en ny rendering.

6.2.4 Kristallstruktur

Kristallstruktur visualiseras som atompositioner i enhetscellen. Sfärer som representerar atomer kan ritas ut genom volymsrendering. I designspecifikationen, se appendix B, finns en mer detaljerad beskrivning av hur detta byggs upp. I kap. 7 i figur 9 ses ett exempel på denna typ av visualisering.

6.2.5 Elektrontäthet

Elektrontäthet visualiseras genom volymsrendering. Det är sannolikheten för att hitta elektroner på olika positioner som visualiseras. I designspecifikationen, se appendix B, finns en mer detaljerad beskrivning av elektrontäthet. I kap. 7 i figur 10 och figur 11 ses två exempel på denna typ av visualisering.

6.2.6 ELF

Data för elektronlokaliseringsfunktionen (electron localization function, ELF) visualiseras på liknande sätt som elektrontäthet eftersom även det är volymdata som ska visualiseras. Hur ELF visualiseras beskrivs mer detaljerat i den tekniska dokumentationen, se appendix K. I kap. 7 i figur 7 ses ett exempel på denna typ av visualisering.

6.2.7 Tillståndstäthet

Tillståndstäthet visualiseras med en 2D-graf med energin på x-axeln och tillståndstätheten på y-axeln. I designspecifikationen, se appendix B, finns en mer detaljerad beskrivning av tillståndstäthet. I kap. 7 i figur 6 ses ett exempel på denna typ av visualisering.

6.3 Sammanlänkning av olika visualiseringar

Det finns även möjlighet att visualisera kristallstrukturer tillsammans med elektrontäthet, ELF eller tillståndstäthet. I kap. 7 i figur 8 ses ett exempel på denna typ av visualisering.

7 Resultat

Projektet bestod av att skapa ett visualiseringssverktyg i Inviwo som ska kunna visualisera resultat från elektronstrukturberäkningar. Det framtagna visualiseringssverktyget ska uppfylla kraven från kravspecifikationen, nedan beskrivs några av kraven som ställts på systemet samt huruvida dessa är uppfyllda.

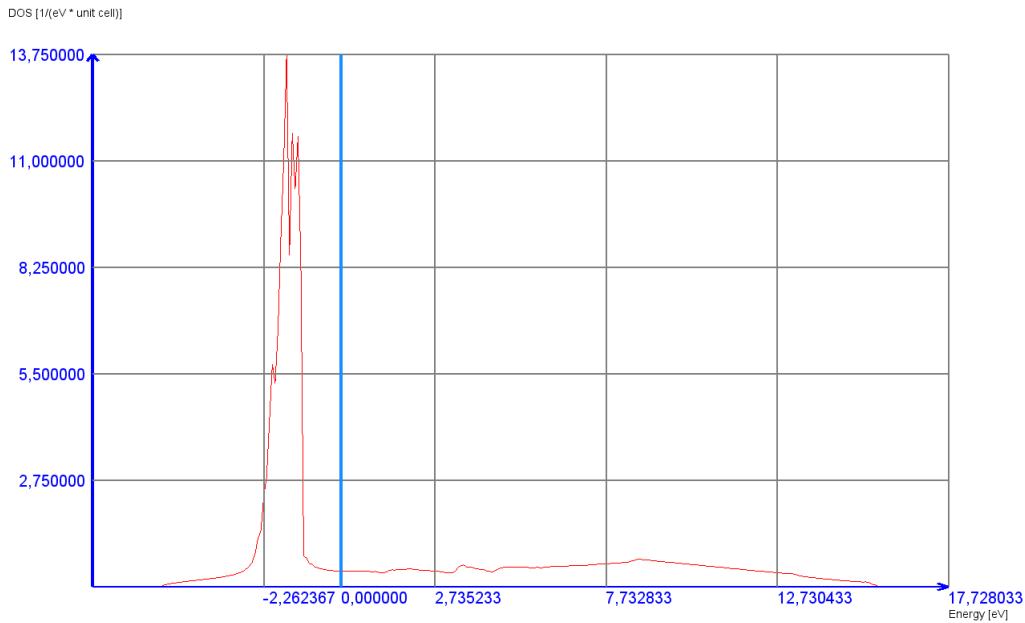
7.1 Krav på systemet

Tabell 1 är en kravlista med några utvalda krav på det generella systemet. För samtliga krav se kravspecifikationen i appendix F. Krav fem är ett krav på egenskaper som ska kunna visualiseras. Projektgruppen valde att implementera visualisering av Fermi-ytor och Total DOS, men även visualisering av ELF har implementerats. 2017 års projektgrupp i visualisering implementerade ur listan i krav fem visualisering av total DOS, visualisering av ELF-data, och visualisering av bandstruktur. Detta års arbete bestod i att uppdatera majoriteten av den skrivna koden till att vara kompatibel med version 0.9.9 av Inviwo. Krav 45 är ett ska-krav som säger att koden ska fungera för version 0.9.9 eller senare versioner av Inviwo. Att uppdatera befintlig kod visade sig bli en stor del av arbetet eftersom tidigare versioner av Inviwo skilde sig mycket från version 0.9.9 då Inviwo uppdaterat sitt python stöd i grunden.

Tabell 1: Reducerad kravlista för det generella systemet.

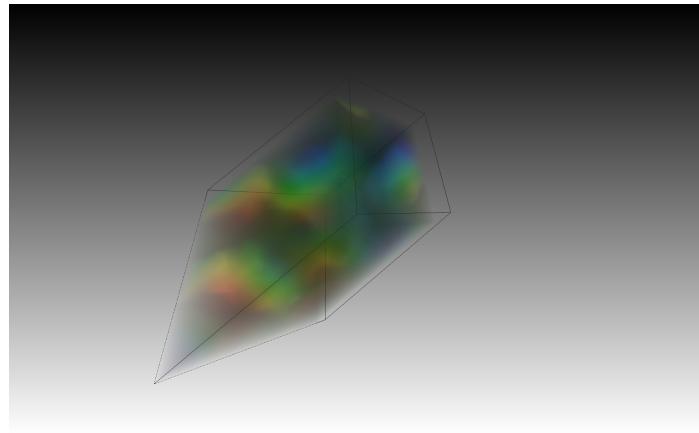
Krav	Förändring	Beskrivning	Prioritet	Uppfyllt
5	Omförhandlat 2018-05-21	Projektgruppen ska utöka alternativt implementera visualisering av minst en av nedanstående egenskaper, samt uppdatera minst en redan existerande implementation av någon av de nedanstående egenskaperna: <ul style="list-style-type: none">• Elastiska konstanter• Fermi-ytor• ELF (Electron Localization Function)• Krafter på atomer• Bandstruktur• Total DOS (Density Of States)• Parkorrelationsfunktionen• Illustration av partiell elektrondensitet	Ska	Ja
7	Original	Tillhandahållna python-moduler ska kunna anropas med enkla funktionsanrop.	Ska	Ja
13	Original	Användaren ska kunna välja vilken typ av visualisering som ska visas.	Ska	Ja
14	Original	Användaren ska kunna länka samman olika python-moduler.	Ska	Ja
15	Original	Systemet ska implementeras i Inviwo.	Ska	Ja
18	Original	Tillhandahållna python-moduler ska inte fortskrida som vanligt vid indata av fel typ.	Ska	Ja
19	Original	Tillhandahållna python-moduler ska vid felaktig indata varna användaren.	Ska	Ja
45	Original	Koden ska fungera för Inviwo 0.9.9 eller någon efterkommande version.	Ska	Ja

I krav 5 valde projektgruppen att utöka implementeringen för Total DOS som kan ses i figur 6 som visar en 2D-graf som ritar upp total tillståndstäthet för koppar, Cu.



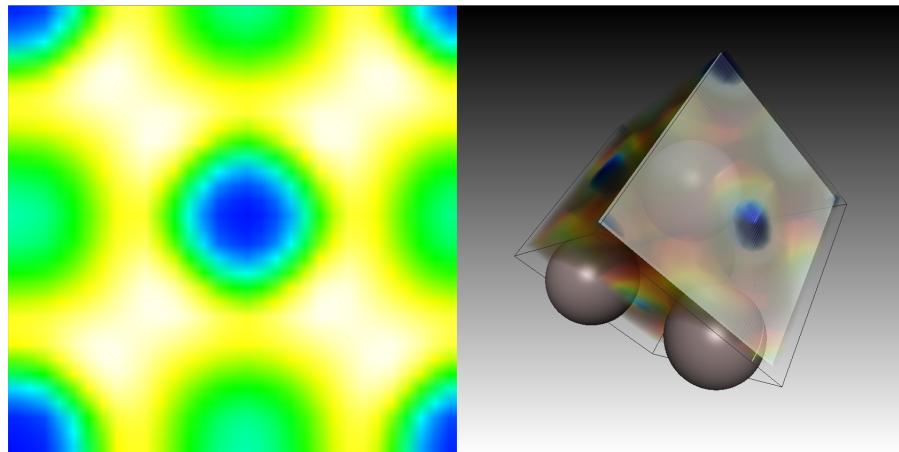
Figur 6: Visualisering av total tillståndstäthet för Cu.

Även befintlig kod för visualisering av ELF har uppdaterats. Figur 7 visar visualisering av ELF-data för diamant.



Figur 7: Visualisering av ELF-data för diamant.

Figur 8 visar hur visualiseringen för en sammanlänkning mellan utritande av enhetscell och utritande av ELF kan se ut, i detta fall för aluminium. Detta uppfyller alltså krav 14. Sammanlänkning kan göras mellan kristallstruktur och volymdata (laddningstäthet och ELF) eller mellan kristallstruktur och tillståndstäthet. Det senare gör det möjligt att se projicerad tillståndstäthet genom att klicka på enskilda atomer, vilket är krav 36 i tabell 3.



Figur 8: Data för enhetscell och ELF för aluminium visualiseras i samma bild.

I tabell 2 ses några av de krav som ställts på delsystemet för datakonvertering. Krav 24 och 29 är krav på vilket beräkningsprogram som systemet ska kunna läsa in resultat ifrån. Krav 25, 26 och 27 är krav på vilka data som ska kunna läsas in och krav 28 är ett krav på till vilket filformat som datan ska konverteras.

Tabell 2: Reducerad kravlista för delsystemet för datakonvertering.

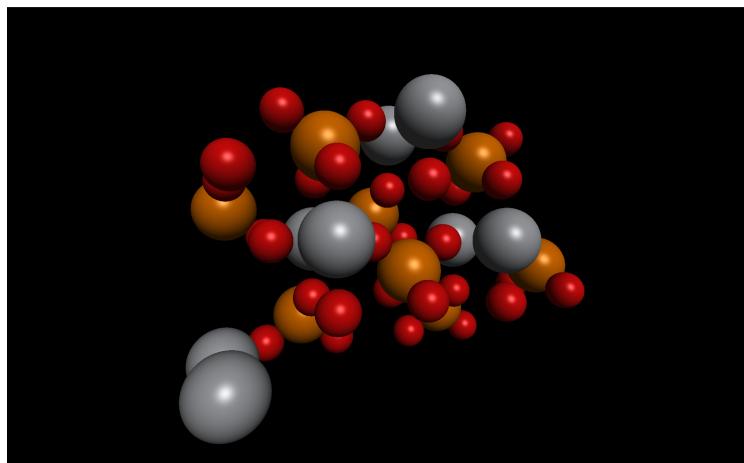
Krav	Förändring	Beskrivning	Prioritet	Uppfyllt
24	Original	Systemet ska kunna läsa in resultat från VASP.	Ska	Ja
25	Original	Systemet ska kunna konvertera data från kristallstrukturberäkningar.	Ska	Ja
26	Original	Systemet ska kunna konvertera data från elektronstrukturberäkningar.	Ska	Ja
27	Original	Systemet ska kunna konvertera data från tillståndstäthetsberäkningar.	Ska	Ja
28	Original	Systemet ska översätta input-filer i text-format till det binära filformatet HDF5.	Ska	Ja
29	Original	Systemet bör kunna läsa in resultat från något annat beräkningsprogram, t.ex. Elk	Bör	Nej

I tabell 3 ses några av de krav som ställts på delsystemet för visualisering. Visualisering av projektionerad tillståndstäthet, kristallstruktur, och elektrontäthet implementerades av 2017 års projektgrupp, och det relaterade arbetet detta år bestod i att uppdatera koden för dessa visualiseringar till att vara kompatibla för version 0.9.9 av Inviwo.

Tabell 3: Reducerad kravlista för delsystemet för visualisering.

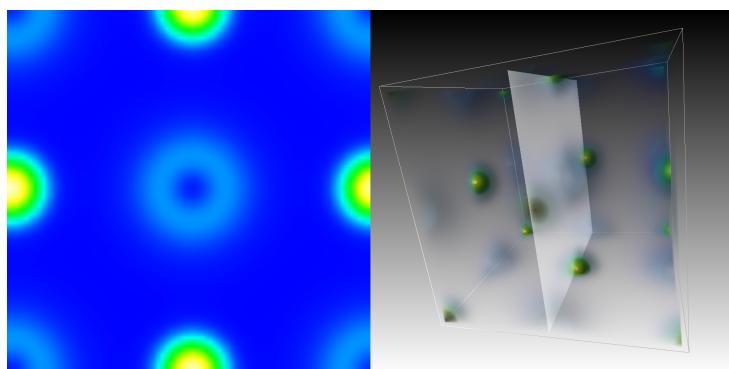
Krav	Förändring	Beskrivning	Prioritet	Uppfyllt
36	Original	Systemet ska visualisera projicerad tillståndstäthet härrörande till varje separat atom i en kristalls enhetscell.	Ska	Ja
37	Original	Systemet ska visualisera kristallstruktur som atompositioner i enhetscellen.	Ska	Ja
38	Original	Systemet ska kunna visualisera den elektrontäthet som resulterar från en beräkning.	Ska	Ja

Figur 9 är ett exempel på visualisering av kristallstruktur, krav 37. Här är det TiPO4 som ritas ut där de röda sfärerna är syreatomer, de grå är titanatomer, och de orangea är fosforatomer.

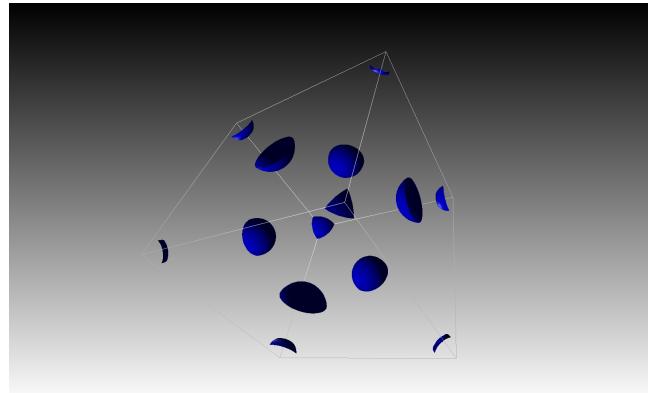


Figur 9: Visualisering av TiPO4.

Figur 10 är ett exempel på visualisering av elektrontäthet, krav 38. Här är det laddningstätheten för NaCl som ritas ut. I figur 11 visas även ett exempel på visualisering av laddningstäthet men i detta fall har en isoyta ritats ut för laddningstäthet lika med 0.15, även här för NaCl.



Figur 10: Visualisering av laddningstäthet för NaCl med slice-funktion.



Figur 11: Visualisering av isoyta för laddningstäthet lika med 0.15 för NaCl.

7.2 Användning

För närmare beskrivning av hur verktyget används se användarmanualen i appendix A. Det framtagna visualiseringssverktyget används genom att öppna programmet Inviwo och därifrån köra önskat Python-skript. Python-skript finns för samtliga typer av visualiseringar och kräver endast små ändringar i koden för att köras. Vilka ändringar som behöver göras beskrivs dels i användarmanualen men även som kommentarer i skripten.

8 Slutsats

Projekterbetet har medfört ett antal lärdomar och insikter. I början av projektet skulle flera dokument skrivas som skulle vara till grund för designen av systemet som skulle implementeras samt genomförandet av projektet. Detta upplägg medförde vissa problem eftersom det krävde mycket kunskap inom främst Inviwo. Nedan beskrivs några slutsaster kring projektet inom några olika områden.

8.1 Tekniskt

Versionerna av Inviwo skilde sig mer än förväntat vilket gjorde att mer tid än beräknat har gått till att uppdatera kod samt lösa problem med Inviwo. Detta har gjort att den tekniska biten har varit väldigt tung. Projektgruppen känner att det gärna hade fått vara mer fokus på visualiseringssproblem och beräkningsfysik än på tekniska problem. Planeringen av projektet utgick snarare från att eventuella problem som skulle uppstå skulle handla om förståelse av de egenskaper som skulle visualiseras. Problemen som uppstod med Inviwo krävde mycket speciellkunskap för att lösas vilket gjorde att en i gruppen fick lägga mycket tid på sådant, vilket syns i tidsfördelning av nedlagd tid.

Programmeringen har varit väldigt lärorik för flera i gruppen. Att uppdatera 2017 års kod har, trots att det svalde mycket tid, varit nyttigt och gett förståelse för visualiseringen. Det har även varit lärorikt att få arbeta i ett lite större projekt, att få planera arbetet och att lösa problem som dykt upp längs vägen.

Under projektets gång har versionshanteringssystemet Git använts för att versionshantera kod och typsättningssystemet LaTeX använts för att skriva dokument, detta har varit väldigt värdefullt att sätta sig in i och använda eftersom det med stor sannolikhet är till nytta även utanför projektet.

8.2 Utförande

I början av projektet bestod arbetet av att skriva de olika dokument som projektarbetet sedan skulle utgå ifrån. Detta var svårt eftersom det saknades mycket kunskap inom Inviwo som var en central del i projektet. Projektgruppen känner sig nöjda med de producerade dokumenten även om t.ex. krav i kravspecifikationen i efterhand kunde ha formulerats annorlunda och varit mer tydliga, vilket var svårt att inse i början av projektet. Det hade varit bra med mer tid i början av projektet, innan skrivning av dokument påbörjades, där en slags förstudie hade kunnat göras för att sätta sig in i projektet. Detta hade kunnat ge en bättre bild av vad som behövde göras så att kravspecifikation och designspecifikation kunde skrivas mer detaljerat och korrekt.

En i gruppen var mer kunnig i programmering än de andra. Detta var mycket värdefullt för gruppen eftersom problem som hade varit mycket komplicerade att lösa för övriga projektmedlemmar ändå kunde lösas. Dock medförde det också att tidsfördelningen av nedlagd arbetstid inte blev helt jämn inom gruppen.

8.3 Framtida arbete

En idé hade varit att göra om saker från grunden istället för att uppdatera 2017 års kod. Det är dock oklart om detta verkligen hade gjort att saker gått snabbare eftersom problem med Inviwo ändå hade uppstått, men dessa hade möjligtvis kunnat kringgås på andra sätt. Arbetet hade möjligtvis kunnat paralleliseras mer men detta hade krävt mer förståelse av Inviwo från början och en bättre insikt i hur stora ändringar som behövde göras för att göra koden kompatibel med version 0.9.9 av Inviwo.

Arbetet har känts effektivt, de nedlagda timmarna har använts bra.

Referenser

- [1] C++. URL: <http://www.cplusplus.com/info/description/> (hämtad 2018-02-23).
- [2] The HDF Group. *Hierarchical Data Format, version 5*. 1997-2018. URL: <https://support.hdfgroup.org/HDF5/> (hämtad 2018-02-21).
- [3] Nationalencyklopedin. *Fermi-tyta*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/fermi-tyta> (hämtad 2018-02-23).
- [4] Overview - Inviwo. URL: <http://www.inviwo.org/overview/> (hämtad 2018-02-23).
- [5] Python. URL: <https://www.python.org> (hämtad 2018-02-23).
- [6] Svensson Tomas och Krysander Christian. *LIPS*. 1:1. Lund: Studentlitteratur, 2011.

A Användarmanual

User guide

Editor: Viktor Bernholtz

Version 0.2

Status

Granskad	DOK	2018-05-25
Godkänd		

PROJECT IDENTITY

2018/Spring, Group 2
Linköpings Tekniska Högskola, IFM

Group members

Name	Role	Phone no.	E-mail address
Anders Rehult	Project leader (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Document manager (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Secretary (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Client: IFM, Linköpings universitet, 581 83 Linköping

Contact person of client: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Course examiner: Per Sandström, 013-282902, persa@ifm.liu.se

Main supervisor: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Contents

Document history	iv
1 Introduction	1
2 How to build and run ENVISIoN	1
2.1 Install git	1
2.2 Download ENVISIoN	1
2.3 Prepare Inviwo using the ENVISIoN install script	2
3 Start Inviwo and run ENVISIoN scripts	2
4 Scripts	3
4.1 Unit cell	3
4.2 ELF	5
4.3 Charge density	6
4.4 DOS	8
4.5 Interconnected networks	10
4.5.1 Unit cell and charge	10
4.5.2 Unit cell and ELF	10
4.5.3 Unit cell and DOS	11

Document history

Version	Date	Changes	Done by	Reviewed
0.1	2018-05-22	First draft.	Project group	PL
0.1	2018-05-25	Second draft.	Project group	DOK

1 Introduction

ENVISIoN is an open source toolkit for electron visualisation.

ENVISIoN is implemented by using a modified version of the Inviwo visualisation framework, developed at the Scientific Visualization Group at Linköpings universitet, LIU.

ENVISIoN has been developed as a part of the course TFYA75: Applied Physics - Bachelor's Project, given at Linköpings universitet, LiU.

The present version was developed during the spring term of 2018 by a project group consisting of: Anders Rehult, Marian Brännvall, Andreas Kempe and Viktor Bernholtz. Supervisor: Johan Jönsson; Requisitioner and co-supervisor: Rickard Armiento; Visualization expert: Rickard Englund; and Course examiner: Per Sandström. The work is based on a previous version by the project group taking the course in the spring term of 2017 consisting of: Josef Adamsson, Robert Cranston, David Hartman, Denise Härnström, Fredrik Segerhammar. Supervisor: Johan Jönsson; Requisitioner and co-supervisor: Rickard Armiento; Visualization expert: Peter Steneteg; and Course examiner: Per Sandström.

ENVISIoN provides a set of Python scripts that allow the user to:

- Read and parse output from electronic structure calculations made by the program VASP and storing the result in a structured HDF5 file.
- Generate interactive Inviwo visualisation networks for common tasks when analyzing electronic structure calculations. Presently there is support for visualising the crystal structure of the unit cell of a material, electron localization function (ELF)-data, electronic charge density, and density of states - both total and partial. The system also provides the ability to interconnect some of the networks mentioned above.

2 How to build and run ENVISIoN

These instructions show how to build Inviwo and ENVISIoN on Ubuntu 18.04 LTS.

2.1 Install git

Start by installing git, which will be used to fetch ENVISIoN in the next step.

```
sudo apt install git
```

2.2 Download ENVISIoN

Go to your home folder and clone ENVISIoN from Github. This guide will assume that both ENVISIoN and Inviwo will be placed directly under the home folder.

```
cd  
git clone https://github.com/rartino/ENVISION
```

2.3 Prepare Inviwo using the ENVISIoN install script

ENVISIoN provides an install script for Ubuntu 18.04 LTS. Executing the installation script will install all required dependencies, clone Inviwo from Github and configure the Inviwo build.

The script should *NOT* be run as root, but as your own user and it will ask for your password when it needs root rights. It is possible that the script will ask for other user input during the process, if that's the case, just accept the default.

```
cd ~/ENVISIoN/scripts  
./install_envision_ubuntu_1804.sh /home/$USER/ENVISIoN /home/$USER/inviwo
```

Once the installation script has run, it prints build instructions. Follow the instructions and start the build. The instructions will tell you to *cd* to the build directory and execute *make*.

An easy way to modify the build settings, if needed, is to install the *cmake curses gui* and run it in the build directory.

To install the *cmake gui*:

```
sudo apt install cmake-curses-gui
```

Running *cmake* in the build directory:

```
cd ~/inviwo/build  
ccmake .
```

When in the GUI, press *c* to apply the current configuration, *g* to generate build files and *q* to quit. If settings have changed, it is possible that you will need to press *c* more than once before the *g* option becomes available.

After having generated the build files, the project can now be rebuilt with the new settings by executing *make* like earlier.

3 Start Inviwo and run ENVISIoN scripts

After the Inviwo build is done, the binary will be available as *bin/inviwo* in the build folder. Start it by executing the command below, while still standing in the build directory.

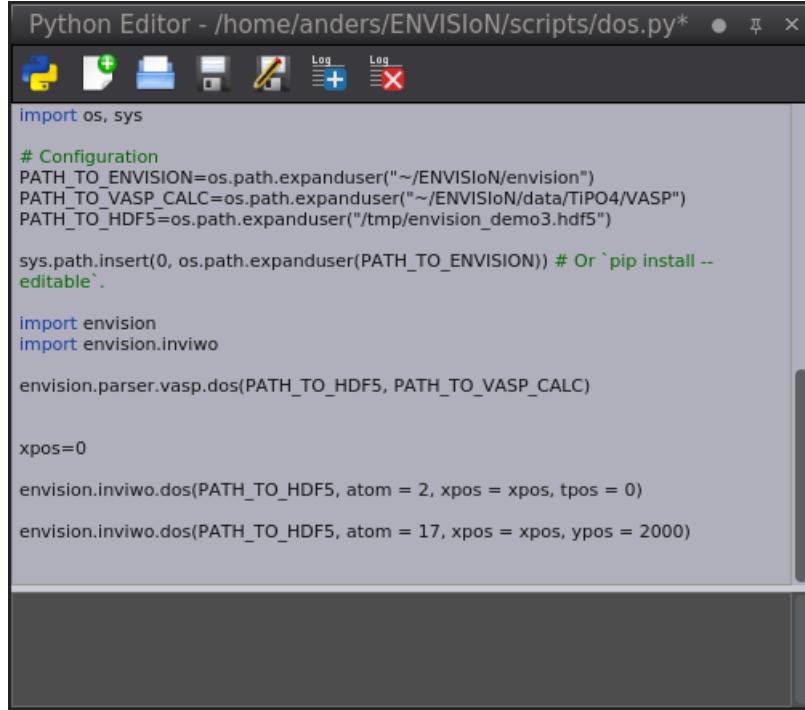
```
cd ~/inviwo/build  
./bin/inviwo
```

- Open Python Editor under Python menu.
- In the Python editor, click Open Script.
- Select one of the scripts.
- Click open.
- Click the python logo in the top left corner to run.

More information about each script and how to use them in chapter 4.

4 Scripts

Descriptions and examples of all the available scripts are found below. The scripts are called through the python editor in Inviwo as shown in figure 1.



```
Python Editor - /home/anders/ENVISIoN/scripts/dos.py* ● □ ×
Python Editor - /home/anders/ENVISIoN/scripts/dos.py* ● □ ×
import os, sys

# Configuration
PATH_TO_ENVISION=os.path.expanduser("~/ENVISIoN/envision")
PATH_TO_VASP_CALC=os.path.expanduser("~/ENVISIoN/data/TiPO4/VASP")
PATH_TO_HDF5=os.path.expanduser("/tmp/envision_demo3.hdf5")

sys.path.insert(0, os.path.expanduser(PATH_TO_ENVISION)) # Or `pip install --editable`.

import envision
import envision.inviwo

envision.parser.vasp.dos(PATH_TO_HDF5, PATH_TO_VASP_CALC)

xpos=0

envision.inviwo.dos(PATH_TO_HDF5, atom = 2, xpos = xpos, tpos = 0)
envision.inviwo.dos(PATH_TO_HDF5, atom = 17, xpos = xpos, ypos = 2000)
```

Figure 1: Python script for calling on the DOS visualisation module of ENVISIoN. Note that the pathway PATH_TO_VASP_CALC here is set to "~/ENVISIoN/data/TiPO4/VASP", the folder containing the relevant VASP output files.

In all of the scripts the user has to specify the path to ENVISIoN, a path where VASP output files are found, and a desired path where ENVISIoN will generate a HDF5-file, or, if one already exists, use that file and skip parsing VASP data. **NOTE:** change the value of PATH_TO_HDF5, delete the HDF5-file, or rename the HDF5-file in between visualisations if a new material is to be visualised.

4.1 Unit cell

In the Python editor of Inviwo the user can choose to generate a visualisation of the unit cell by opening the folder *scripts* and running the file *unitcell.py*.

Figure 2 is an example of a unit cell visualisation, in this case of TiPO4. Figure 3 shows the network responsible for the aforementioned visualisation.

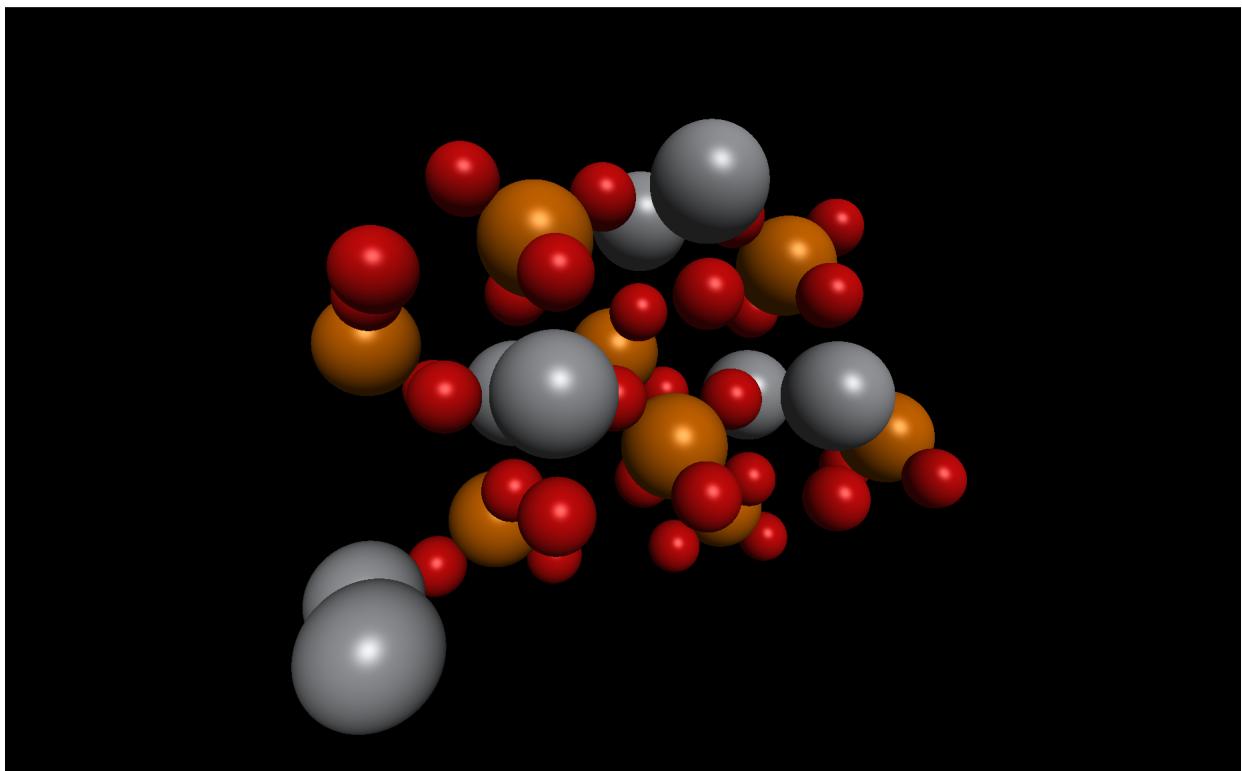


Figure 2: Visualisation of the crystal structure of the unit cell of TiPO4 in Inviwo.

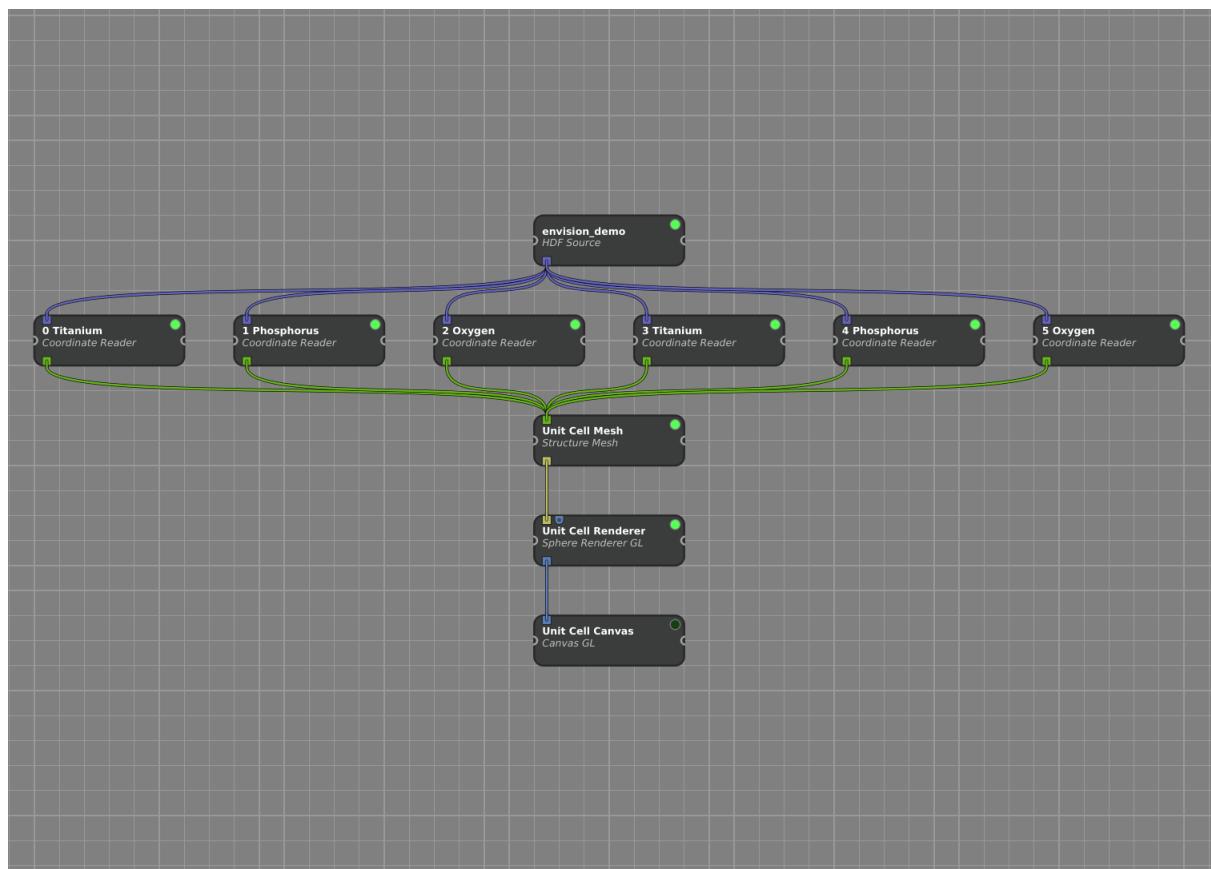


Figure 3: Network for visualisation of TiPO4 in Inviwo.

4.2 ELF

In the Python editor of Inviwo the user can choose to generate a visualisation of the Electron Localisation Function (ELF)-data as shown in figure 4 with corresponding network shown in figure 5 by opening the folder *scripts* and running the file *elf.py*. A second image showing the ELF-data of a slice of the unit cell can be added by changing the argument *Slice* of the function *elf* to True. This slice corresponds to a plane intersecting the unit cell, which can be moved using the W and S keys on the keyboard. The orientation of this plane can be changed by selecting the processor "Volume Slice" in the network by clicking on it, then choosing another value from the drop down menu in the property "Slice along axis".

The ELF-data can also be visualised as an isosurface. This is achieved by changing the argument *iso* of the function *elf* from None to any value between 0 and 1. The slice function described above is not compatible with isosurface visualisation.

The colors and opacities of different values can be changed by selecting the processor "ELF raycaster" in the network, clicking on the property "Transfer function", and interacting with the window that pops up. This window shows multicolored dots which can be manipulated to control opacity. Dots can be added by double clicking, and removed by clicking a dot and pressing the Delete key on the keyboard. Dots can be dragged and dropped using the mouse. The horizontal axis of the plane on which the dots are placed corresponds to ELF values between 0 and 1, while the vertical axis corresponds to opacities between 0 (transparent) and 1 (completely opaque). Figure 9 in chapter 4.3 shows what this window looks like.

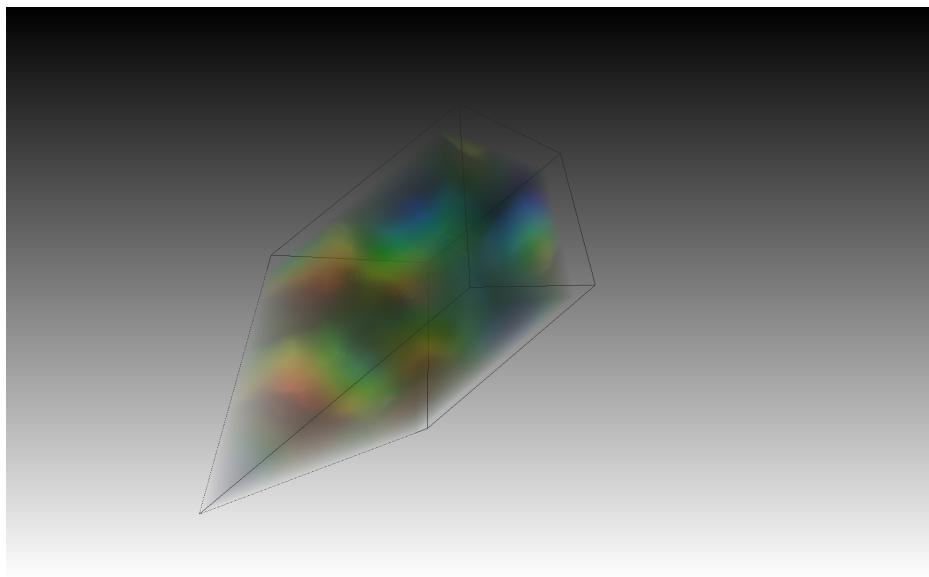


Figure 4: Visualisation of ELF-data for diamond in Inviwo.

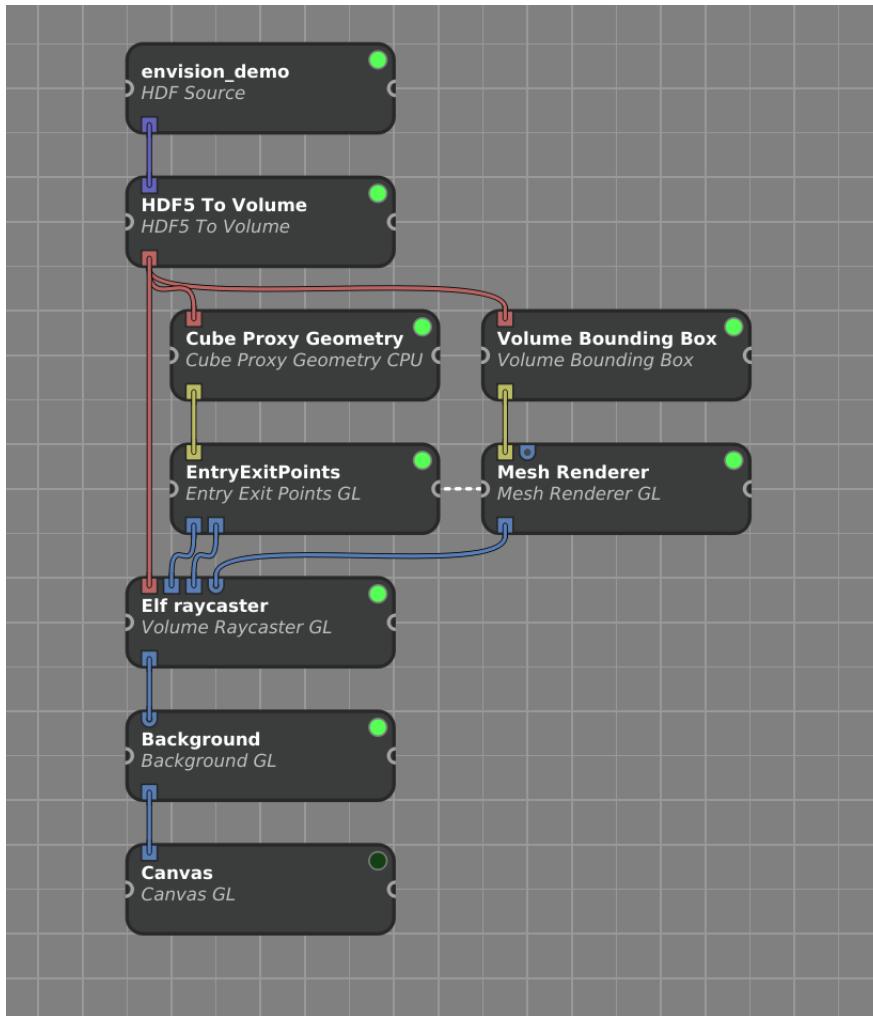


Figure 5: Network for visualisation of ELF-data for diamond in Inviwo.

4.3 Charge density

In the Python editor of Inviwo the user can choose to generate a visualisation of the charge density by opening the folder *scripts* and running the file *charge.py*. A second image showing the charge density of a slice of the unit cell can be added by changing the argument *Slice* of the function *charge* to True. This slice corresponds to a plane intersecting the unit cell, which can be moved using the W and S keys on the keyboard. The orientation of this plane can be changed by selecting the processor "Volume Slice" in the network by clicking on it, then choosing another value from the drop down menu in the property "Slice along axis". This type of visualisation is shown in figure 6, and the generating network is found in figure 7.

The charge density can also be visualised as an isosurface. This is achieved by changing the argument *iso* of the function *charge* from None to any value between 0 and 1. Figure 8 shows an example of this. The slice function described above is not compatible with isosurface visualisation.

The colors and opacities of different values can be changed by selecting the processor "Charge raycaster" in the network, clicking on the property "Transfer function", and interacting with the window that pops up. This window shows multicolored dots which can be manipulated to control opacity. Dots can be added by double clicking, and removed by clicking a dot and

pressing the Delete key on the keyboard. Dots can be dragged and dropped using the mouse. The horizontal axis of the plane on which the dots are placed corresponds to charge density values between 0 and 1, while the vertical axis corresponds to opacities between 0 (transparent) and 1 (completely opaque). Figure 9 shows what this window looks like.

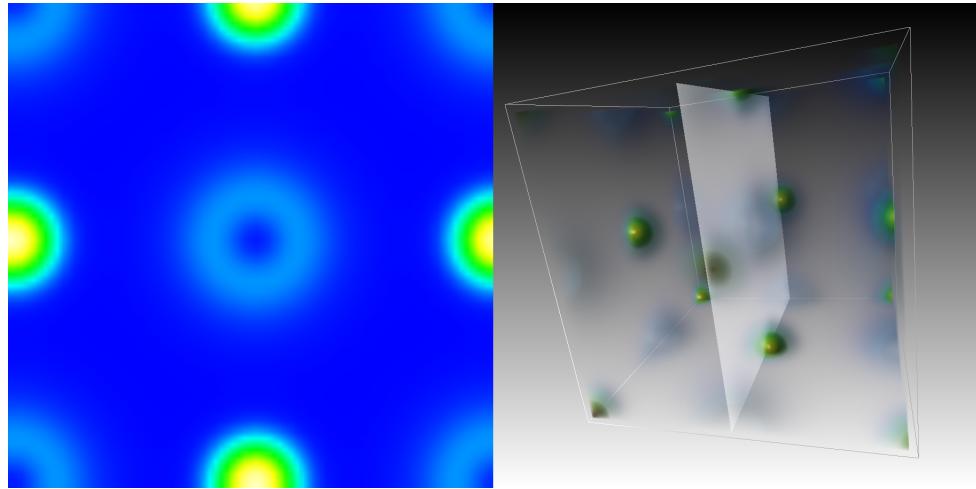


Figure 6: Visualisation of charge density for NaCl with slice function in Inviwo.

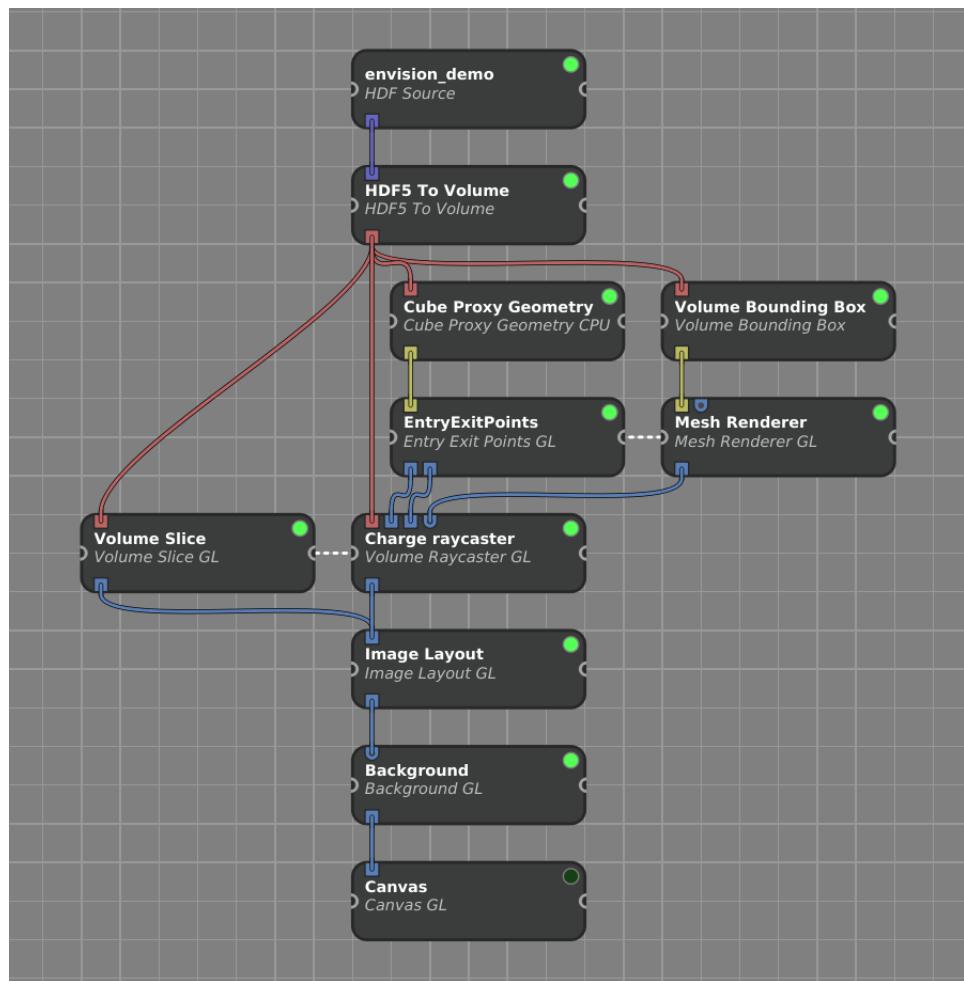


Figure 7: Network for visualisation of charge density for NaCl with slice function in Inviwo.

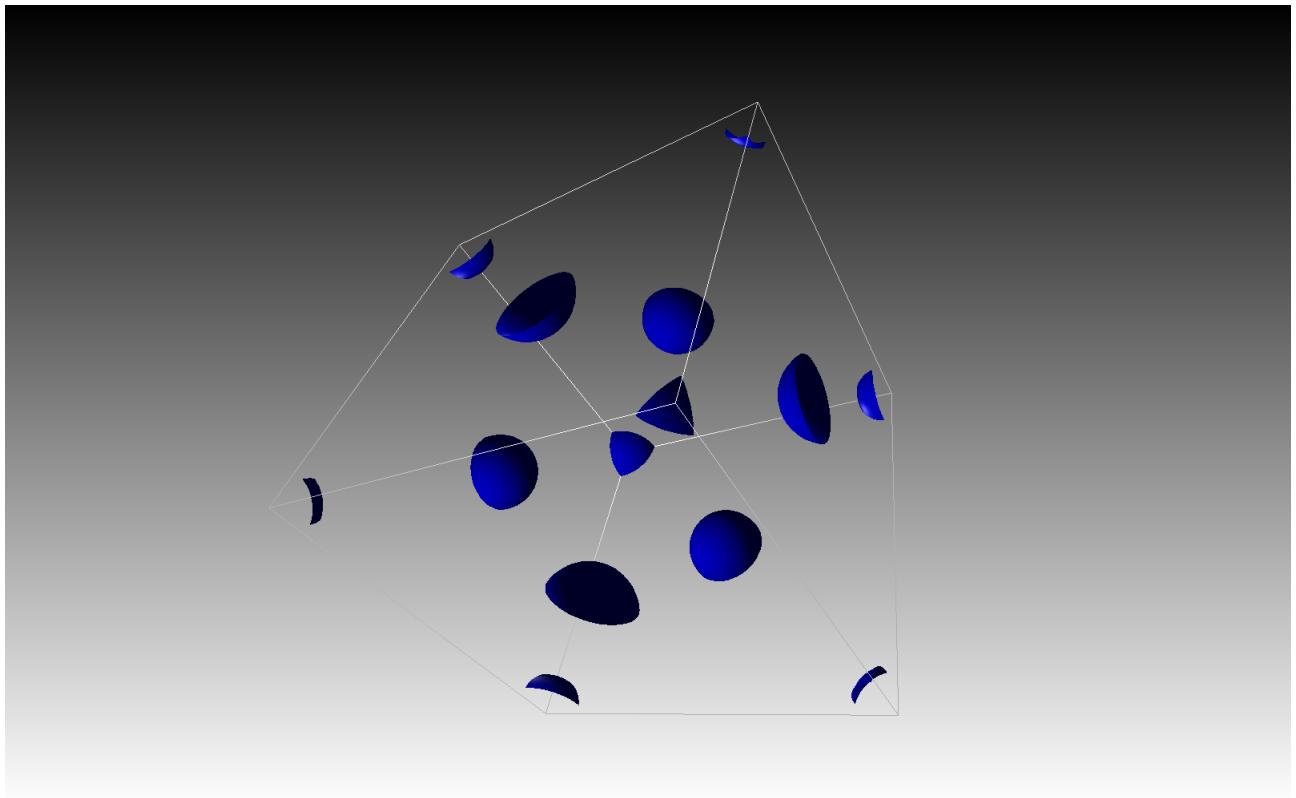


Figure 8: Visualisation of charge density for NaCl with ISO raycasting.

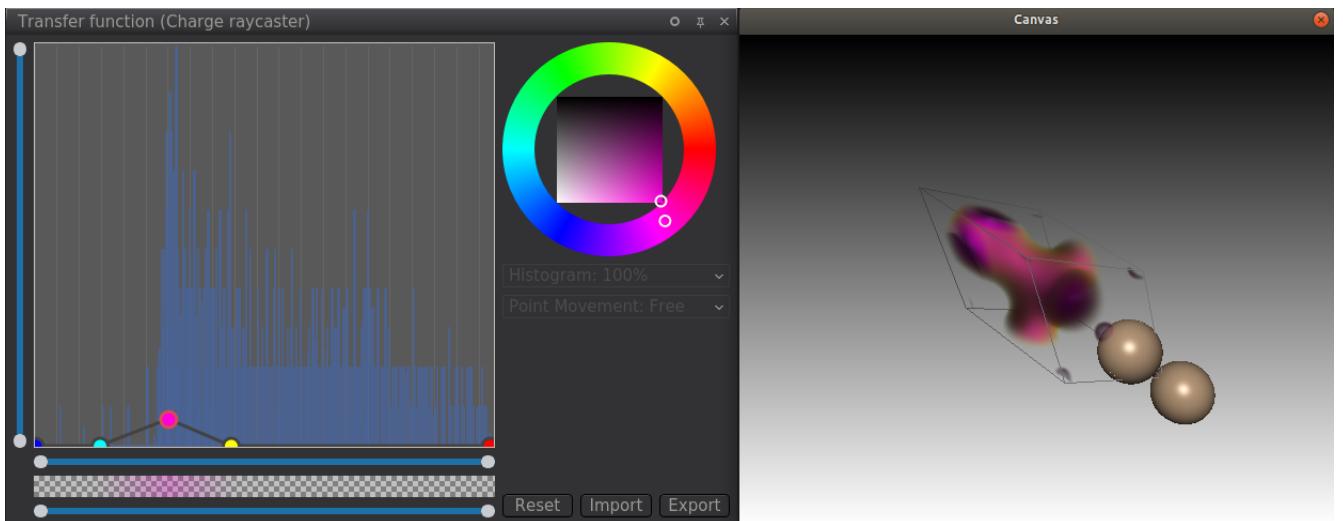


Figure 9: Transfer function of the Charge raycaster processor and visualisation of charge density of Si connected with unit cell visualisation. Charge density values between 0.15 and 0.45 are illustrated as a somewhat transparent purple volume.

4.4 DOS

In the Python editor in Inviwo, the user can choose to generate a visualisation of the density of states by opening the folder *scripts* and running the file *dos.py*.

Figure 10 is an example of a visualisation of the total density of states, in this case for Cu. Figure 11 shows the network for the visualisation. This network generates both a 2D-graph of the total density of states and a similar graph but for the partial density of states. The partial density of states is by default shown for one atom, in two graphs for spin-polarized calculations, determined by the argument *atoms* in the function *dos*. See chapter 4.5.3 for a description of how to pick the atom for which to show projected DOS.

Visualisation of the partial DOS for one specific orbital (s, p, d, or f) of an atom can be achieved by deleting the connections between the undesired processors of type "HDF5 To Function" and the corresponding processor with name "Partial Add". To do this, select a connection by clicking on it and press the Delete key.

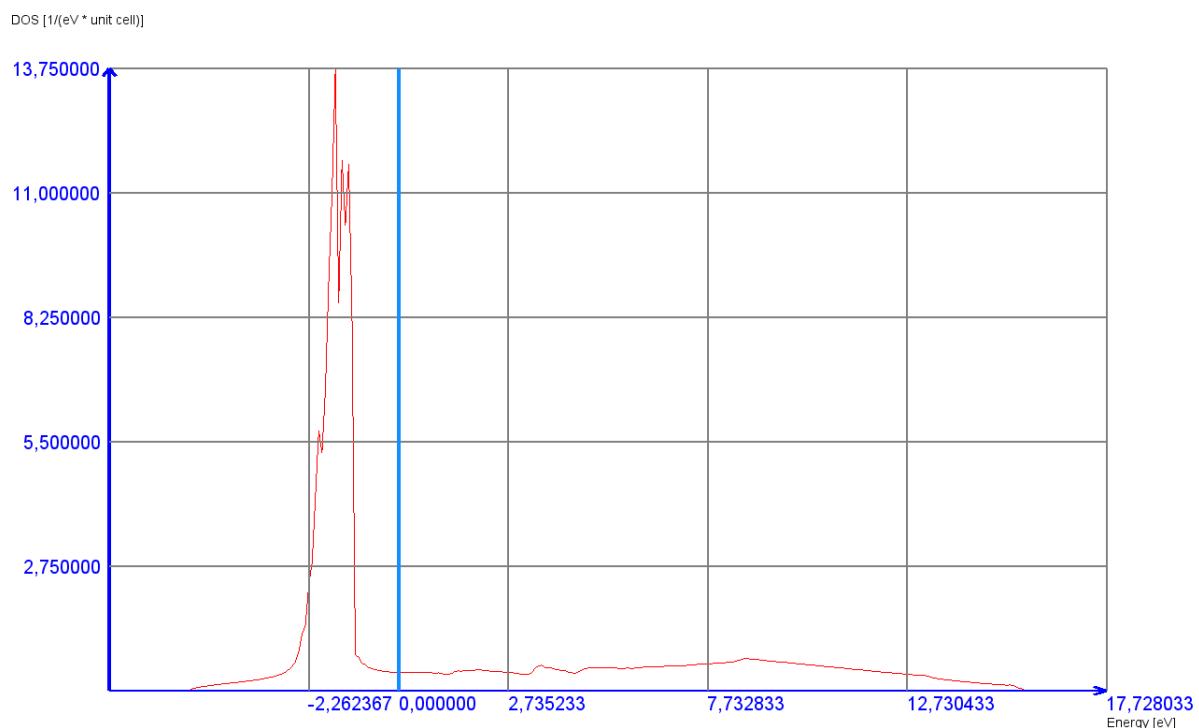


Figure 10: Visualisation of total DOS for Cu.

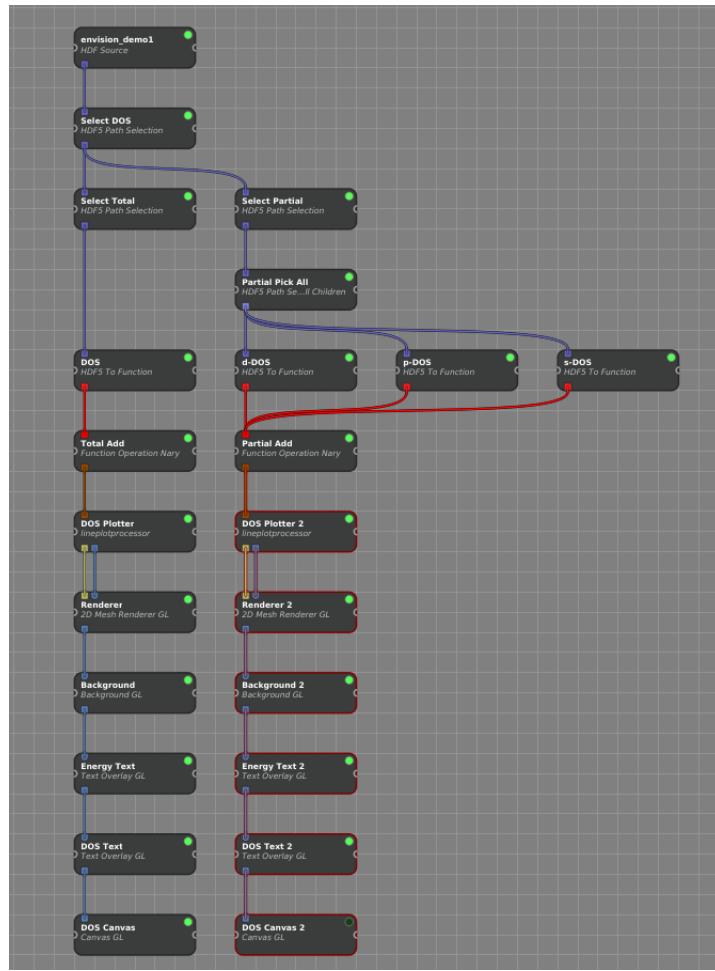


Figure 11: Network for visualisation of total and partial DOS for Cu.

4.5 Interconnected networks

Descriptions and examples of the python scripts generating interconnected networks can be seen below.

4.5.1 Unit cell and charge

In the Python editor in Inviwo, the user can choose to generate a visualisation of unitcell and charge by opening the folder *scripts* and running the file *unitcellAndCharge.py*.

4.5.2 Unit cell and ELF

In the Python editor in Inviwo, the user can choose to generate a visualisation of unitcell and ELF by opening the folder *scripts* and running the file *unitcellAndElf.py*. Figure 12 shows what this can look like.

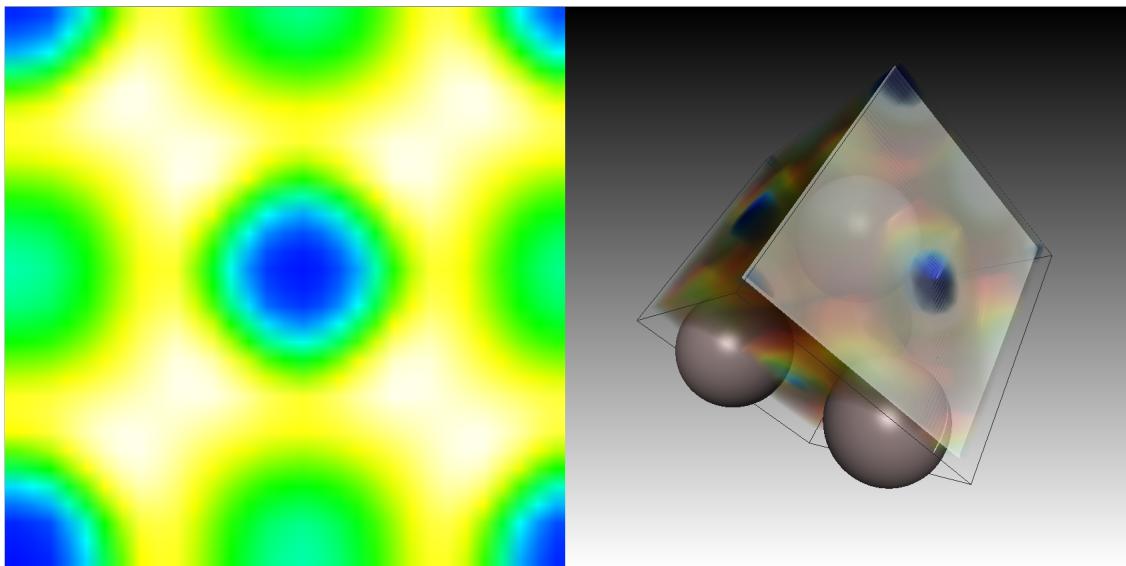


Figure 12: Unit cell and ELF data for aluminum visualised side by side, with volume raycasting and slice function.

4.5.3 Unit cell and DOS

In the Python editor in Inviwo, the user can choose to generate a visualisation of unitcell and density of states by opening the folder *scripts* and running the file *unitcellAndDos.py*. Here the user can pick an atom for which to show partial DOS by clicking on the desired atom in the unit cell. See figure 13 for an example of what this can look like.

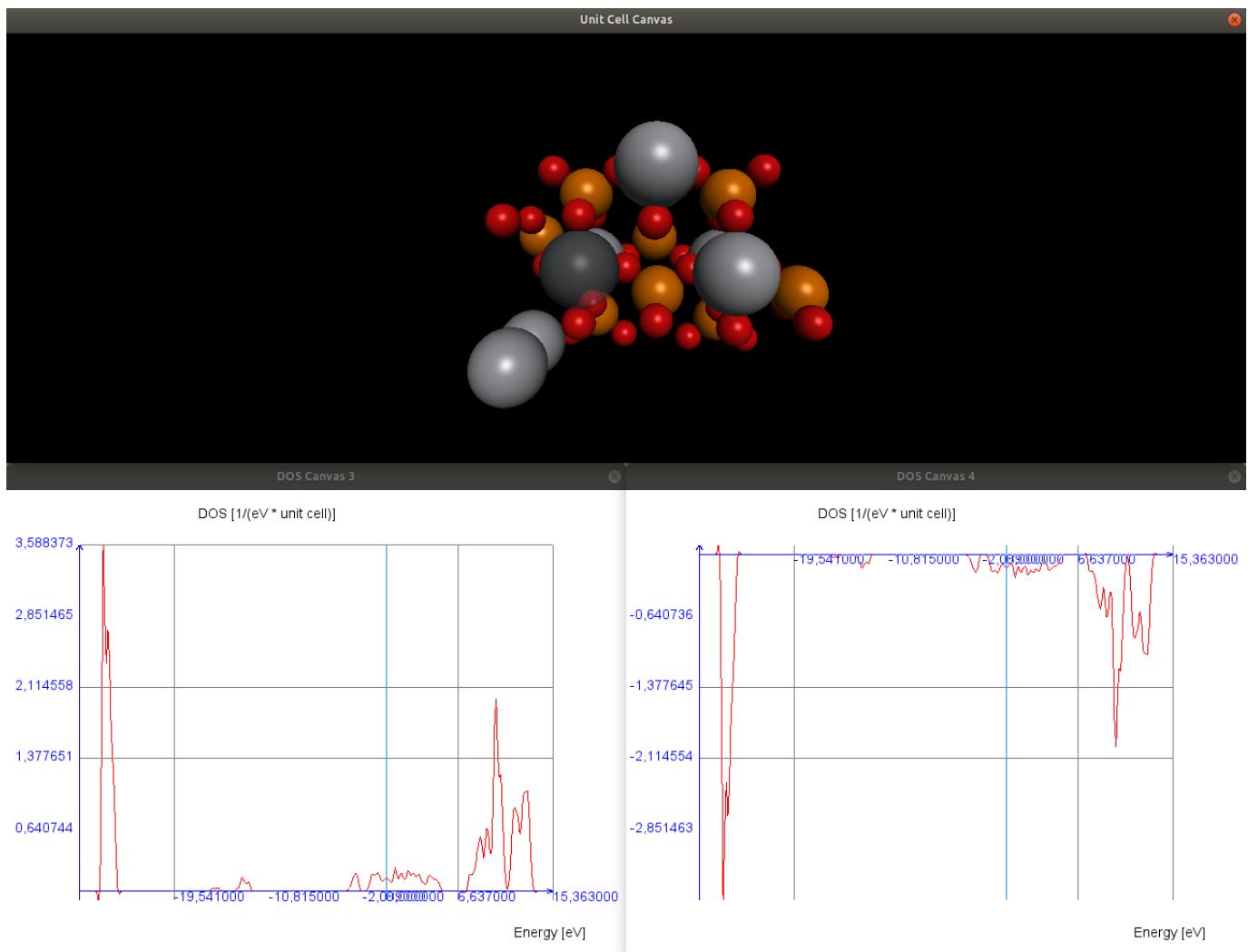


Figure 13: Visualisation of unit cell data and partial DOS of a titanium atom. The two graphs show DOS for electron spin up and electron spin down.

B Designspecifikation

Designspecifikation

Redaktör: Anders Rehult

Version 1.1

Status

Granskad	VB	18-05-25
Godkänd	Johan Jönsson	18-04-11

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	v
1 Inledning	1
1.1 Parter	1
1.2 Projektets bakgrund	1
1.3 Syfte och mål	1
1.4 Användning	2
1.5 Begränsningar	2
1.6 Definitioner	2
2 Översikt av systemet	3
2.1 Beroenden till andra system	3
2.2 Delsystem	3
2.3 Designfilosofi	4
3 Datakonvertering	4
3.1 HDF5	4
3.1.1 HDF5-struktur för VASP-data	6
3.2 VASP	7
3.2.1 Kristallstruktur	7
3.2.2 Elektrontäthet	8
3.2.3 Tillståndstäthet	9
3.2.4 Fermi-ytor	10
3.2.5 Dynamisk kristallstruktur	10
3.3 Elk	10
4 Visualisering	11
4.1 Utritning	11
4.2 Användarindata	12
4.3 Interaktivitet	13
4.4 HDF5-indata	13
4.5 Kristallstruktur	13
4.5.1 Översiktlig beskrivning	13
4.5.2 Inviwo-nätverk för visualisering av kristallstruktur	13
4.6 Elektrontäthet	14
4.6.1 Översiktlig beskrivning	14

4.6.2	Inviwo-nätverk för visualisering av elektrontäthet	14
4.7	Tillståndstäthet	15
4.7.1	Översiktlig beskrivning	15
4.7.2	Inviwo-nätverk för visualisering av tillståndstäthet	15
4.8	Fermi-ytor	17
4.8.1	Inviwo-nätverk för Brillouin-zonuppritning	17
4.8.2	Inviwo-nätverk för Fermi-yte-uppritning	18
4.9	Dynamisk visualiseing av kristallstruktur	19
4.9.1	Översiktlig beskrivning	19
4.9.2	Inviwo-nätverk för dyanmisk visualisering av kristallstruktur	19
Referenser		21
Bilaga A	HDF5-datastruktur	22
Bilaga B	HDF5-datastruktur i två delar	23

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-02-27	Första utkast.	Projektgruppen	SE
0.2	2018-03-05	Andra utkast.	Projektgruppen	DOK
0.3	2018-03-28	Tredje utkast.	Projektgruppen	VB
1.0	2018-03-28	0.3 godkänd.	Projektgruppen	VB
1.1	2018-03-28	Uppdaterat licens för bilder.	Projektgruppen	VB

1 Inledning

Dokumentet är en designspecifikation för kandidatprojektet i visualisering av elektronstrukturer. Visualisering av elektronstrukturer är ett av projekten i kursen TFYA75 vid Linköpings universitet. Designspezifikationen är en fördjupning av systemskissen och beskriver detaljerat hur systemet kommer att designas.

1.1 Parter

I Tabell 1 listas de olika parter som är inblandade i projektet.

Roll	Namn	E-post
Beställare	Rickard Armiento	rickard.armiento@liu.se
Expert	Rickard Englund	rickard.englund@liu.se
Handledare	Johan Jönsson	johan.jonsson@liu.se
Projektledare	Anders Rehult	andre449@student.liu.se

Tabell 1: Parter i organisationen.

1.2 Projektets bakgrund

Ett viktigt verktyg inom teoretisk fysik är elektronstrukturberäkningar. Med hjälp av dessa beräkningar går det att förutsäga hur material med vissa egenskaper kommer att bete sig.

För att förstå beräkningarna är det viktigt att kunna analysera data som beräknats. I vissa fall förenklas detta med hjälp av visualisering och i andra fall krävs det mer eller mindre att man visualisera sin data för att förstå den. Ett kraftfullt verktyg för att visualisera data är visualiseringsprogrammet Inviwo.

Inviwo är ett forskningsverktyg som utvecklats vid Linköpings universitet och ger användaren möjlighet att styra visualisering med hjälp av programmering i Python 3 eller grafiskt. Det tillhandahåller även användargränssnitt för interaktiv visualisering.

1.3 Syfte och mål

Målet med projektet är att utveckla ett system för visualisering av resultat av elektronstrukturberäkningar. Detta ska göras i visualiseringsverktyget Inviwo och systemets funktionalitet ska demonstreras genom att använda det för att illustrera resultat från befintliga beräkningar. I och med att den framtagna mjukvaran ämnas användas i forskningssammanhang måste projektet hålla en hög vetenskaplig och teknisk kvalitet.

Utöver det konkreta målet med framtagandet av mjukvara för visualisering ska även projektet ge projektmedlemmarna erfarenhet av att arbeta i projekt och utöka deras förmåga till analytiskt och fysikaliskt tänkande för att ge värdefull erfarenhet inför arbetslivet.

1.4 Användning

Denna produkt kommer användas vid Linköpings universitet för att analysera data från elektronstruktursberäkningar.

1.5 Begränsningar

I projektet kommer visualiseringssverktyget Inviwo och programmeringspråken Python och C++ användas. Det kommer inte utredas om det är bättre att använda andra verktyg.

1.6 Definitioner

API (Application Programming Interface) är en specifikation av hur olika applikationer kan använda och kommunicera med en specifik programvara. Detta utgörs oftast av ett dynamiskt länkat bibliotek. [11]

BSD2 är en licens för öppen källkod. [3]

C++ är ett programmeringsspråk. [4]

I Inviwo används C++ för att skriva programkod till processorer.

Elk är ett program som använder sig av metoden linjäriserad förstärkt planvåg (Linearized Augmented Planewave (LAPW)) för att lösa ekvationer av tätthetsfunktionalteori (Density Functional Theory (DFT)). [6]

Fermi-energi defineras som en energinivå där antalet tillstånd som har en energi lägre än Fermi-energin är lika med antalet elektroner i systemet. [2]

Fermi-yta är, för elektronusers k-punkter i reciproka rymden, isoytan där elektronernas energi är lika med Fermi-energin. [12]

Git är ett decentraliserat versionshanteringssystem. [7]

GUI (Graphical User Interface) är ett grafiskt användargränssnitt. [13]

HDF5 är ett filformat som kan hantera stora mängder data. [8]

Inviwo (Interactive Visualization Workshop) är programvara för visualisering som tillhandahåller en nätverksredigerare för designen av dataflödesnätverk. Noderna i dessa dataflödesnätverk kallas processorer. Indata till nätverket behandlas i dessa processorer och utdata genereras. [15]

Processor är benämningen på ett visuellt funktionsblock i Inviwos nätverksredigerare. I detta dokument avser en processor alltid en inviwoprocessor om inte annat anges.

Python är ett programmeringsspråk. [16]

I Inviwo används Python för att knyta samman processorer.

Reciproka rymden, även känd som k-rummet, är en fouriertransformering av det normala rummet. I reciproka rymden motsvarar varje punkt ett visst k-värde för en partikel. [14]

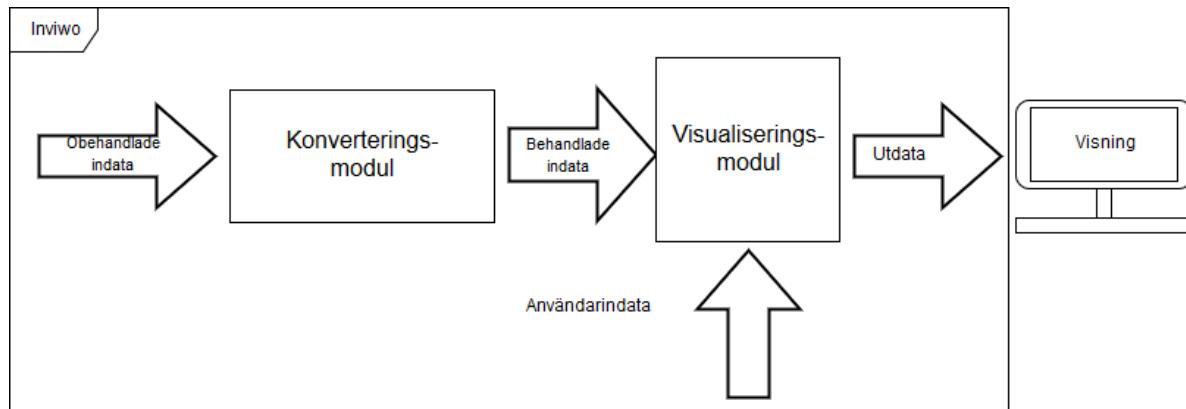
VASP är ett program för modellering på atomnivå, för t.ex. elektronstruktursberäkningar och kvantmekanisk molekyldynamik. [1]

Wigner-Seitz-radie är en radie tillhörande en atom definierad så att summan av volymerna av alla sfärer, bildade för varje atom i enhetscellen utifrån denna radie, är samma som enhetscellens totala volym. I en kristall bestående av flera atomslag finns inget entydigt sätt att välja denna radie. [17]

2 Översikt av systemet

Det färdigställda systemet, se Figur 1, ska kunna hantera indata från VASP och, i mån av tid, även från Elk. Daten kommer från elektronstrukturberäkningar och ska konverteras till HDF5-format för att därefter visualiseras. Användaren ska kunna välja vad som ska visualiseras, kontrollera renderingens utseende och ställa in egenskaper såsom färg, transparens och rotation.

Systemet har delats in i delsystem och dessa kommer utvecklas och testas enskilt i den mån det inte finns beroenden till andra delsystem.



Figur 1: Grov design av systemet

2.1 Beroenden till andra system

Projektet använder sig av Inviwo för att tillhandahålla ett gränssnitt, behandla data och rendera visualiseringar.

2.2 Delsystem

Detta avsnitt behandlar de olika delsystemen projektet innehåller.

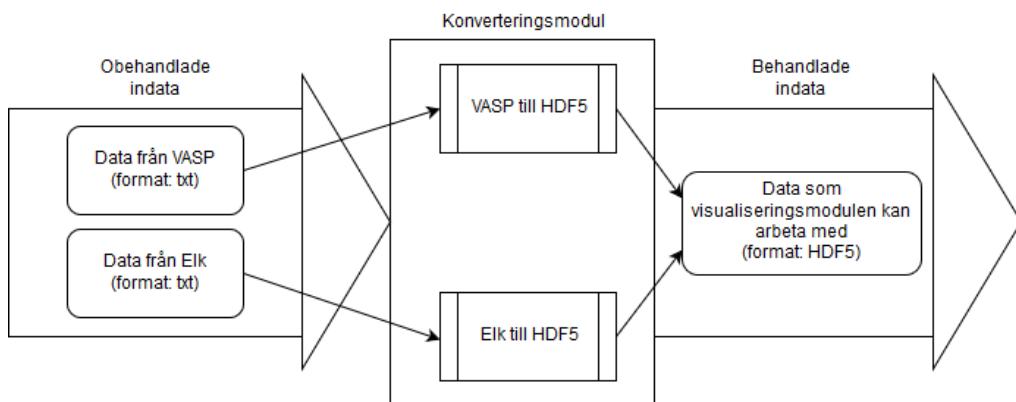
- Datakonvertering VASP
- Datakonvertering Elk
- Visualisering av kristallstruktur
- Visualisering av elektrontäthet
- Visualisering av total och partiell tillståndstäthet
- Visualisering av Fermi-ytor
- Dynamisk visualisering av kristallstruktur

2.3 Designfilosofi

Projektet kommer att drivas med hjälp av versionshanteringssystemet Git och koden kommer vara licensierad med BSD 2, men även utvecklas under Inviwos utvecklaravtal för att, om önskvärt, kunna officiellt integreras i programvaran. Tester kommer skrivas löpande i den mån det är möjligt.

3 Datakonvertering

Detta avsnitt behandlar delsystemet datakonvertering, figur 2 ger en grov översikt av delsystemet. Obehandlad data skickas in, obehandlad data är i form av textfiler med data från beräkningar gjorda i VASP eller Elk. Indata behandlas sedan i konverteringsmodulen. Från konverteringsmodulen kommer behandlad data i HDF5-format.

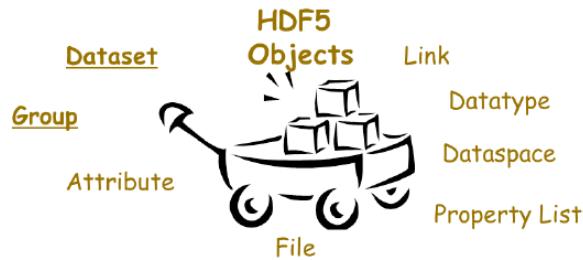


Figur 2: Grov design av konverteringsmodulen

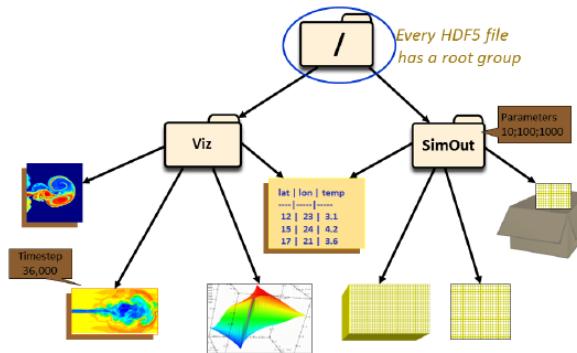
3.1 HDF5

Systemet måste kunna hantera de olika utdatafilerna från VASP. Det bör även kunna hantera utdatafiler från Elk. Detta görs genom att konvertera datan till HDF5-format.

HDF5 är ett filformat som är designat för att hantera stora mängder data på ett flexibelt sätt [8]. HDF5 har flera olika datatyper, se Figur 3, och ett HDF5-objekt som antingen är lagrat på disk eller hålls i minnet är uppdelat i två huvudsakliga underobjekt, nämligen grupper och dataset. [9, s. 3-4]



Figur 3: Överblick av de ingående objektstyperna i HDF5-formatet. Tagen från dokumentet *High Level Introduction to HDF5* [9, s. 3]. Bilden fri att använda enligt HDF Group [5].



Figur 4: Exempel på en HDF5-datastruktur. Tagen från dokumentet *High Level Introduction to HDF5* [9, s. 5]. Bilden fri att använda enligt HDF Group [5].

Alla HDF5-objekt har en rotgrupp som äger alla andra objekt i datastrukturen. Denna grupp innehåller i sin tur all övrig data i form av andra grupper, länkar till andra grupper eller dataset. Dataset innehåller rådata av något slag. Rådata kan i sammanhanget vara bilder, utdata från beräkningar, programdata, etcetera. [9, s. 4-5]

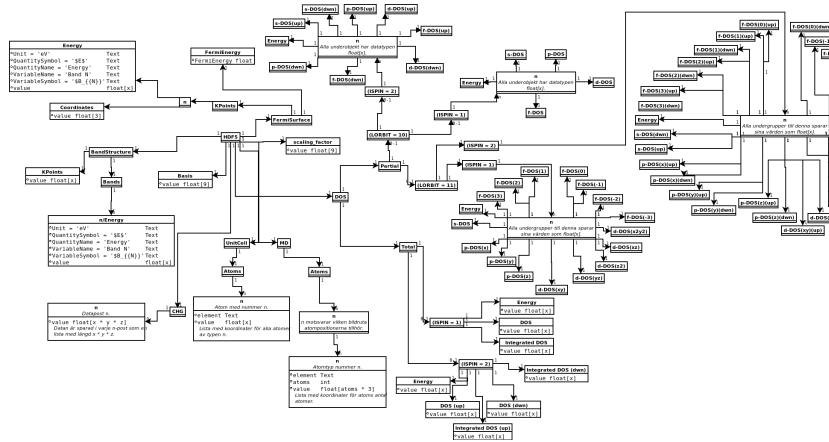
I Figur 4 ses ett exempel på hur en HDF5-datastruktur kan se ut. Den har en rotnod betecknad med ett snedstreck som har två undergrupper med namnen Viz och SimOut. De två undergrupperna innehåller sedan data av olika slag. [9, s. 5]

Strukturen i Figur 4 kan liknas vid ett Unix-filsystem [9, s. 5]. Noden Viz kan således kommas åt via sökvägen /Viz, medan noden SimOut kan kommas åt via sökvägen /SimOut [9, s. 5]. Skulle en extra grupp kallad NewGroup läggas till under SimOut skulle den således kommas åt via sökvägen /SimOut/NewGroup.

Två sökvägar kan peka på samma dataset då data kan delas mellan grupper [9, s. 6].

De övriga objektstyperna går inte igenom i detalj i detta dokument, men finns väl beskrivna i *High Level Introduction to HDF5* [9].

3.1.1 HDF5-struktur för VASP-data



Figur 5: Dataformatet som används när VASP konverteras till HDF5. Se större bild i Bilaga A och B

I Figur 5 finns det HDF5-formatet som VASP-datan ska läsas in till beskrivet. En ruta i diagrammet motsvarar en del i sökvägen i HDF5-objektet, om inte titeln är inom paranteser, då det innebär att rutans barnnoder blir åtkomliga om jämförelsen inom paranteserna är sann. Om en ruta har namngivna fält så innebär fältet *value* den data som är sparad i ett dataset, medan övriga datafält är attribut. En ruta som heter *n*, eller har det i namnet, innebär att det finns flera dataset och att de är numrerade med heltal.

På grund av platsbrist saknar Figur 5 attribut för DOS-datan, men alla fält som innehåller DOS-data i diagrammet, till exempel p-DOS, d-DOS(xy), Energy, etcetera, har attribut som beskriver formatet på datan i fältet. En lista över attributen följer nedan:

- **VariableName** är fältets namn.
- **VariableSymbol** är en symbol som representerar variabeln.
- **QuantityName** är ett för en människa läsligt namn på fältet.
- **QuantitySymbol** är symbol som representerar storheten.
- **Unit** är storhetens fysikaliska enhet.

Den högra texten i varje ruta avser alltid datatypen för en datapost. I de fall datatypen följs av hakparanteser, till exempel *float[x]*, betyder det att data är sparad i en lista där värdet inom hakparanteserna utgör listans längd.

Som exempel så koms tillståndstätheten för den totala tillståndstären åt via */DOS/Total/DOS* oberoende av LORBIT. Ett till exempel är första energin för första bandet i bandstrukturen som nås via */BandStructure/Bands/0/Energy*.

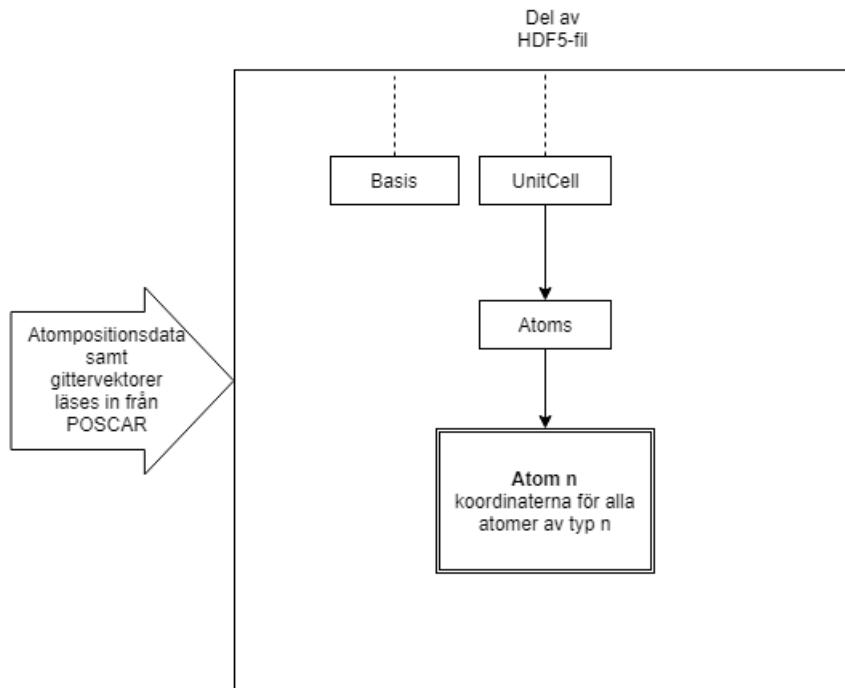
3.2 VASP

Från beräkningsprogrammet VASP fås en rad olika utdatafiler och dessa listas nedan.

- POSCAR innehåller data för enhetscellen samt atompositionsdata.
- CHG innehåller laddningstäthetsdata.
- DOSCAR innehåller tillståndstäthetsdata.
- EIGENVAL innehåller data för alla energier i k-rummet.
- POTCAR innehåller data om atomtyper.
- OUTCAR innehåller all utdata. I detta fall är information om Fermi-energi, den irreducibla 1:a Brillouin-zonens basvektorer och den 1:a Brillouin-zonens symmetriegenskaper vad som söks.
- XDATCAR innehåller data om enhetscell, atompositionsdata för varje beräkningssteg och även atomtyp.
- CONTCAR på samma format som POSCAR men CONTCAR fylls med information om atompresentationer uppdaterats.

3.2.1 Kristallstruktur

Atompositionsdata läses in från POSCAR. Dessa kan vara angivna i kartesiska koordinater och måste därför konverteras till koordinater med gittervektorerna som bas. Gittervektorerna läses in från POSCAR och läggs in i gruppen Basis i HDF5-filen. Atompositionsdatan läggs in i gruppen Unitcell och sorteras in efter atomslag.

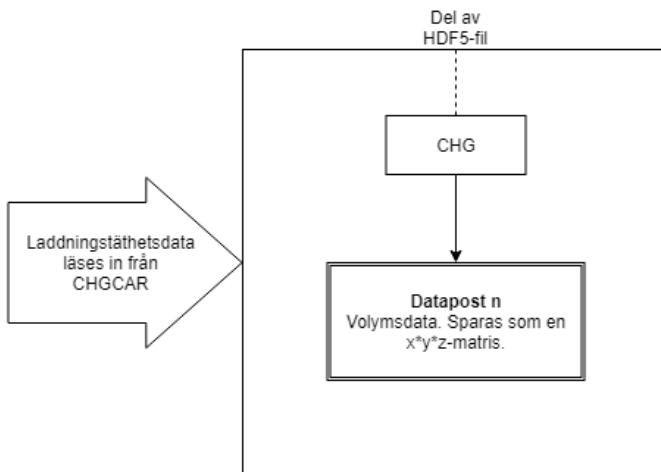


Figur 6: Schematisk bild över sorteringen av data från POSCAR fil.

Figur 6 illustrerar var i HDF5-filen som gittervektorer och atompositionsdata hamnar. Den dubbelstreckade rektangeln representerar datasetet för ett atomslag n och innehåller koordinaterna för alla atomer av det slaget.

3.2.2 Elektrontäthet

Elektrontäthetdata läses in från utdatafilen CHGCAR. Det är alltså volymsdata som behöver läsas in. Denna volymsdata förs sedan in i HDF5-filen i gruppen CHG och sparas som en matris av dimensionen $x * y * z$.

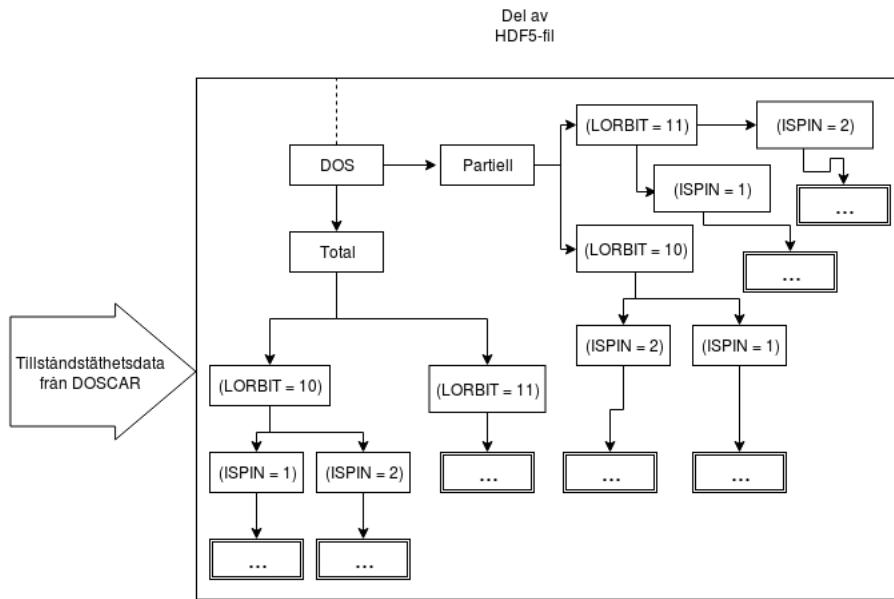


Figur 7: Schematisk bild över sortering av data från CHGCAR fil.

Figur 7 illustrerar sorteringsprocessen för volymsdatablocken från CHGCAR.

3.2.3 Tillståndstäthet

Tillståndstäthetdata läses in från utdatafilen DOSCAR. Flaggorna ISPIN, RWIGS och LORBIT sätts i INCAR-filen och avgör vad som skrivs i DOSCAR-filen. ISPIN-flaggan informerar om spinn har tagits hänsyn till vid beräkningar, RWIGS-flaggan specificerar Wigner-Seitz-radien för varje atomtyp och LORBIT-flaggan (kombinerat med RWIGS) avgör om PROCAR- eller PROOUT-filer skrivs. Vad flaggorna är satta till medförs att datan sorteras in på olika sätt i HDF5-filen. Datatan förs in i HDF5-filen enligt Figur 8.



Figur 8: Schematisk bild över sortering av data från DOSCAR fil.

Figuren illustrerar en del av HDF5-filen som en stor rektangel och inuti den beskrivs uppdelningen av den inlästa datan från DOSCAR-filen. Detaljer kan läsas ur Figur 5.

3.2.4 Fermi-ytor

Varje k-punkts möjliga energier parsas från EIGENVAL-filen. OUTCAR-filen parsas för att hitta Fermi-energin och basvektorer för den irreducibla 1:a Brillouin-zonen i kartesiska koordinater, samt den irreducibla zonens symmetrigrupp.

3.2.5 Dynamisk kristallstruktur

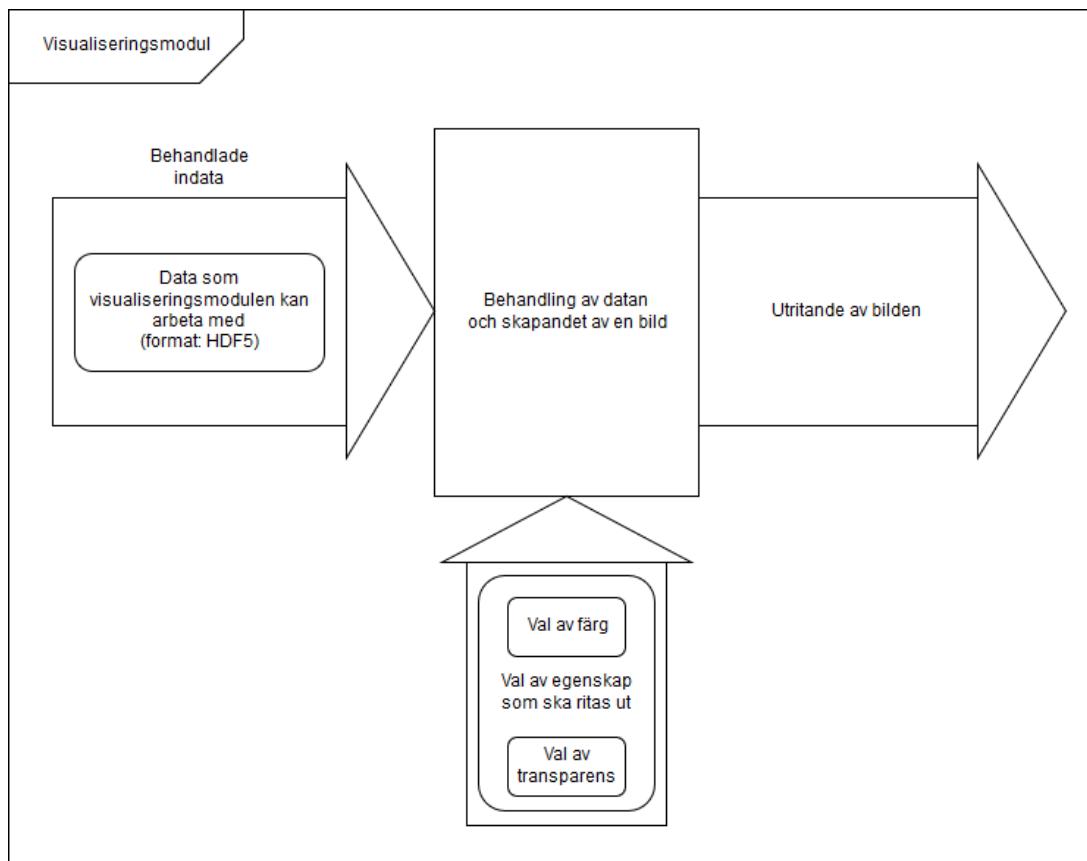
Delsystemet läser in utdatat från XDATCAR. Även data för enhetscell och atomposition fås från XDATCAR. Den data som fås är alltså en lista med data om enhetscell och atomposition för varje beräkningssteg där någon atomposition har ändrats.

3.3 Elk

Om tid finns så kommer även utdata för elektrontäthet från Elk, ett annat beräkningsprogram än VASP, behandlas. Från Elk ska elektrontäthet visualiseras. För detta måste utdatafilerna GEOMETRY.OUT och RHO3D.OUT läsas. I GEOMETRY.OUT finns bland annat data för atompresentationer för att kunna bygga upp geometrin. Från filen RHO3D.OUT får elektrontäthetsdata.

4 Visualisering

Figur 9 visar hur delsystemet Visualisering är uppbyggt. Behandlad samt användarindata skickas in. Användarindata består av val av färg, val av egenskap som ska ritas ut, val av transparens etc. Dessa behandlas sedan i visualiseringsmodulen som skapar en bild utifrån dem. Den utritade bilden skickas sedan ut från modulen. Detta är en allmän beskrivning av visualiseringen, för de olika egenskaperna som ska visualiseras kommer mittenrektangeln i Figur 9 att vara uppbyggd på olika sätt med olika processorer. Detta går mer in på i avsnitt 4.5-4.8.



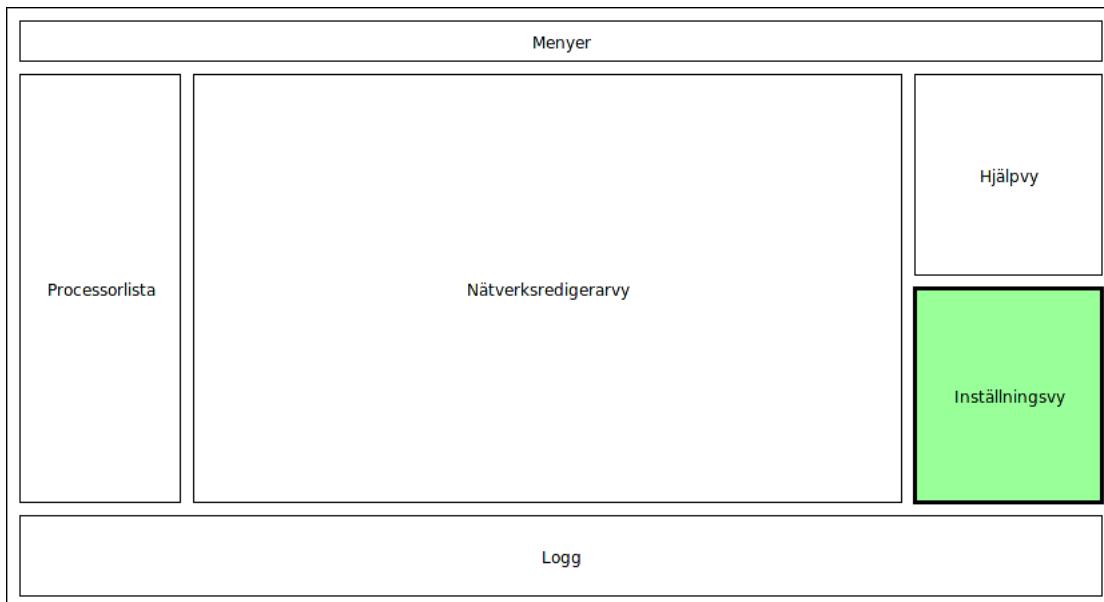
Figur 9: Grov design av delsystemet Visualisering

4.1 Utritning

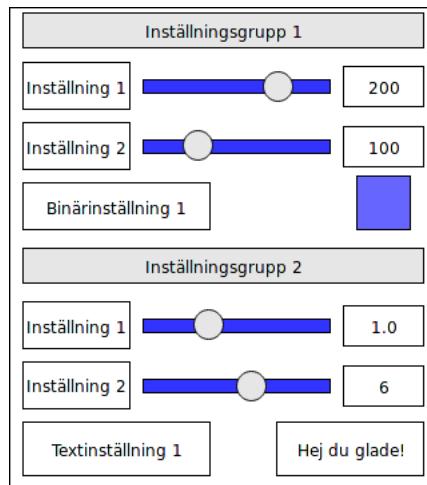
Delsystemet ska rita ut bilder utefter den behandlade datan. Utritningen skall ge en visualisering av den egenskap som modulen behandlar och kan till exempel vara en volymrendering eller en vanlig 2D-graf, beroende på vilket som är mest lättolkat.

Utritningen görs via Inviwos inbyggda funktionalitet för att rendera via OpenGL. Grafisk behandling görs också med Inviwo och OpenGL, där möjligheten finns att både använda färdig funktionalitet och att utveckla nya renderingsfunktioner.

4.2 Användaridata



Figur 10: Skiss som visar det tänkta användargränssnittet.



Figur 11: Skiss som visar den tänkta inställningsrutan i Inviwo.

Användaren kommer kunna ändra inställningar som kontrollerar utseendet på visualiseringarna. Denna indata matas in i Inviwos användargränssnitt, se Figur 10, och i inställningsrutan för en processor, se Figur 11.

Visualiseringsfunktionaliteten kommer att läggas till i form av processorer som via Inviwo tillhandahåller inställningsmöjligheter på det sätt som beskrivits ovan. Således blir inte användarinställningar ett eget separat delsystem, utan kommer att bakas in i visualiseringssystemen. Detta görs genom att processorerna skriver ut exponera de inställningar som användaren är menad att justera. Inviwo kommer då automatiskt lägga till inställningarna i rutan skisserad i Figur 11.

4.3 Interaktivitet

Användaren ska kunna modifiera visualiseringen genom att reglera ett intervall av värden för någon egenskap, där full transparens fås för alla värden inom intervallet, rotera 3D-bilder etc.

Ny användaridata, som skickas in efter att den första bilden har ritats upp, skickas tillbaka till visualiseringsmodulen för att utföra en ny rendering.

4.4 HDF5-indata

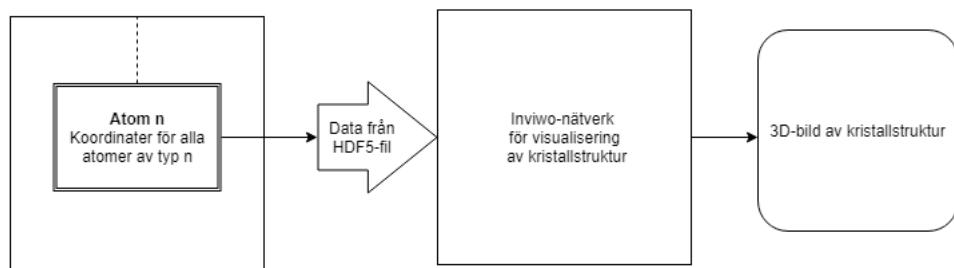
I Inviwo 0.9.9 finns en modul som tillåter importering av data från HDF5-fil [10]. Denna kommer användas för att ladda in data som sedan ska visualiseras. Se avsnitt 3.1 för beskrivning av uppbyggnaden av HDF5-filen.

4.5 Kristallstruktur

Kristallstruktur ska visualiseras som atompositioner i enhetscellen. Sfärer som representerar atomer ska ritas ut genom volymsrendering. Nedan följer en översiktig beskrivning av hur detta ska byggas upp.

4.5.1 Översiktig beskrivning

Koordinaterna (tre dimensioner) för atomerna läses in från HDF5-fil. Utifrån dessa positioner ritas sedan atomerna ut. Färginställningar för atomerna kan ändras av användaren.



Figur 12: Översiktligt flödesdiagram över dynamisk visualisering av kristallstruktur.

I Figur 12 beskrivs detta med ett flödesschema. Inviwo-nätverket näst sist i flödesschemat skickar ut en 3D-bild av kristallstrukturen där atomerna har placerats ut på sina positioner som hämtats från HDF5-filen.

4.5.2 Inviwo-nätverk för visualisering av kristallstruktur

En 3D-bild ska ritas ut vilket kommer kräva ett antal steg. Varje steg kommer att implementeras i Inviwo med en eller flera processorer.

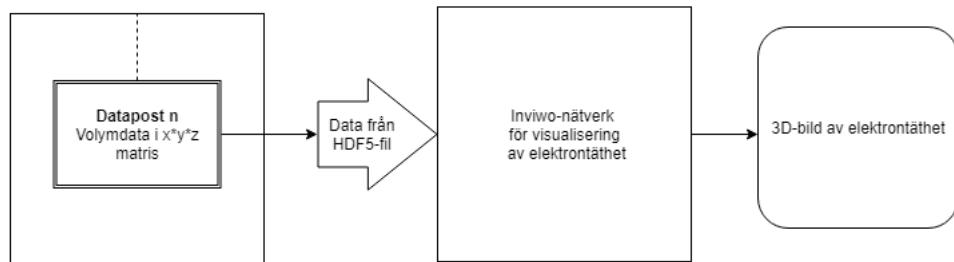
- HDF5-inläsning
 - Ladda in data från HDF5-fil.
- Koordinatläsning
 - Läsa in koordinaterna för atomerna från HDF5-fil.
- Mesh-konstruering
 - Bygga upp ett mesh utifrån koordinaterna från koordinatläsar-processorn.
- Canvas
 - Visa 3D-bilden.

4.6 Elektrontäthet

Elektrontäthet visualiseras genom volymsrendering. Nedan följer en översiktig beskrivning av hur detta ska byggas upp.

4.6.1 Översiktig beskrivning

Datan som ritas ut är sannolikheten för att hitta elektroner på olika positioner. Sannolikheterna visualiseras med färger och skiftande transparens.



Figur 13: Översiktligt flödesdiagram över visualisering av elektrontäthet.

4.6.2 Inviwo-nätverk för visualisering av elektrontäthet

En 3D-bild ska ritas upp vilket kräver ett antal steg som implementeras med processorer i Inviwo.

- HDF5-inläsning
 - Ladda in data från HDF5-fil.
- Konvertering till volym

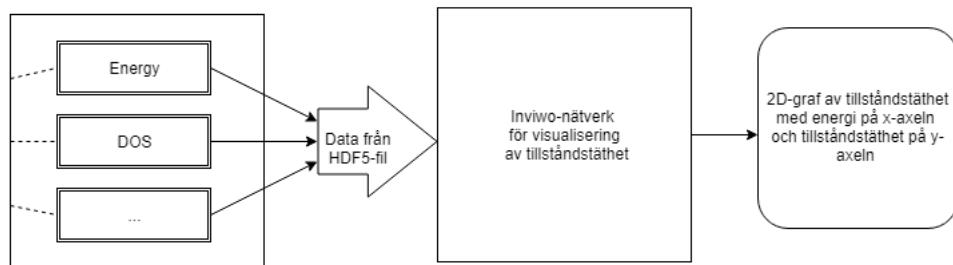
- Konvertera datan till en volym.
- Canvas
 - Visa 3D-bilden.

4.7 Tillståndstäthet

Tillståndstätheten, total och partiell, ska visualiseras med en 2D-graf. Nedan följer en översiktlig beskrivning av hur detta ska byggas upp.

4.7.1 Översiktlig beskrivning

Data hämtas från dataseten i gruppen tillståndstäthet i HDF5-filen. För partiell-tillståndstäthet hämtas energi, tillståndstäthet och så vidare för varje atom. Detta ska sedan plottas i en 2D-graf med tillståndstätheten på y-axeln och energin på x-axeln. Energin ska ha sitt nollställe vid Fermi-energin.



Figur 14: Översiktligt flödesdiagram över visualisering av tillståndstäthet.

I figur 14 beskrivs förloppet översiktligt. För partiell-tillståndstäthet hämtas alltså data i första rektangeln i figuren för varje atom. För total-tillståndstäthet är detta datasetten för gruppen Total. Ett Inviwo nätverk för visualisering av tillståndstäthet måste skapas, detta nätverk illustreras i figur 14 av en stor kvadrat som skickar ut en 2D-graf på en kanvas.

4.7.2 Inviwo-nätverk för visualisering av tillståndstäthet

En 2D-graf ska ritas upp vilket kräver ett antal steg. Dessa implementeras i Inviwo på en eller flera processorer.

- HDF5-inläsning
 - Ladda in data från HDF5-fil.
- Plottning
 - Plotta en graf av den laddade datan.

- Canvas
 - Visa den plottade grafen.

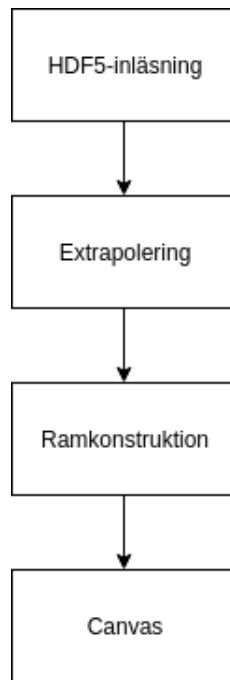
4.8 Fermi-ytor

Fermi-ytor och godtyckliga energi-isoytor ska kunna ritas upp i reciproka rymden, k-rummet. Detta görs genom att den 1:a Brillouin-zonen ritas upp och inuti denna ritas ytan.

4.8.1 Inviwo-nätverk för Brillouin-zonuppritning

Fermi-ytor ritas upp inuti 1:a Brillouin-zonen, som är den samling k-punkter i reciproka rymden vars närmsta gitterpunkt är origo. Den irreducibla 1:a Brillouin-zonen är den minsta samling vektorer som behövs för att extrapolera den fullständiga zonen, som kan byggas upp från kristallens symmetrigrupp. Brillouin-zonuppritningsnätverkets uppgift är att ta den irreducibla 1:a Brillouin-zonens basvektorer i kartesiska koordinater samt dess symmetriegenskaper, beräkna alla k-punkter som ingår i den 1:a Brillouin-zonen och rita upp en ram för denna. Basvektorerna parsas från OUTCAR-filen av en VASP-beräkning rörande Fermi-ytor, och återfinns på HDF5-formatet.

Inviwo-nätverket för att rita upp den 1:a Brillouin-zonen består av ett antal steg som beskrivs nedan. Se figur 15 för en skiss av hur dessa förhåller sig till varandra. Varje steg implementeras i Inviwo i form av en eller flera processorer.



Figur 15: Översiktlig beskrivning av de nödvändiga stegen för Brillouin-zonuppritning

- HDF5-inläsning
 - Hämtar data om basvektorerna för den irreducibla 1:a Brillouin-zonen
- Extrapolering
 - Använder basvektorerna och symmetrigruppen från HDF5-inläsningen för att extrapola vilka punkter som ingår i den 1:a Brillouin-zonen

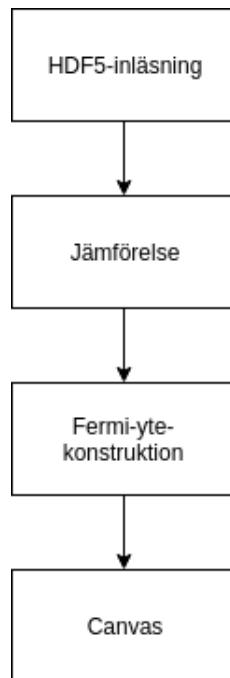
- Ramkonstruktion
 - Beräknar vilka k-punkter som utgör ”ramen” för 1:a Brillouin-zonen
- Canvas
 - Ritar upp ramen

4.8.2 Inviwo-nätverk för Fermi-YTE-uppritning

En sorts utdata från VASP-beräkningar är möjliga energier för alla k-punkter i reciproka rymden. Fermi-YTE-uppritningsnätverkets uppgift är att ta dessa punkter med tillhörande energier och beräkna samt rita upp isoytan där energin är lika med Fermi-energin inuti ramarna för 1:a Brillouin-zonen. Energierna för alla k-punkter parsas från EIGENVAL-filen. Fermi-energin parsas från OUTCAR-filen. Dessa data återfinns på HDF5-form.

Denna visualisering ska även kunna göras för en godtycklig energi, inte bara för Fermi-energin. Denna energi väljs med hjälp av en slider eller genom att manuellt skriva in önskad energi.

Inviwo-nätverket för att rita upp Fermi-energin består av ett antal steg som beskrivs nedan. Se figur 16 för en skiss av hur dessa förhåller sig till varandra. Varje steg implementeras i Inviwo i form av en eller flera processorer.



Figur 16: Översiktlig beskrivning av Inviwo-nätverket för Fermi-YTE-uppritning

- HDF5-inläsning
 - Hämtar data om Fermi-energin samt de möjliga energierna hos alla k-punkter inuti 1:a Brillouin-zonen
- Jämförelse

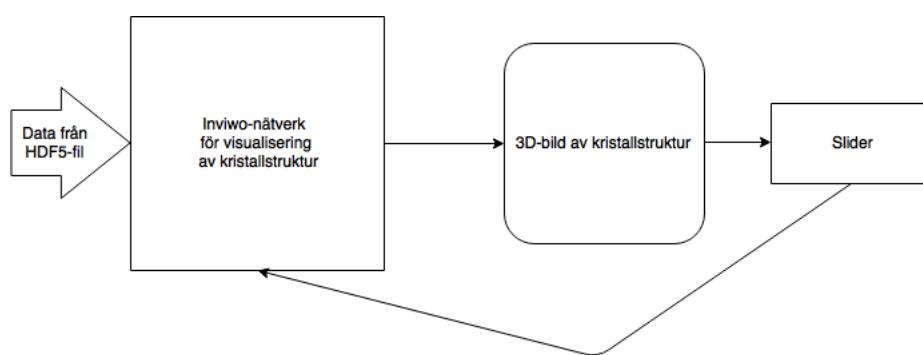
- Jämför de möjliga energierna för varje k-punkt med Fermi-energin för att se vilka k-punkter som ingår i Fermi-ytan
- Fermi-yte-konstruktion
 - Beräknar vilka k-punkter som utgör Fermi-ytan
- Canvas
 - Ritar upp Fermi-ytan

4.9 Dynamisk visualiseing av kristallstruktur

Dynamisk visualisering baserad på en serie atompositioner. Användaren ska med hjälp av en slider kunna ändra visualiseringen.

4.9.1 Översiktlig beskrivning

Koordinaterna (tre dimensioner) för atomerna läses in från HDF5-fil. Utifrån dessa positioner ritas sedan atomerna ut. Användaren kan sedan med hjälp av en slider ändra tiden och se hur visualiseringen ändrar sig därefter. Se figur 17 för ett enklare flödesschema.



Figur 17: Översiktligt flödesdiagram över dynamik.

4.9.2 Inviwo-nätverk för dyanmisk visualisering av kristallstruktur

En 3D-bild ska ritas ut med hjälp av ett antal steg. Det är denna 3D-bild som sedan kommer att uppdateras om användaren väljer att justera slidern för tid. Varje steg kommer att implementeras i Inviwo med en eller flera processorer.

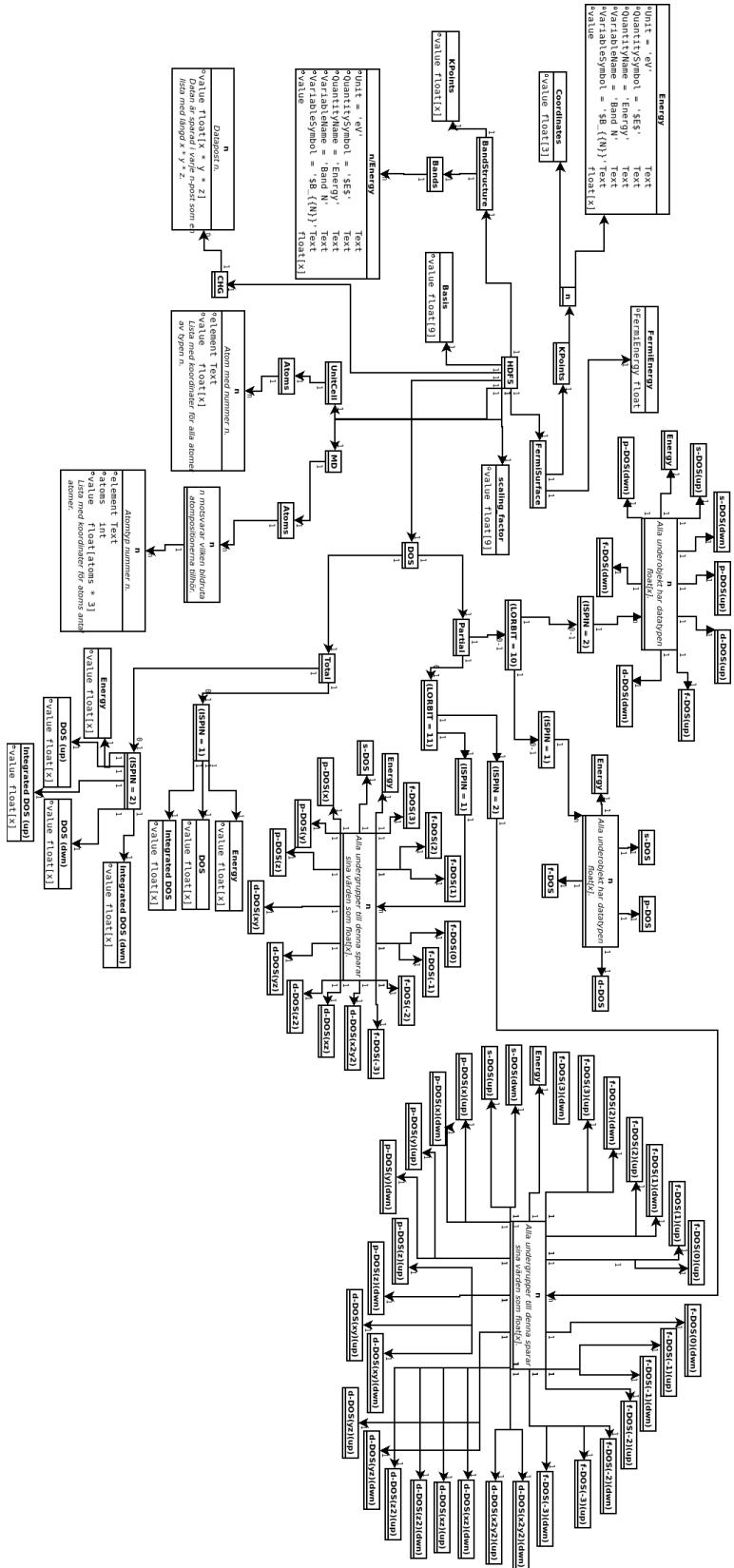
- HDF5-inläsning
 - Ladda in data från HDF5-fil.
- Koordinatläsning
 - Läsa in koordinaterna för atomerna från HDF5-fil.
- Mesh-konstruering

- Bygga upp ett mesh utifrån koordinaterna från koordinatläsar-processorn.
- Canvas
 - Visa 3D-bilden.
- Slider
 - Ändra tiden.
- Koordinatläsning...

Referenser

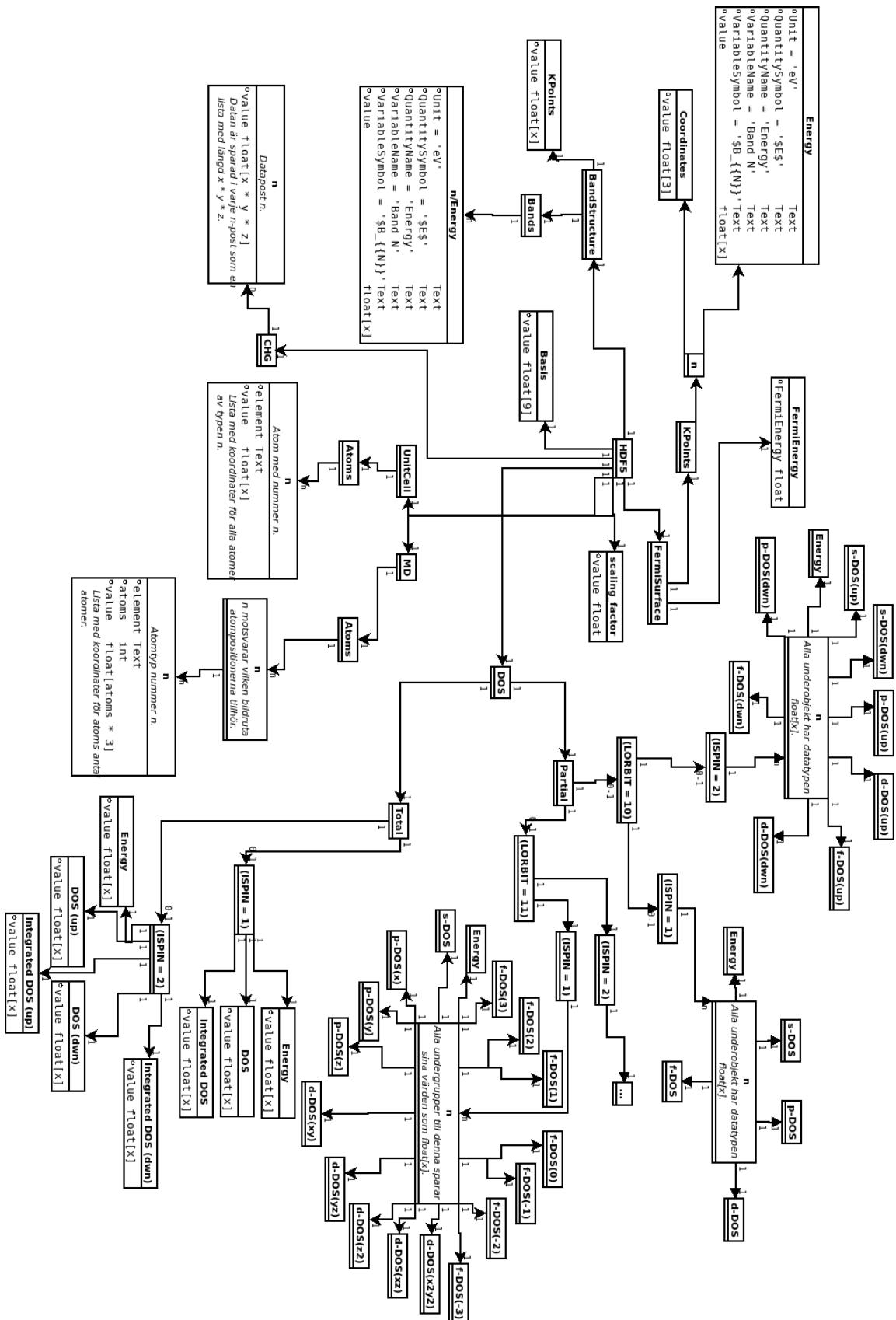
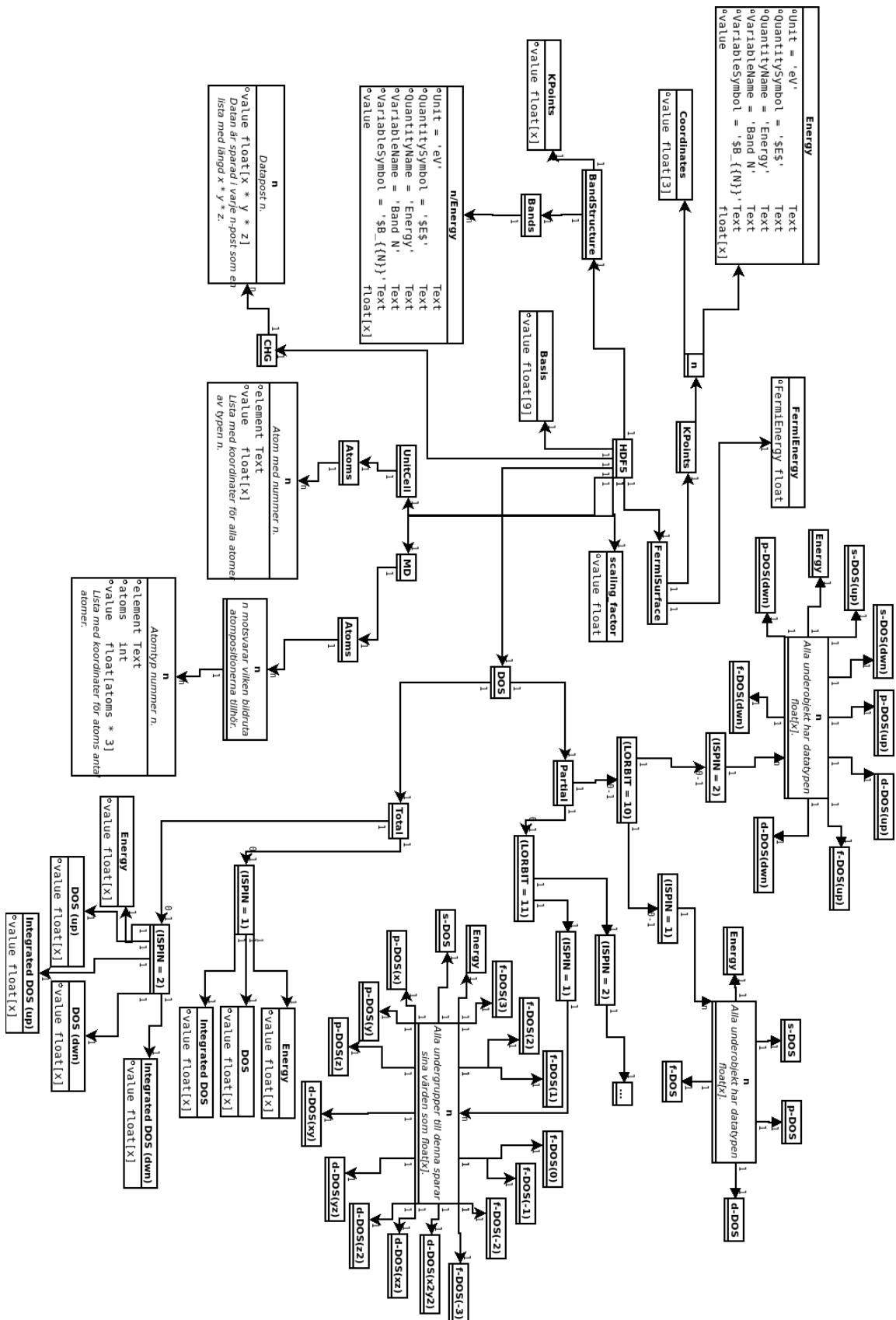
- [1] *About VASP*. URL: <https://www.vasp.at/index.php/about-vasp/59-about-vasp> (hämtad 2018-02-22).
- [2] Neil Ashcroft och David Mermin. *Solid State Physics*. 1976, s. 141.
- [3] *BSD2*. URL: <https://opensource.org/licenses/BSD-2-Clause> (hämtad 2018-02-23).
- [4] *C++*. URL: <http://www.cplusplus.com/info/description/> (hämtad 2018-02-23).
- [5] *Copyright Notice and License Terms for HDF5 (Hierarchical Data Format 5) Software Library and Utilities*. URL: <https://support.hdfgroup.org/ftp/HDF5/releases/COPYING> (hämtad 2018-05-25).
- [6] *ELK*. 31 mars 2017. URL: <http://elk.sourceforge.net/elk.pdf> (hämtad 2018-02-22).
- [7] *Git*. URL: <https://git-scm.com> (hämtad 2018-02-23).
- [8] The HDF Group. *Hierarchical Data Format, version 5*. 1997-2018. URL: <https://support.hdfgroup.org/HDF5/> (hämtad 2018-02-21).
- [9] The HDF Group. *High Level Introduction to HDF5*. 23 sept. 2016. URL: <https://support.hdfgroup.org/HDF5/Tutor/HDF5Intro.pdf> (hämtad 2018-02-21).
- [10] *Inviwo Release. HDF5 module*. URL: <https://www.inviwo.org/2017/10/04/inviwo-0-9-9-released/> (data) (hämtad 2018-02-27).
- [11] *Nationalencyklopedin. API*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-> (data) (hämtad 2018-02-23).
- [12] *Nationalencyklopedin. Fermi-yta*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/fermi-yta> (hämtad 2018-02-23).
- [13] *Nationalencyklopedin. GUI*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-> (data) (hämtad 2018-02-23).
- [14] *Nationalencyklopedin. reciproka rymden*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/reciproka-rymden> (hämtad 2018-02-27).
- [15] *Overview - Inviwo*. URL: <http://www.inviwo.org/overview/> (hämtad 2018-02-23).
- [16] *Python*. URL: <https://www.python.org> (hämtad 2018-02-23).
- [17] *VASP. Wigner-Seitz-radie*. URL: <https://cms.mpi.univie.ac.at/vasp/vasp/RWIGGS.html> (hämtad 2018-03-08).

A HDF5-datastruktur

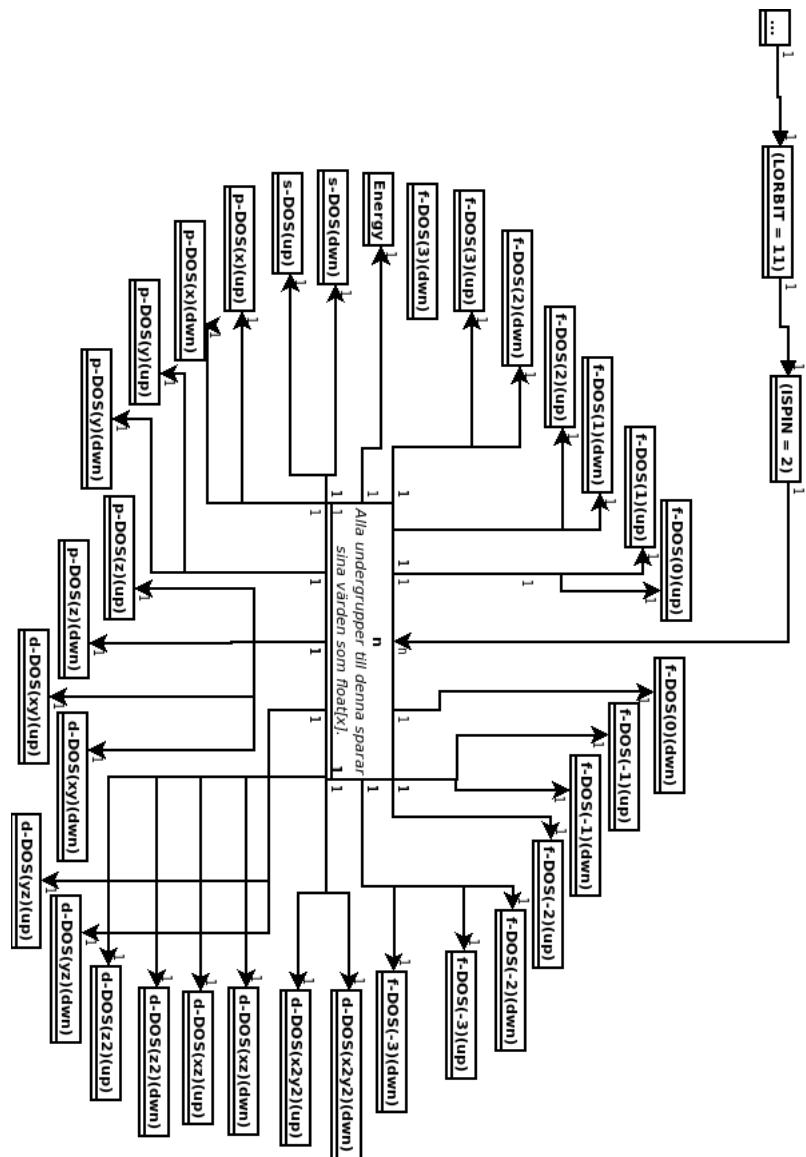


Figur 18: Dataformatet som används när VASP konverteras till HDF5.

B HDF5-datastruktur i två delar



Figur 19: Dataformatet som används när VASP konverteras till HDF5 del 1.



Figur 20: Dataformatet som används när VASP konverteras till HDF5 del 2.

C Fördjupningsarbete - Fermi-ytor

Fermi-ytor

Fördjupningsarbete

Anders Rehult och Marian Brännvall

Version 0.4

Status

Granskad	PL	18-05-25
Godkänd		

Sammanfattning

Fermi-ytor är viktiga teoretiska verktyg för att förstå elektroners beteenden i fasta material. Dessa ytor kan tas fram genom kvantmekaniska beräkningar och möjliggör en djupare förståelse av materialens elektriska egenskaper. Detta fördjupningsarbete ämnar svara på frågorna ”*vad är Fermi-ytor?*” och ”*hur kan dessa beräknas numeriskt?*”. Detta görs rent kvalitativt genom en introduktion till de koncept inom fasta tillståndets fysik som ligger till grund för de beräkningar som beskrivs i arbetets andra hälft. Vägen från Schrödingerekvation till Fermi-tyta stegas sedan, med mellanleden bandstruktur och tillståndstäthet.

PROJEKTIDENTITET

2018/VT,
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-03-20	Första utkast.	PL, DOK	DOK
0.2	2018-03-27	Andra utkast.	PL, DOK	DOK
0.3	2018-05-07	Tredje utkast.	PL, DOK	DOK
0.4	2018-05-25	Fjärde utkast.	PL, DOK	PL

Innehåll

Dokumenthistorik	iv
1 Inledning	1
2 Syfte	1
3 Förberedande teori	1
3.1 Bakgrund	1
3.2 Kristallstruktur	2
3.2.1 Terminologi	2
3.2.2 Olika typer av Bravais-gitter med kubisk symmetri	3
3.3 Fermi-energi	7
3.4 Bandstruktur	7
3.5 Fermi-yta	8
3.6 Brillouin-zon	8
3.7 Tillståndstäthet	9
4 Numeriska metoder för beräkning av Fermi-ytor (utkast)	10
4.1 Approximativ lösning av Schrödingerekvationen	10
4.2 Bandstrukturerberäkning	10
4.3 Tillståndstäthetsberäkning	11
4.4 Fermi-yteberäkning	11

1 Inledning

De stora upptäckterna inom fysik under 1800-talet skedde inom de makroskopiska disciplinerna så som termodynamik och elektromagnetism. Under 1900-talet kom förståelsen för de mikroskopiska disciplinerna att revolutioneras bland annat tack vare upptäckten av röntgenstrålar och radioaktivitet [1]. Under detta århundrade började även den teoretiska grunden till kvantmekaniken att utvecklas av bland annat Erwin Schrödinger och Werner Heisenberg, som en fortsättning på arbete utfört av bland annat Max Planck, Albert Einstein, Niels Bohr och Louis de Broglie.

Frågor kring hur temperatur påverkar elektrisk och termisk konduktivitet hos metaller och legeringar, typiska ingenjörsproblem, kom att bli intressanta även inom den akademiska fysiken eftersom de blev viktiga för teorier kring elektroner [1].

Det som kom att bli det fasta tillståndets fysik var till en början ett splittrat forskningsområde. Först med kvantmekaniken kunde beskrivningar av olika egenskaper hos fasta ämnen förenas, men även efter detta dröjde det innan det fasta tillståndets fysik blev en självständig disciplin [1].

Bandstrukturteorin var en av de tidiga framgångarna från insikten under 1920-talet att man, med kvantmekanikens hjälp, kunde förstå elektroners beteende i fasta ämnen. Teorin utvecklades främst av Felix Bloch, som också har namngivit flera koncept inom bandstrukturteorin såsom Bloch-vågor. Förståelsen av elektronernas beteenden bidrog till förståelsen av skillnaden mellan metaller och isolatorer. Båda innehåller elektroner som nästan är fria men i isolatorer är elektronernas energinivåer grupperade i helt fyllda band (se kapitel 3.4), och material med denna egenskap leder inte ström. Ett fast ämne med delvis fyllda band leder däremot ström och dessa har en Fermi-yta. Fermi-ytor är viktiga för att förstå metaller och deras elektriska egenskaper. Vikten av Fermi-ytor blir särskilt tydlig med fysikern Allan Mackintoshs påstående att definitionen av metaller som ”fasta material med en Fermi-yta” kan vara den mest meningsfulla definitionen av metaller som kan ges [2].

2 Syfte

Syftet med detta arbete är att svara på dessa frågor: Vad är Fermi-ytor och hur kan dessa beräknas numeriskt?

3 Förberedande teori

I detta kapitel förklaras några av de termer och koncept som behövs för att förstå de beräkningsmetoder som behandlas i kapitel 4.

3.1 Bakgrund

Fördjupningsarbetet görs som en del av kursen TFYA75, kandidatprojekt i fysik, vid Linköpings universitet. Ett av projekten i denna kurs är ett visualiseringsprojekt. I visualiseringsprojektet för år 2018 ska bland annat Fermi-ytor visualiseras. Denna rapport ämnar bidra till djupare förståelse för dessa ytor.

3.2 Kristallstruktur

En kristall kan modelleras som identiska grupper av atomer i tre dimensioner.

3.2.1 Terminologi

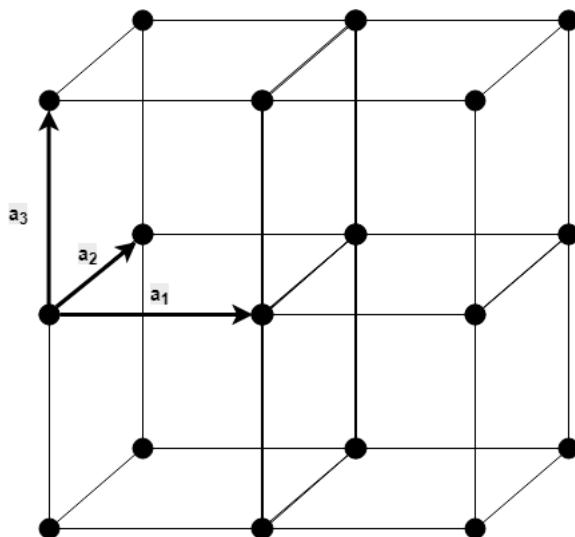
Bas: En ideal kristall är uppbyggd av oändligt upprepade identiska atomgrupper. En sådan atomgrupp kallas bas. En atomgrupp som innehåller minsta möjliga antal atomer kallas för den primitiva basen [3, s. 4].

Gitter: Är den mängd matematiska punkter som basen är kopplad till. Närmare definition av gitter återfinns under beskrivningen av translationsvektorer.

Translationsvektorer: Är de vektorer som gittret kan definieras utifrån. I tre dimensioner kan gittret definieras utifrån translationsvektorerna \mathbf{a}_1 , \mathbf{a}_2 och \mathbf{a}_3 så att atomernas placering i kristallen ser likadana ut från punkt \mathbf{r} som från varje punkt \mathbf{r}' translaterad med en hetalsmultipel av translationsvektorerna, vilket beskrivs av sambandenet (1).

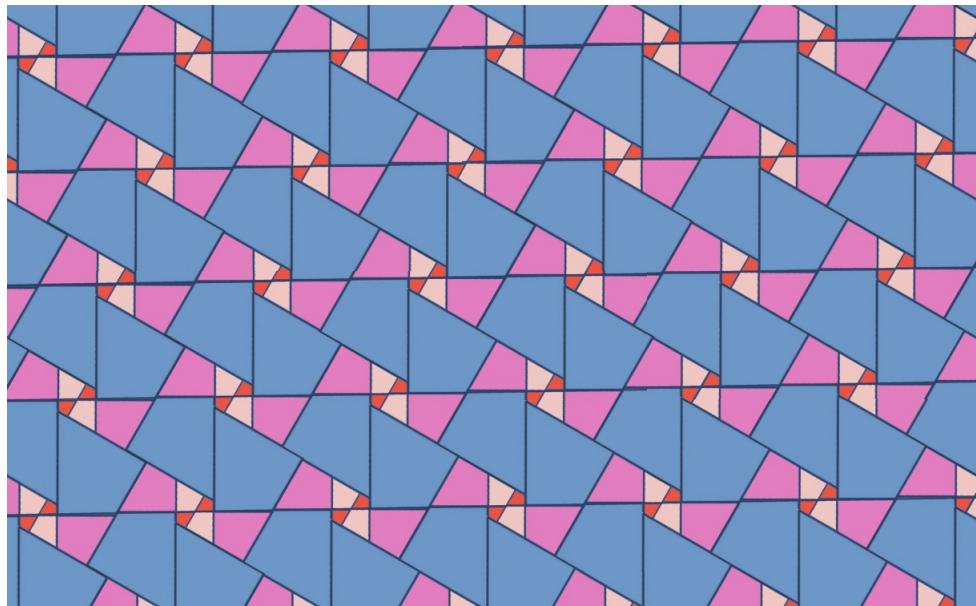
$$\mathbf{r}' = \mathbf{r} + u_1\mathbf{a}_1 + u_2\mathbf{a}_2 + u_3\mathbf{a}_3 \quad (1)$$

Där u_1 , u_2 och u_3 alltså är heltal. Mängden av punkter \mathbf{r}' som definieras av (1) för alla u_1 , u_2 , u_3 definierar gittret [3, s. 4]. I figur 1 ses ett exempel på translationsvektorer i ett gitter.



Figur 1: Translationsvektorer i ett rymdgitter.

Enhetscell: En enhetscell är en volymsdel av en kristall som genom upprepad duplicering och translation kan fylla hela rummet. Se figur 2 för ett exempel på en enhetscell i två dimensioner som vid upprepning fyller hela rummet.



Figur 2: Representation av en enhetscell (en “rektangel” i bilden) som upprepas för att fylla ut hela bilden. Figur av Shtilman A. [4]

Primitiv cell: Den primitiva cellen är den cell som besittande minsta möjliga volym omsluter den primitiva basen. Den primitiva cellen innehåller endast en gitterpunkt och är en enhetscell.

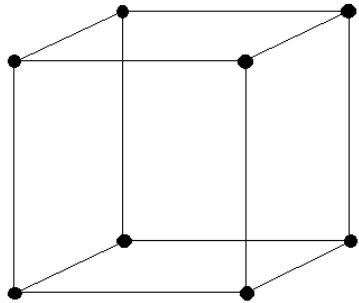
Bravais-gitter: Ett Bravais-gitter är en oändlig mängd diskreta punkter som ser exakt likadan ut oberoende av vilken gitterpunkt den betraktas från [5, s. 64]. När hela rummet fylls av upprepning av en enhetscell fås ett Bravais-gitter. Om mönstret i figur 2 hade upprepats oändligt i alla riktningar så hade alla mängder av punkter i bilden stora nog för att täcka in minst en rektangel varit Bravais-gitter.

Reciproka gitter: Varje Bravais-gitter har en motsvarighet i det reciproka rummet vars form beror av Bravais-gittrets periodicitet. Ett Bravais-gitters reciproka gitter är även det ett reciprokt gitter [5, s. 86].

3.2.2 Olika typer av Bravais-gitter med kubisk symmetri

Det finns tre typer av Bravais-gitter med kubisk symmetri: enkel (eng. simple cubic), ytcentrerad (eng. face-centered cubic) och rymdcentrerad (eng. body-centered cubic).

Enkelt (sc): Detta Bravais-gitter har gitterpunkter i varje hörn av kuben, se figur 3.

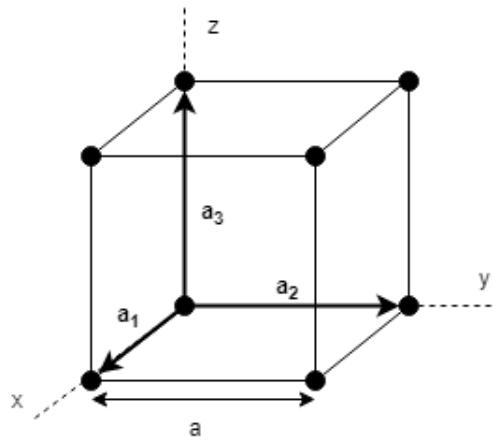


Figur 3: Det kubiska Bravais-gittret, simple-cubic.

Ett enkelt kubiskt Bravais-gitter innehåller en gitterpunkt per cell [3, s. 9-11]. I kartesiska koordinater är ett möjligt val av translationsvektorerna för gittret följande:

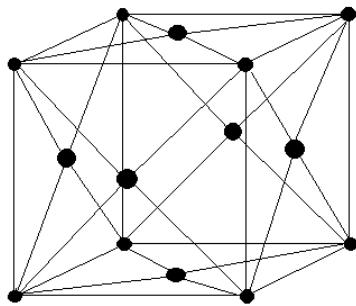
$$\mathbf{a}_1 = a\hat{x}; \mathbf{a}_2 = a\hat{y}; \mathbf{a}_3 = a\hat{z}$$

där a är kubens sidolängd. Detta illustreras i figur 4.



Figur 4: Det kubiska Bravais-gittret, simple-cubic, med translationsvektorer.

Ytcentrerad (fcc): Detta Bravais-gitter har, förutom gitterpunkter i varje hörn, även gitterpunkter i mitten av varje sida på kuben, se figur 5.

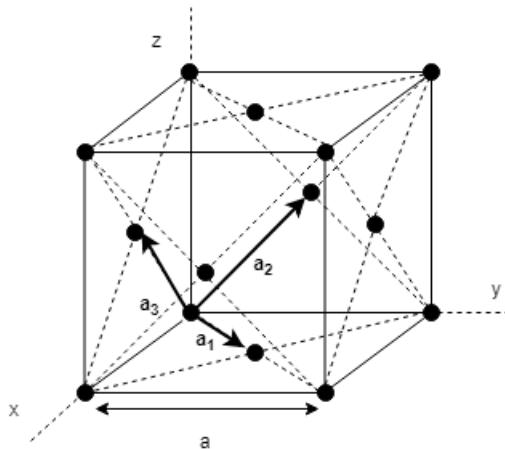


Figur 5: Det kubiska Bravais-gittret, face-centered-cubic.

Det ytcentrerade kubiska Bravais-gittret innehåller fyra gitterpunkter per cell. I kartesiska koordinater är ett möjligt val av translationsvektorerna för gittret följande:

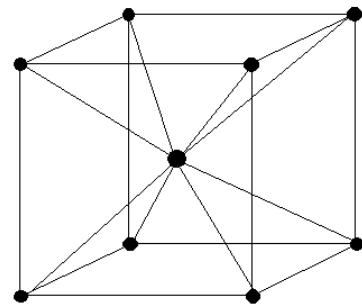
$$\mathbf{a}_1 = 0.5a(\hat{x} + \hat{y}); \mathbf{a}_2 = 0.5a(\hat{y} + \hat{z}); \mathbf{a}_3 = 0.5a(\hat{z} + \hat{x})$$

där a är kubens sidolängd [3]. Detta illustreras i figur 6.



Figur 6: Det kubiska Bravais-gittret, face-centered-cubic, med translationsvektorer.

Rymdcentrerad (bcc): Detta Bravais-gitter har, förutom gitterpunkter i varje hörn, även en gitterpunkt i mitten av kuben, se figur 7.

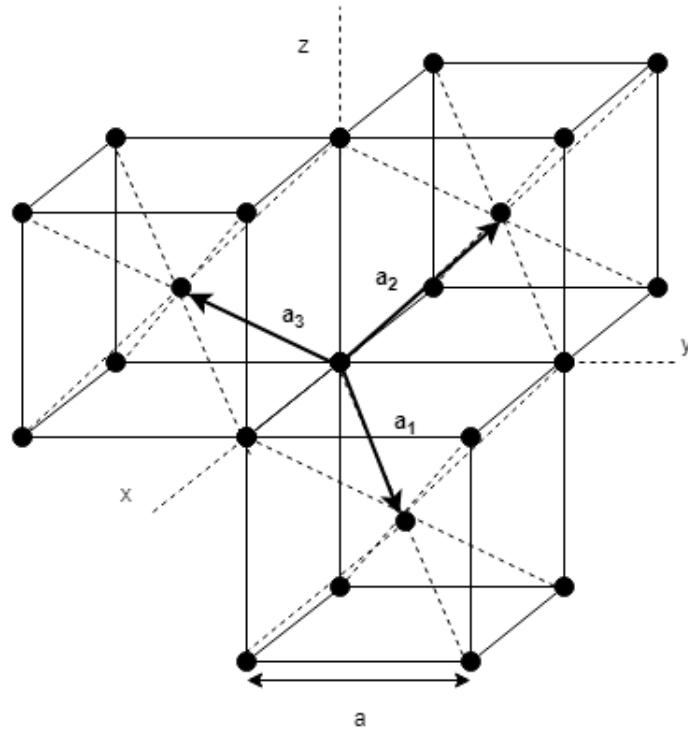


Figur 7: Det kubiska Bravais-gittret, body-centered-cubic.

Det rymdcenterade kubiska Bravais-gittret innehåller två gitterpunkter per cell. I kartesiska koordinater är ett möjligt val av translationsvektorer för gittret följande:

$$\mathbf{a}_1 = 0.5a(\hat{x} + \hat{y} - \hat{z}); \mathbf{a}_2 = 0.5a(-\hat{x} + \hat{y} + \hat{z}); \mathbf{a}_3 = 0.5a(\hat{x} - \hat{y} + \hat{z})$$

där a är kubens sidolängd.[3]. Detta illustreras i Figur 8.



Figur 8: Det kubiska Bravais-gittret, body-centered-cubic, med translationsvektorer.

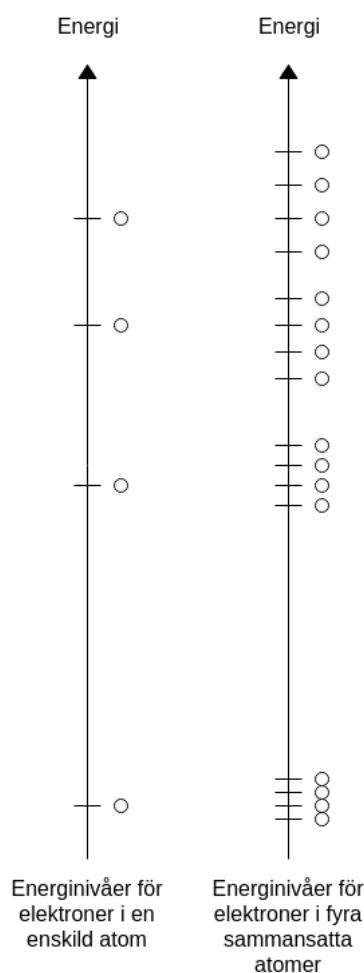
3.3 Fermi-energi

Fermi-energin är en speciell energi för ett flerelektronsystem och kan definieras på flera sätt. Den mest relevanta definitionen för numeriska beräkningar är denna: Fermi-energin är en energi där antalet enelektron tillstånd med energi lägre än Fermi-energin vid temperaturen 0 K är lika med det totala antalet elektroner i systemet [5, s. 141].

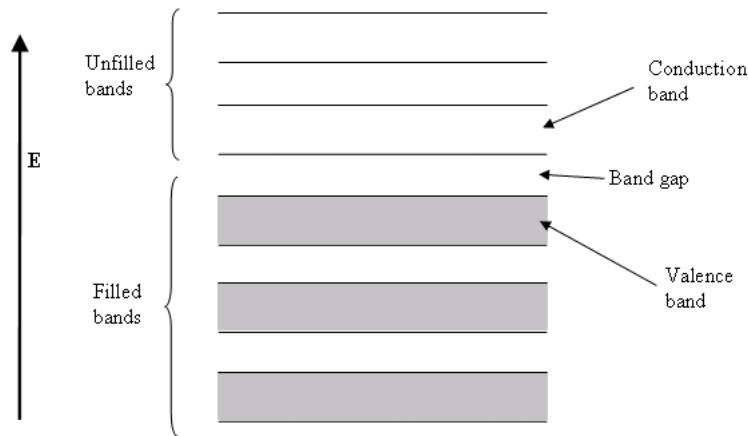
3.4 Bandstruktur

Elektroner i en atom ordnas enligt Pauliprincipen - inga två elektroner kan dela samma värden på alla kvanttal (inklusive spinn). När atomer sätts ihop till molekyler eller kristaller gäller fortfarande principen, så en förändring i besättning fås då elektroner besätter samma orbital [3, s. 56-57]. Detta gör att energin av elektronerna ändras något, se figur 9. Dessa energiändringar är mycket små, vilket orsakar att elektronernas energinivåer i en kristall packas mycket tätt i band i en kristallstruktur, se figur 10 [6, s. 178].

I en kristall beror elektronernas energi av vågtalet, \mathbf{k} . Se kapitel 4.2 för hur detta används.



Figur 9: Illustration av hur elektroners energinivåer i en molekyl eller kristall samlas tätt kring energinivåerna hos de enskilda atomerna som utgör molekylen eller kristallen.



Figur 10: Bandstruktur i en halvledare. Figur av Starling T. [7]

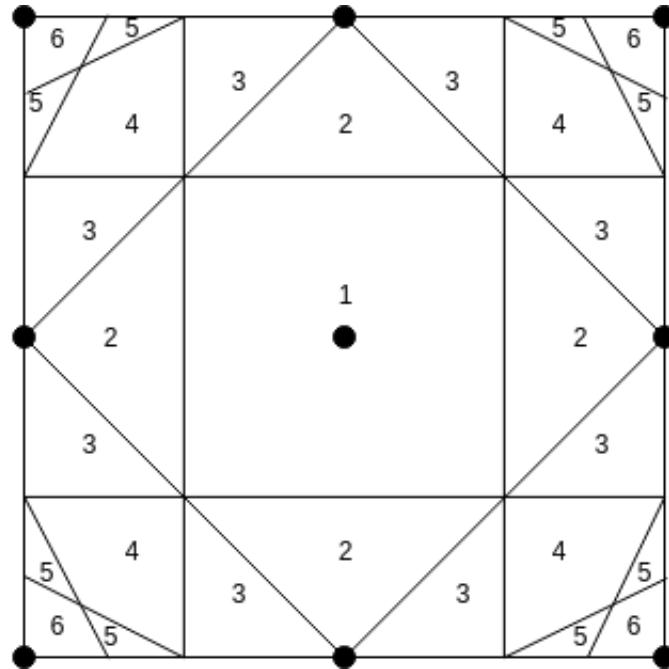
3.5 Fermi-yta

Om ett antal band inte är helt fylda finns det för varje band en yta i reciproka rummet som separerar fylda elektronenergitillstånd från ofylda. Mängden av alla sådana ytor i en kristall är känd som kristallens Fermi-yta, medan delarna av Fermi-ytan som kommer från de individuella banden kallas för grenar till Fermi-ytan [5, s. 142]. Fermiytan är alltså i det reciproka rummet en sammansättning av ytor med konstant energi, en isoyta.

3.6 Brillouin-zon

Den 1:a Brillouin-zonen är Wigner-Seitz-cellen av det reciproka gittret, det vill säga mängden av punkter i k-rummet som ligger närmre origo än någon annan reciprok gitterpunkt [5, s. 73]. Ett Bragg-plan är ett plan som vinkelrätt bisektrrar vektorerna mellan origo och andra reciproka gitterpunkter [5, s. 99], så en ekvivalent definition är denna: Den 1:a Brillouin-zonen är mängden av punkter i k-space som kan nås från origo utan att korsa något Bragg-plan. På samma sätt fortsätter man: Den 2:a Brillouin-zonen är mängder av punkter i k-space som nås när exakt ett Bragg-plan korsas (utan att korsa tillbaka till en redan passerad zon), och så vidare. Se figur 11 för en illustration av detta.

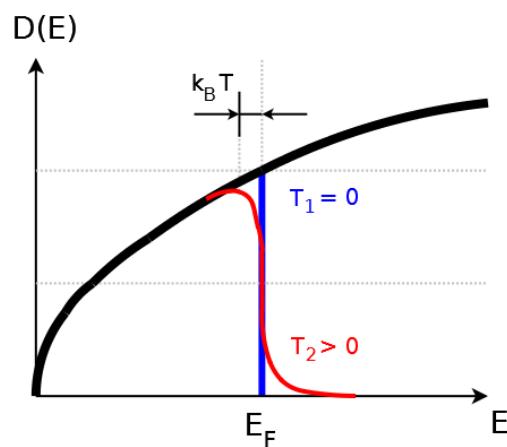
Alla Brillouin-zoner är primitiva celler till det reciproka gittret [5, s. 163]. Eftersom Brillouin-zonerna beror av geometrin hos det gitter som finns så ser samtliga Brillouin-zoner olika ut för olika kristallstrukturer.



Figur 11: Illustration av definitionen av Brillouin-zoner för ett tvådimensionellt Bravais-gitter. Figuren visar gitterpunkter i form av prickar och Bragg-plan som linjer. Siffrorna indikerar de olika Brillouin-zonerna (1:a, 2:a, och så vidare). Figur inspirerad av figur i Solid State Physics av Ashcroft, N. och Mermin, N. [5]

3.7 Tillståndstäthet

Tillståndstätheten beskriver hur många tillstånd som finns för olika energier. I Figur 12 beskrivs tillståndstätheten som funktion av energin i den så kallade Fermi-Dirac-fördelningen [8, s. 374]. Den blå och den röda linjen visar skillnaden i hur tillstånd är ockuperade beroende på temperaturen. När temperaturen är noll Kelvin är inga tillstånd med energi över Fermi-energin ockuperade men ökas temperaturen så kommer vissa elektroner genom termisk excitation kunna ha energier över Fermi-energin.



Figur 12: Teoretisk tillståndstäthet som funktion av energin. Figur av Feidlimid H. [9]

4 Numeriska metoder för beräkning av Fermi-ytor (utkast)

För att beräkna Fermi-ytan hos ett material behövs två saker:

- Förhållandet mellan elektroners energi och vågtal \mathbf{k} .
- Fermi-energin.

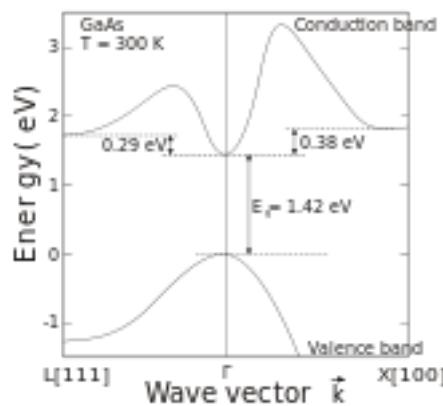
Den första fås genom att gå från Schrödingerekvationen till energiekvationer för de olika banden i bandstrukturen via täthetsfunktionalteori (eng. density functional theory, DFT), se kap. 4.1. Den andra fås genom att börja på samma sätt och sedan sätta in ett antal k-punkter i energiekvationerna för att få ut bandstrukturen (se kap. 4.2), beräkna tillståndstätheten (se kap. 4.3), och till slut integrera tillståndstätheten från negativa oändligheten till Fermi-energin, sätta integralen lika med antalet elektroner i systemet, och sedan lösa ut Fermi-energin.

4.1 Approximativ lösning av Schrödingerekvationen

Schrödingerekvationen som beskriver en partikels position och energi är omöjlig att lösa exakt för komplicerade system. För komplicerade system används därför metoden DFT för att gå från Schrödingerekvationen till energiekvationerna för de olika banden i bandstrukturen.

4.2 Bandstrukturberäkning

Från energiekvationerna för varje band fås energin för k-punkter belägna längs en slinga som täcker viktiga symmetripunkter i 1:a Brillouin-zonen, vilket ger ett förhållande mellan vågtal och energi längs denna slinga. Figur 13 beskriver bandstrukturen för Galliumarsenid (GaAs) som är en halvledare och därmed en icke-metall. Icke-metaller har ett bandgap mellan valensband och ledningsband och det är i detta gap som Fermi-energin ligger.

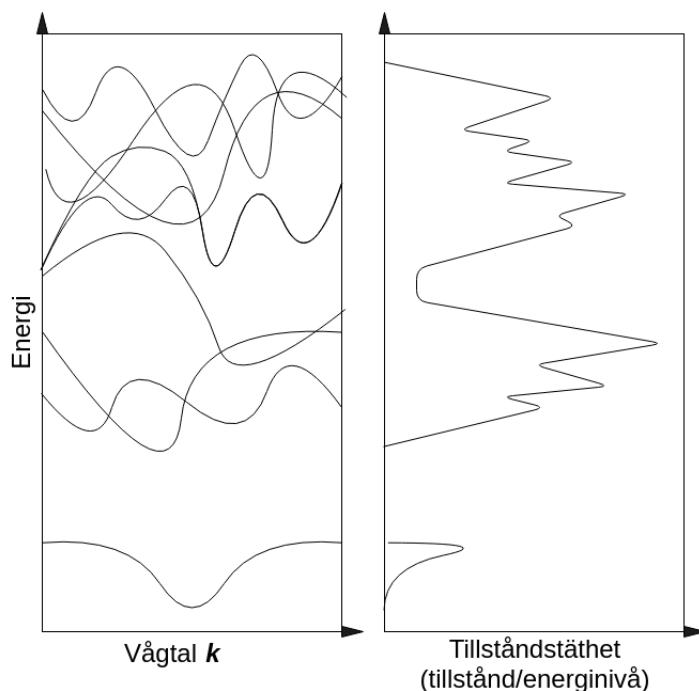


Figur 13: Bandstrukturen för GaAs. Figur av Cepheiden [10].

Att Fermi-energin ligger mitt i bandgapet innebär att inga k-punkter har den energin och en Fermi-uta kan inte fås ut. I metaller kommer ledningsbandet och valensbandet att överlappa och Fermi-energin kommer vara någonstans i överlappet vilket alltså kommer ge en Fermi-uta.

4.3 Tillståndstäthetsberäkning

Tillståndstätheten kan beräknas utifrån bandstrukturen. Detta genom att i bandstrukturplotten gå uppåt längs energi-axeln ett litet steg i taget och räkna antalet gånger som något band korsas vid varje energinivå, det vill säga hur många tillstånd som finns för den energin. I Figur 14 illustreras förhållandet mellan bandstruktur och tillståndstäthet (det är ett påhittat exempel och alltså inte en illustration av ett faktiskt ämnes bandstruktur och tillståndstäthet). I denna bild syns tydligt att i ett bandgap korsas inga band och alltså är tillståndstätheten noll vid dessa energier. I praktiken utgår man inte från en bandstrukturplot, utan istället från den tredimensionella "bandstrukturen" som fås då man beräknar energierna för ett stort antal k -punkter utspridda i den 1:a Brillouin-zonen.

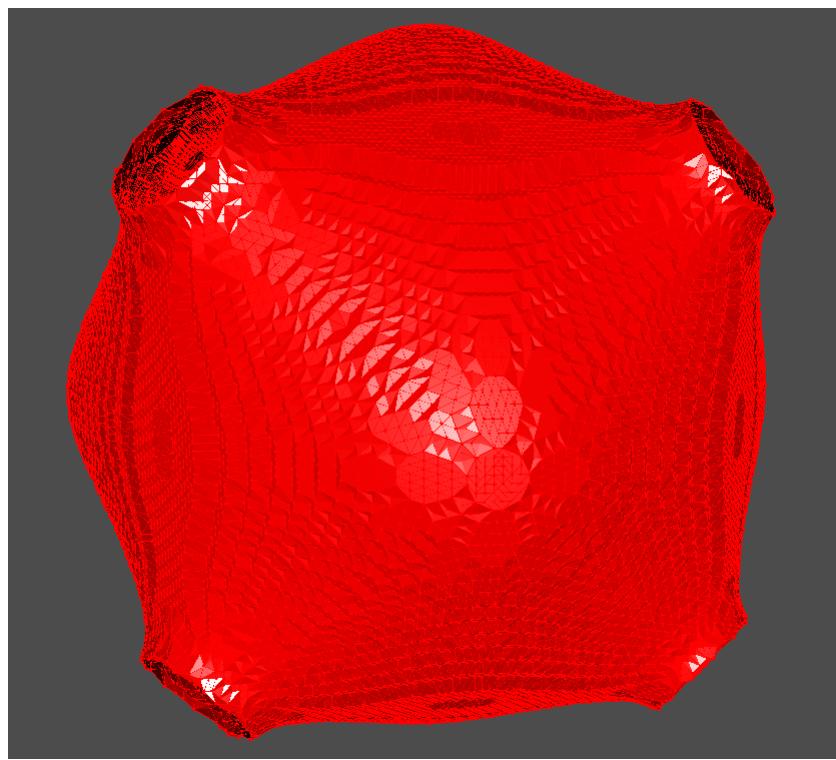


Figur 14: Förhållandet mellan bandstruktur och tillståndstäthet.

Om tillståndstätheten integreras från negativa oändligheten till Fermi-energin så fårs antalet elektroner i systemet, se definitionen i kap. 3.3. Antalet elektroner i systemet är känt vilket gör att Fermi-energin kan lösas ut.

4.4 Fermi-yteberäkning

När energiekvationerna för varje band har tagits fram och Fermi-energin har lösats ut kan Fermi-ytan beräknas i form av en energiisoya i det reciproka rummet med energi lika med den framtagna Fermi-energin (se kapitel 3.5). Figur 15 beskriver Fermi-ytan för silver.



Figur 15: Fermi-yta för silver. Figur av Marmodoro A. [11]

Referenser

- [1] Hoddeson L. m. fl. *Out of the Crystal Maze: Chapters from the History of Solid-State Physics*. New York: Oxford University Press, 1992, s. 3–5.
- [2] Stephen Dugdale. “Life on the edge: a beginner’s guide to the Fermi surface”. I: *Physica Scripta* 2016. 91 (53009), s. 1–3.
- [3] Kittel C. *Introduction to Solid State Physics*. 8th edition. New Jersey: John Wiley & Sons, 2005.
- [4] Shtilman A. 4sizetile [bild]. 2008 [citerad 2018 April 30].
Hämtad från <https://upload.wikimedia.org/wikipedia/commons/3/35/4sizetile.jpg>. Public domain, se <https://commons.wikimedia.org/wiki/File:4sizetile.jpg>.
- [5] Neil Ashcroft och David Mermin. *Solid State Physics*. 1st edition. California: Cengage Learning, 1976.
- [6] Sharon Ann Holgate. *Understanding Solid State Physics*. 1st edition. Boca Raton, Florida: CRC Press, Taylor & Francis Group, 2009, s. 178.
- [7] Starling T. Semiconductor band structure (lots of bands). [citerad 2018 April 30]
Hämtad från https://upload.wikimedia.org/wikipedia/commons/a/a3/Semiconductor_band_structure_%28BY-SA_3.0%29.png ([CC BY-SA 3.0](https://creativecommons.org/licenses/by-sa/3.0/deed.en)) <https://creativecommons.org/licenses/by-sa/3.0/deed.en>.
- [8] Harris R. *Modern Physics*. 2nd edition. San Francisco: Pearson, 2008.
- [9] Feidlimid H. Electron gas Density of states sketched. 2014 [citerad 2018 April 30].
Hämtad från https://upload.wikimedia.org/wikipedia/commons/c/c1/Electron_gas_Density_of_states_sketched.png. ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/)) <https://creativecommons.org/licenses/by-sa/4.0/deed.en>.
- [10] Cepheiden (Wikipedia-användare). Bandstruktur GaAs. 2009 [citerad 2018 April 30].
Hämtad från https://upload.wikimedia.org/wikipedia/commons/6/67/Bandstruktur_GaAs_en.svg. ([CC BY-SA 3.0](https://creativecommons.org/licenses/by-sa/3.0/)) <https://creativecommons.org/licenses/by-sa/3.0/deed.en>.
- [11] Marmodoro A. Bulk Ag, Fermi surface. 2012 [citerad 2018 April 30]
Hämtad från https://upload.wikimedia.org/wikipedia/commons/b/b2/Bulk_Ag%2C_Fermi_surface.png. ([CC BY-SA 3.0](https://creativecommons.org/licenses/by-sa/3.0/)) <https://creativecommons.org/licenses/by-sa/3.0/deed.en>.

D Fördjupningsarbete - Visualisering av volymsdata

Visualisering av elektronstrukturer

Visualisering av volymsdata

Andreas Kempe och Viktor Bernholtz

Version 0.4

Status

Granskad	VB	2018-05-25
Godkänd		

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	vi
1 Inledning	1
1.1 Bakgrund	1
1.2 Syfte	1
1.3 Frågeställning	1
1.4 Avgränsningar	1
1.5 Definitioner	1
2 Volymsrendering	2
2.1 Renderingspipeline	4
2.2 Pathtracing	5
2.2.1 Matematisk beskrivning	5
2.2.2 Homogena koordinater	6
2.2.3 Ortografisk projektion	6
2.2.4 Perspektivisk projektion	7
2.2.5 Model-view-matris	8
2.2.6 Invers projektion	8
3 Marching cubes	9
3.1 Förbättringsmöjligheter	11
3.1.1 Mindre kuber	11
3.1.2 Interpolation av kantlinjer	12
4 Prestandamätning	14
4.1 Metod	14
4.1.1 Inviwonätverk	15
4.2 Resultat	16
4.2.1 Volymsrendering	16
4.2.2 Marching tetrahedon	17
4.3 Diskussion	18
Referenser	20

Figurer

1	Grov skiss över 2D-stack och 3D-objekt	3
2	Lådan är en voxel med ett enda värde	3
3	Lådan är en voxel med fler värden knutna till sig	3
4	Renderingspipeline för volymsrendering. [13, s. 29]	4
5	Illustration av en ortografisk projektionskub. [13, s. 55]	6
6	Illustration av en perspektivisk projektionspyramid. [13, s. 53]	7
7	Indexering av hörn och kantlinjer i en kub. Inspirerad av Paul Brouke [3].	9
8	Exempel där ett hörn är inuti objektet. Inspirerad av Paul Brouke [3].	10
9	Olika kombinationer av marching cubes, figur av Xu D och Zhang Y [6]. Figuren är lisensierad med creative commons (CC BY-NC-ND 4.0).	11
10	Exempel på konstruktion av ytor med stora och små kvadrater. Inspirerad av Ben Anderson [2]	12
11	Exempel på interpolation av kantlinje i y-led.	13
12	Exempel på konstruktion av ytor med interpolation och kombination av interpolation och små kvadrater. Inspirerad av Ben Anderson [2]	14
13	Inviwonätverk för prestandajämförelse.	15
14	Volymsrendering med en sampel per stråle.	16
15	Volymsrendering med 10 sampel per stråle.	17
16	Isoytor genererade med tre olika isovärden.	18

Tabeller

1	Mätdata från tidtagning av volymsrendering.	16
2	Mätdata från tidtagning av Marching tetrahedon.	17

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.4	2018-05-25	En del korrigeringar efter återkoppling från beställare.	AK, VB	VB
0.3	2018-05-14	En del korrigeringar efter återkoppling från handledare.	AK, VB	VB
0.2	2018-05-03	Korrigering av språk efter återkoppling från Tema.	AK, VB	AK, VB
0.1	2018-04-27		AK, VB	AK, VB

1 Inledning

1.1 Bakgrund

Inom många olika fält undersöks volymer och deras innehåll. Ett välkänt område är MR, magnetisk resonanstomografi, där kroppen utsätts för starka elektromagnetiska fält som gör att olika vävnader i kroppen kan läsas av [14]. Ett annat område är beräkningar gjorda med hjälp av kvantmekanik på olika kristallstrukturer. Mjukvaran VASP kan bland annat användas till att beräkna laddningstätheten i en kristall och resultatet fås som laddningstätheten för olika punkter i kristallens volym [1].

Det är svårt att bilda sig en uppfattning om vad en stor mängd siffror kopplade till koordinater betyder utan att åskådliggöra resultatet på ett lättfattligt sätt. Vid Linköpings universitet används en mjukvara vid namn Inviwo som utvecklas av forskargruppen *Scientific Visualization Group* vid Linköpings universitet i Norrköping [23]. Den klarar av att visualisera data på en mängd olika sätt, bland annat genom volymsrendering [12].

1.2 Syfte

Arbetet ämnar göra en undersökning av hur ett par metoder som används i mjukvaran Inviwo fungerar för att sedan göra en jämförelse av deras resultat och prestanda.

1.3 Frågeställning

Det här arbetet ämnar besvara förljande frågor: Hur fungerar volymsrendering? Hur fungerar ytextraktion med marching cubes? Hur skiljer sig prestandan och resultatet mellan de två metoderna?

1.4 Avgränsningar

Arbetet avgränsas till att undersöka volymsrendering genom strålföljning och extrahering av isoytor med hjälp av algoritmen *Marching Cubes*. Detta då dessa två metoder är aktuella för kandidatarbetet *Visualisering av elektronstrukturer* som rapporten skrivs för.

1.5 Definitioner

Gradient betecknas ∇f och är en vektor som har de partiella derivatorna till en funktion f som sina komponenter. Detta leder till att gradienten pekar i den riktning för vilken funktionen i fråga har störst ökning [16].

Implicit funktion definieras i nationalencyklopedin enligt följande: "implicit funktion, i matematiken en funktion som defineras indirekt genom en eller fler ekvationer som funktionsvärdena skall uppfylla." [17].

Interpolering är inom matematiken en metod att approximativt beräkna mellanliggande variabelvärden utgående från en funktion där bara vissa värden är kända [18].

Inviwo är en mjukvara för visualisering av data som utvecklas vid Linköpings universitet [23].

Isoyta är den yta som defineras av den implicita funktionen.

Isovärde är ett värde på isoytan.

Sampel se *Sampling*.

Sampling är en process där diskreta mätningar görs på en storhet och ett mätvärde, s.k. sampel, erhålls [19].

Skalär är en storhet med ett värde som saknar riktning [24].

Skalärfält är det fält som knyter samman skalärer med punkter i ett rum [22, s. 6].

Strålgång är den väg ljuset tar genom ett objekt.

Tetraeder är en geometrisk kropp som begränsas av fyra triangelformade plana ytor [20].

Uppslagningstabeller används för data som programmet ofta efterfrågar, i vårt fall matematiska tabeller. Dessa används för att det går fortare att slå upp ett värde än att räkna ut det [4].

Volumetrisk betyder att någonting kan betraktas i tre faktiska dimensioner. Det betyder att vi kan titta på någonting från alla håll eller rotera på det [5].

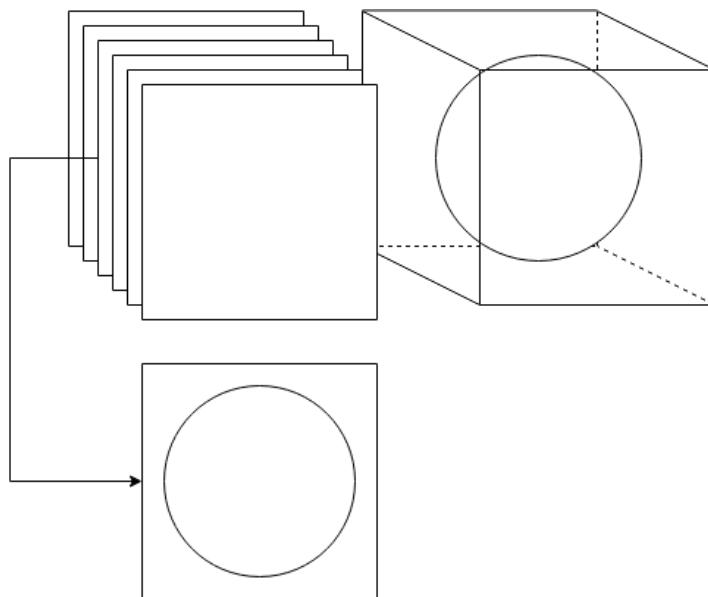
Voxel är ett individuellt volymelement, precis som pixel är ett individuellt bildelement [10].

Överföringsfunktion är ett sätt att representera ett linjärt tidsinvariant system med avseende på in-utförhållande [15].

2 Volymsrendering

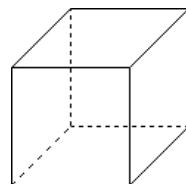
Volymsrendering är en teknik som tillåter användaren att se in i 3D-objekt. Volymsrendering kan göra delar av objekten transparenta och andra icke transparenta, det går ofta att bestämma nivå av transparens och färg [11].

Metoden genererar bilder av 3D-volymdata utan att explicit extrahera geometriska ytor från datan. Den här tekniken använder en optisk modell för att koppla data till optiska egenskaper så som färg och transparens. Under renderingen skapas de optiska egenskaperna längs varje strålgång för att sedan skapa en bild av datan. Även om datan här tolkas som en kontinuerlig funktion i rymden så representeras den, av praktiska skäl, som en uppsättning likformiga 3D-matriser. I grafikminnet sparas sedan volymdata i antingen en stack av 2D-bilder eller som ett enda 3D-objekt. I figur 1 ses en stack av 2D-bilder till vänster som är tvärsnitt av 3D-bilden till höger.

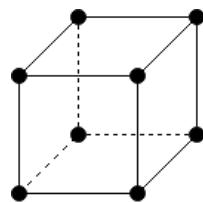


Figur 1: Grov skiss över 2D-stack och 3D-objekt

En 3D-bild byggs sedan upp av voxlar, där varje voxel har en specifik plats i rymden och har ett eller flera datavärden knutna till sig, se figur 2 respektive figur 3. Datavärden som ligger mellan två specifika platser i rymden erhålls med hjälp av interpolering med närmsta granne. Denna process kallas rekonstruktion och är en viktig del i volymsrendering.



Figur 2: Lådan är en voxel med ett enda värde



Figur 3: Lådan är en voxel med fler värden knutna till sig

Det huvudsakliga syftet med den optiska modellen är att beskriva hur partiklarna i en volym interagerar med ljus. I den vanligaste, lite enklare modellen antas volymen innehålla partiklar som samtidigt emitterar och absorberar ljus. De lite mer komplexa modellerna har lokala ljuskällor, volumetriska skuggor och tar hänsyn till hur ljuset sprids. De optiska egenskaperna anges antingen direkt av datavärden eller så beräknas de med en eller fler överföringsfunktioner. Överföringsfunktionerna har som mål att tydliggöra eller klassificera data som är av intresse för modellen. Då komplexa överföringsfunktioner tar tid att beräkna slås de ofta upp i uppslagningstabeller, medan de enklare funktionerna kan beräknas direkt.

Bilden skapas sedan genom att sampla volymen längs alla strålgångar och tillsätta de optiska egenskaperna. För den vanliga emissions-absorptionsmodellen som nämnts ovan används ekvation (1) för att beräkna färg och ekvation (2) för att beräkna transparens.

$$C = \sum_{i=1}^n C_i \prod_{j=1}^{i-1} (1 - A_j) \quad (1)$$

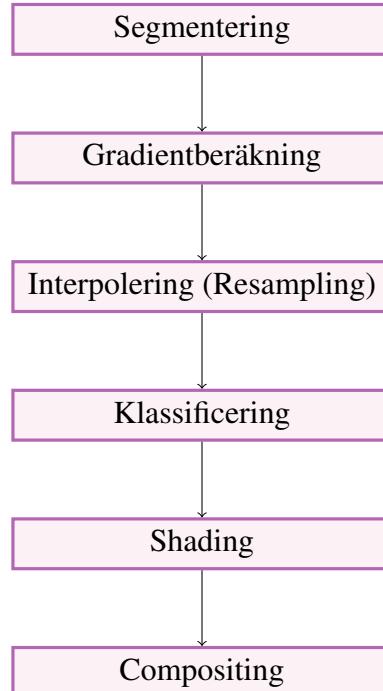
$$A = 1 - \prod_{j=1}^n (1 - A_j) \quad (2)$$

C_i och A_j är färgen respektive transparensen som överföringsfunktionen tilldelar datavärde för sampel i och j. Transparensen A_j approximerar absorptionen och den transparensviktade färgen C_i approximerar emissionen och absorptionen längs strålgången mellan sampel i och $i + 1$. C_i representerar alltså den mängd ljus som emitteras vid ett sampel i och hur mycket ljuset dämpas innan det når betraktaren. Både ekvation (1) och (2) löses effektivt genom att sortera alla sampel längs strålgången och iterativt beräkna både färgen och transparensen [10].

I avsnitt 2.1 nedan beskrivs de olika stegen i denna process.

2.1 Renderingspipeline

En volymsrendering har, som tidigare nämnts, som mål att ta en mängd data bestående av voxelar i 3D-rymden för att skapa en projektion av dessa till en 2D-bild. För att åstadkomma detta behöver ett antal problem lösas. I boken *Introduction To Volume rendering* beskrivs en renderingspipeline i sex steg, illustrerat av figur 4 [13, s. 29].



Figur 4: Renderingspipeline för volymsrendering. [13, s. 29]

I det första steget, segmentering, märks datapunkterna för att avgöra vad olika voxlar representerar i materialet. Detta då voxlarna i sig bara utgör delar av rymden. De kan till exempel representera en viss densitet, ett visst material eller vilken annan storhet eller egenskap som helst som går att associera till en punkt i rummet [13, s. 29-30].

Gradientberäkningssteget går ut på att hitta grader mellan de olika materialen. En gradient är en vektor som pekar i den riktning som har störst förändring och på så vis kan övergångar mellan olika material hittas. Denna information används senare i klassificerings- och shadingsteget [13, s. 30].

När strålar skickas genom volymen för att sampla dess data leder det vanligtvis till att strålen missar voxlar och passerar mellan dem. I interpoleringssteget lösas problemet genom att, för varje punkt, interpolera dess värde från de kringliggande voxlarnas värden [13, s. 30].

Stegen klassificering och shading färglägger respektive ljusätter punkterna i volymen. Klassificeringen utgår från de etiketter som tilldelats punkterna i segmenteringssteget och färglägger dem baserat på vilken typ de tilldelats, varefter shadingsteget gör beräkningar på ljussättning av desamma [13, s. 30]. Den matematiska beskrivningen hittas tidigare i avsnitt 2, ekvation 1 och ekvation 2.

Slutligen nås compositingsteget under vilket datan reduceras. Efter att ha interpolerat datavärden längs en stråle finns det i regel en väldigt stor mängd datapunkter som måste reduceras till en enskild punkt på 2D-ytan som volymen projiceras på. I detta steg går punkterna igenom och ett enskilt värde för varje pixel i bilden beräknas [13, s. 30]. En enkel metod för att bestämma färgen på pixeln är att översätta värdet för den voxel längs strålen som har störst värde till en färg, vilket kan ge en tydlig bild av isolerade strukturer [13, s. 49-50].

2.2 Pathtracing

I volymsrenderingssammanhang finns det en metod som heter pathtracing. Det är en strålföljningsmetod och den går ut på att strålar skickas ut från pixlarna, vilka bygger upp 2D-bilden, genom den volym som skall renderas. Längs med dessa strålar sampelas volymens voxlar och deras värden interpoleras för att sedan kopplas till färger i slutrenderingen. Denna process sker i stegen interpolering, klassificering, shading och compositing i renderingspipelinen som kan ses i figur 4 och beskrivs övergripande i avsnitt 2.1 [13, s. 58-59].

2.2.1 Matematisk beskrivning

En transformation är en process som tillåter att en uppsättning värden tas från ett rum till ett annat och kan då definieras matematiskt som $T : U \rightarrow V$. Där funktionen T transformrar punkter från rummet U till rummet V [21, s. 150-151].

Inom linjär algebra är det vanligt att uttrycka transformationer medelst matriser, vilket görs enligt uttrycket i (3) som är ett exempel på en transformation i tre dimensioner. Transformationer i högre antal dimensioner uttrycks lätt på samma sätt genom att lägga till fler rader och kolumner i matriserna [21, s. 151].

$$T_A(X) = AX = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} ax + by + cz \\ dx + ye + fy \\ gx + hy + iz \end{bmatrix} \quad (3)$$

I fallet med volymsrendering skall voxlarna, som tidigare nämnts, tas från volymsrummet i tre dimensioner till det tvådimensionella rum som den önskade bilden utgör. För att göra detta används en transformationsmatris som utför den önskade projektionen [13, s. 52].

2.2.2 Homogena koordinater

En rotation, skalning och translation kan i tre dimensioner representeras av en 4×4 -matris. För att åstadkomma detta läggs en fjärde koordinat till de tre existerande rumskoordinaterna och en punkt representeras således av koordinaterna (x, y, z, w) , vilket då kallas homogena koordinater. Punkten $(0, 0, 0, 0)$ är inte tillåten, utan åtminstone en homogen koordinat måste vara nollskild. Normalt sätts $w \neq 0$ och punkten (x, y, z, w) kan då uttryckas som $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}, 1)$, vilket kallas de kartesiska koordinaterna för den homogena punkten. Vanligtvis utförs divisionen med w om $w \neq 1$ [8, s. 204].

När en punkt nu representeras av de fyra homogena koordinaterna kan därför en translation av punkten $(x, y, z, 1)$ till $(x + d_1, y + d_2, z + d_3, 1)$ beskrivas med följande matrisekvation:

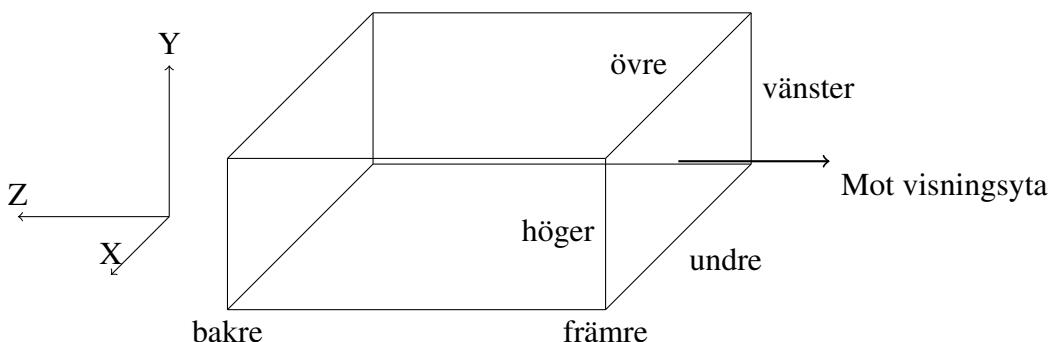
$$\begin{bmatrix} 1 & 0 & 0 & d_1 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + d_1 \\ y + d_2 \\ z + d_3 \\ 1 \end{bmatrix}$$

Värt att notera är att $w = 1$ även efter multiplikationen och den nya punkten således har rört sig inom samma tredimensionella volym i det fyrdimensionella rummet. Detta då varje w -koordinat ger ett nytt tredimensionellt rum. Trippeln $(x + d_1, y + d_2, z + d_3)$ har alltså utfört en enkel translation jämfört med (x, y, z) .

2.2.3 Ortografisk projektion

En ortografisk projektion är en projektion där strålarna sänds in parallellt över hela bildytan. Därmed formar volymen som strålarna passerar ett rätblock, se figur 5, och är en enklare projektion än perspektivistiska projektionen, se avsnitt 2.2.4, som tar hänsyn till betraktarens perspektiv när bilden projiceras [13, s. 54-55].

Som en följd av att strålarna skickas parallellt genom volymen, i form av ett rätblock, bevaras storleken för alla objekt i volymen för en ortografisk projektion [13, s. 54].



Figur 5: Illustration av en ortografisk projektionskub. [13, s. 55]

Den generella transformationsmatrisen för en ortografisk projektion ges av matrisen i ekvation (4), där beteckningarna hittas i figur 5, hämtad ur *Introduction To Volume rendering* [13, s. 54-55].

$$\begin{bmatrix} \frac{2}{\text{höger} - \text{vänster}} & 0 & 0 & t_x \\ 0 & \frac{2}{\text{övre} - \text{undre}} & 0 & t_y \\ 0 & 0 & \frac{-2}{\text{bakre} - \text{främre}} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$t_x = \frac{\text{höger} + \text{vänster}}{\text{höger} - \text{vänster}}$$

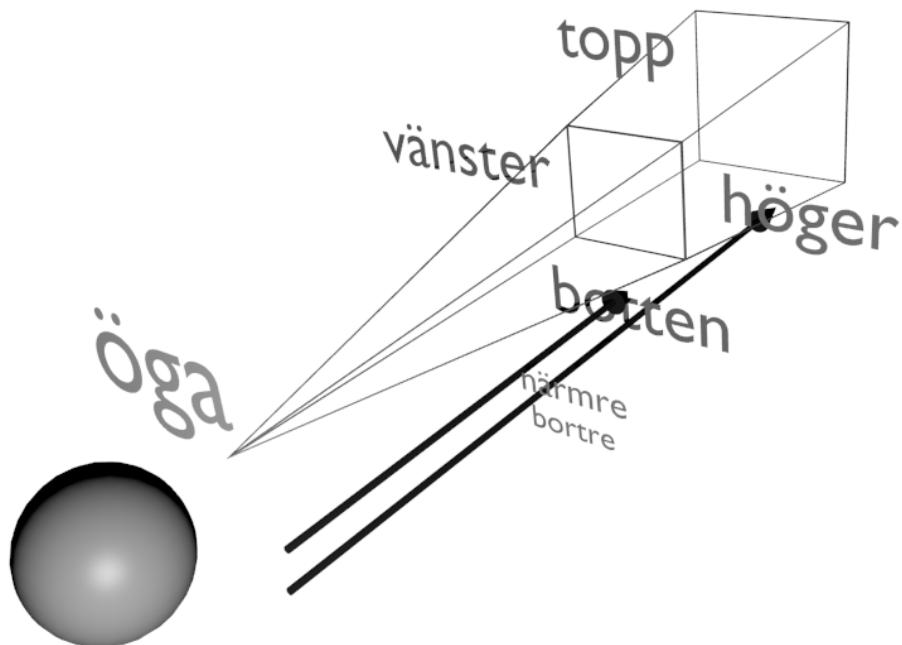
$$t_y = \frac{\text{övre} + \text{undre}}{\text{övre} - \text{undre}}$$

$$t_z = -\frac{\text{bakre} + \text{främre}}{\text{bakre} - \text{främre}}$$

2.2.4 Perspektivisk projektion

Den perspektiviska projektionen är en projektion som tar hänsyn till betraktarens perspektiv och leder till att objekt i förgrunden ser större ut än objekt som befinner sig längre bort [13, s. 52].

För att åstadkomma en perspektivisk projektion skickas strålarna genom volymen i form av en pyramid med avhuggen topp enligt figur 6.



Figur 6: Illustration av en perspektivisk projekionspyramide. [13, s. 53]

Även den perspektiviska projektionen uttrycks som en matris och har formen som hittas i ekvation (5).

$$\begin{bmatrix} \frac{2 \times närmre}{höger - vänster} & 0 & A & 0 \\ 0 & \frac{2 \times närmre}{topp - bottan} & B & 0 \\ 0 & 0 & C & D \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad (5)$$

$$\begin{aligned} A &= \frac{höger + vänster}{höger - vänster} \\ B &= \frac{topp + bottan}{topp - bottan} \\ C &= -\frac{bortre + närmre}{bortre - närmre} \\ D &= -\frac{2 \times bortre \times närmre}{bortre - närmre} \end{aligned}$$

2.2.5 Model-view-matris

För att kunna utföra en projektion räcker det inte med en projekionsmatris, utan det krävs även en beskrivning av från vilket håll volymen betraktas. Denna beskrivning görs med en så kallad model-view-matris, vilken multipliceras med projekionsmatrisen för att få den fullständiga transformationen. Model-view-matrissen är en 4x4-matris som innehåller ett antal konstanter [13, s. 55-56]. Då model-view-matrissen enbart flyttar betraktningsvinkeln kommer den inte beskrivas närmare här.

2.2.6 Invers projektion

När transformationsmatriser som beskriver den önskade renderingen av bilden erhållits återstår problemet att beräkna transformen av den faktiska datan.

Pathtracing drar nytta av att transformationsmatriserna är inverterbara. Genom att invertera matrisen får en transformation som översätter varje punkt på bildens 2D-yta till en punkt i 3D-rummet. På detta sätt behöver inte varje punkt i rummet gås igenom, utan det är möjligt att utgå från varje pixel i bilden och endast variera djupet, vilket motsvarar att skicka strålar in genom volymen [13, s. 57-59].

Inversen av model-view-matrissen kombinerat med projektionen beskrivs enligt ekvation (6), där varje element är namngivet med I på slutet för att indikera att det är den inversa matrisen. Notera således att t.ex. både $a = aI$ och $a \neq aI$ kan vara sant för olika inverser [13, s. 60].

$$\text{invers} \left(\begin{bmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ 0 & 0 & 0 & 1 \end{bmatrix} \right) = \begin{bmatrix} aI & eI & iI & mI \\ bI & fI & jI & nI \\ cI & gI & kI & oI \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

Genom att ta två punkter på bildens 2D-yta, (x'', y'') , och välja något z'' med $w = 1$, fås, genom att multiplicera med inversa transformen, homogena koordinater $(x', y', z', 1)$. Där $(x, y, z) = (x', y', z')$ är en punkt i volymen som önskas projiceras, se ekvation (7) [13, s. 60].

$$\begin{bmatrix} aI & eI & iI & mI \\ bI & fI & jI & nI \\ cI & gI & kI & oI \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x'' \\ y'' \\ z'' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (7)$$

Genom att att variera z'' kan nu olika punkter i volymen samplas. Bilden byggs upp genom att variera värdet på z'' för alla (x'', y'') på bildens 2D-yta [13, s. 60].

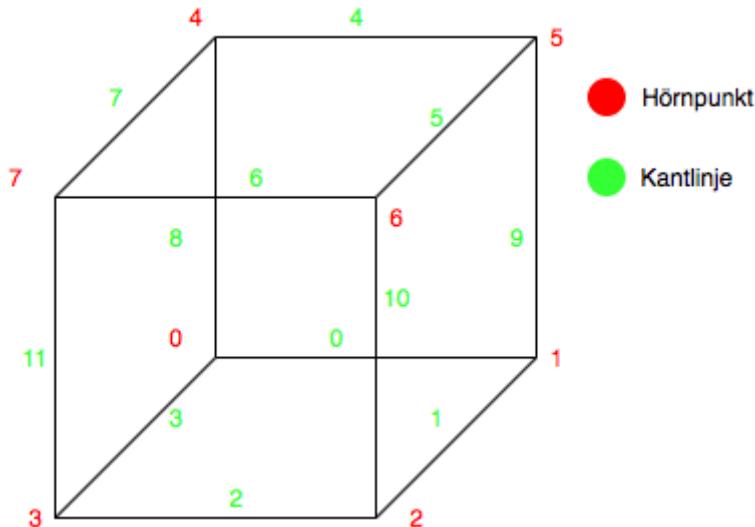
3 Marching cubes

Marching cubes är en algoritm för att skapa en isoyta utifrån volymetrisk data. Marching cubes kan appliceras inom ett flertal områden men de vanligaste är medicin och matematik. Inom medicin används det till exempel för att återskapa 3D-ytor av organ eller liknande och inom matematik för att skapa 3D-konturer av skalärfält [3].

Marching cubes kan definieras enligt följande: Givet ett objekt är det ett test för att avgöra om en godtycklig punkt ligger inom objektet samt inom vilka gränser objektet existerar. Dela in objektet i ett godtyckligt antal kuber. Testa om hörnen på varje kub är inuti objektet. För varje kub, där några hörn är inuti och några är utanför objektet, måste ytan passera genom kuben. Objektet skär alltså kubens kanter mellan hörnen av motsatt klassificering. Rita en yta i varje kub som förbinder dessa skärningar. Objektet är färdigt [2].

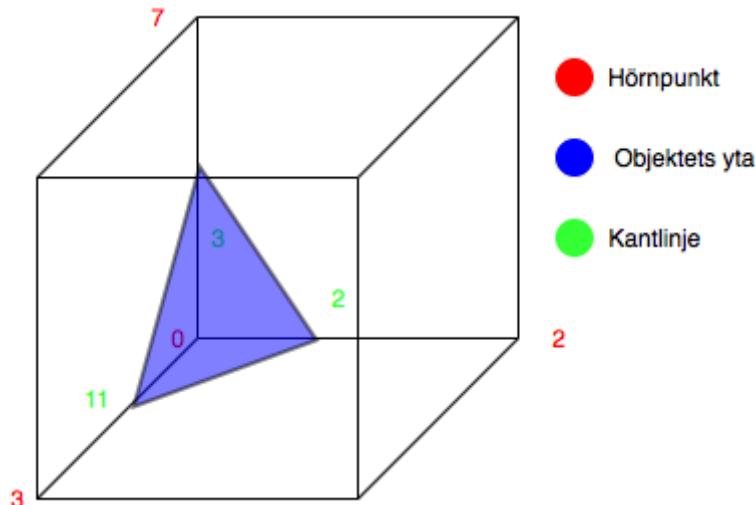
Marching cubes är alltså en algoritm för att skapa ett triangelnät från en implicit funktion. Det fungerar genom att iterera (engelska marching) över ett enhetligt kubiskt rutnät. Alla åtta hörnpunkter i kuben har ett binärt värde. Är alla hörnpunkter antingen ett eller noll kommer således ingen triangel att skapas, då hela kuben är under respektive över ytan. Är detta inte fallet kommer en eller flera trianglar blidas i kuben. Dessa trianglar kopplas sedan samman och bildar ytobjekten. I figur 7 kan vi se index för alla de åtta hörn algoritmen läser av värden på [7].

Figur 7 nedan, visar även kantlinjerna för en kub, det är alltså ett antal av dessa som skärs när en triangel skapas [3].



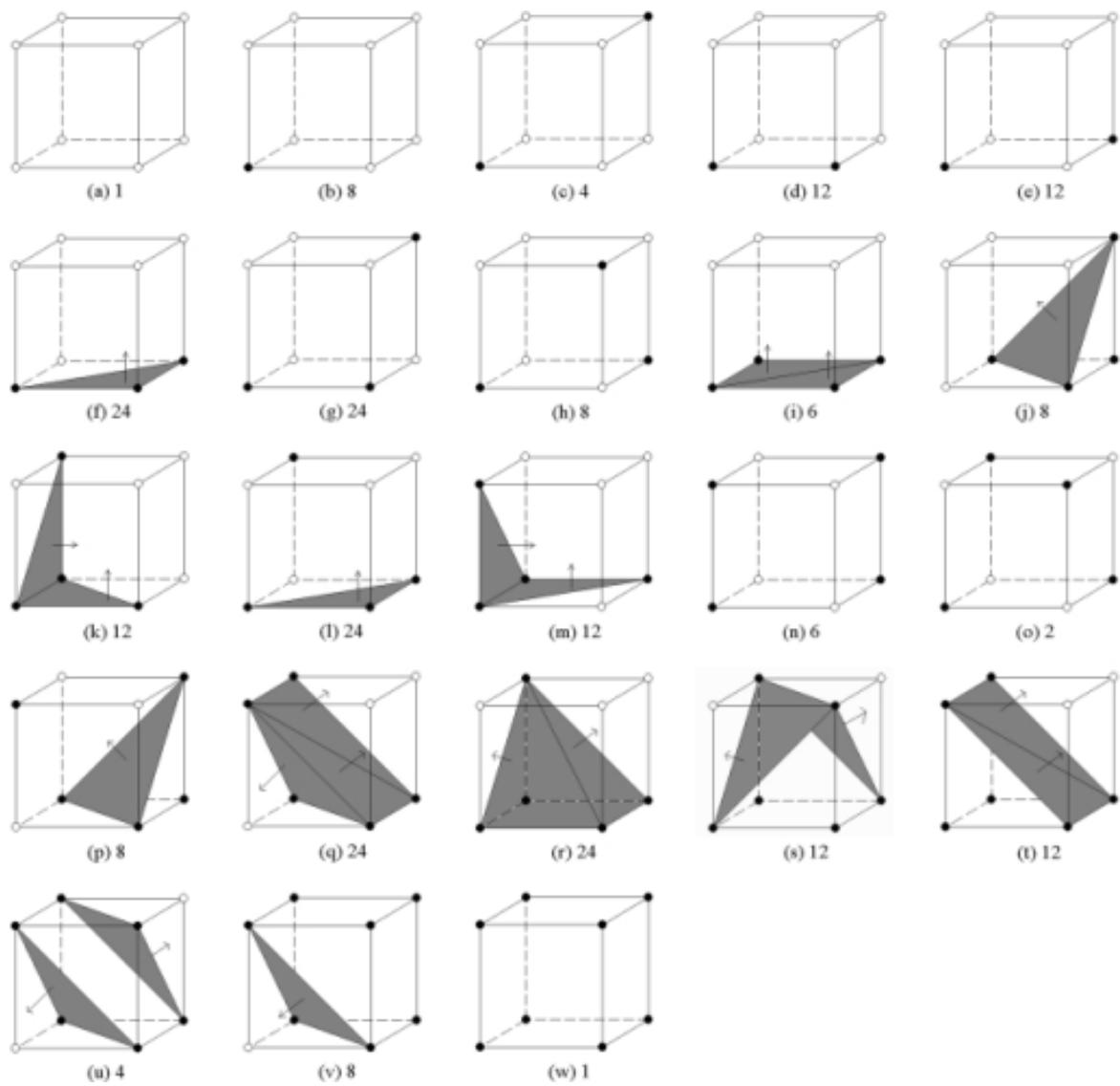
Figur 7: Indexering av hörn och kantlinjer i en kub. Inspirerad av Paul Brouke [3].

Ett exempel då en triangel skapas. Ett hörn är inuti objektet har värdet ett och om hörnet är utanför objektet har det värdet noll. Om alla hörn förutom hörn 0, har värdet noll kommer en triangel bildas enligt figur 8. I ett så här enkelt fall med har binära hörnpunkter kommer kantlinjerna, i detta exempel 2, 3 och 11 att skäras på mitten.



Figur 8: Exempel där ett hörn är inuti objektet. Inspirerad av Paul Brouke [3].

Då alla hörn kan vara både i objektet eller utanför objektet har vi $2^8 = 256$ kombinationer av trianglar i kuben. Många av dessa är ekvivalenta på grund av spegling och rotation så det finns det ett antal unika kombinationer som kan ses i figur 9 [3].



Figur 9: Olika kombinationer av marching cubes, figur av Xu D och Zhang Y [6]. Figuren är lisensierad med creative commons (CC BY-NC-ND 4.0).

3.1 Förbättringsmöjligheter

Det finns ett antal olika sätt att förbättra resultatet för marching cubes. Nedan redogörs det för två enkla men effektiva metoder och även när de kombineras.

3.1.1 Mindre kuber

Ett enkelt sätt att göra hyfsade förbättringar till en viss nivå är att göra kuberna mindre [2].

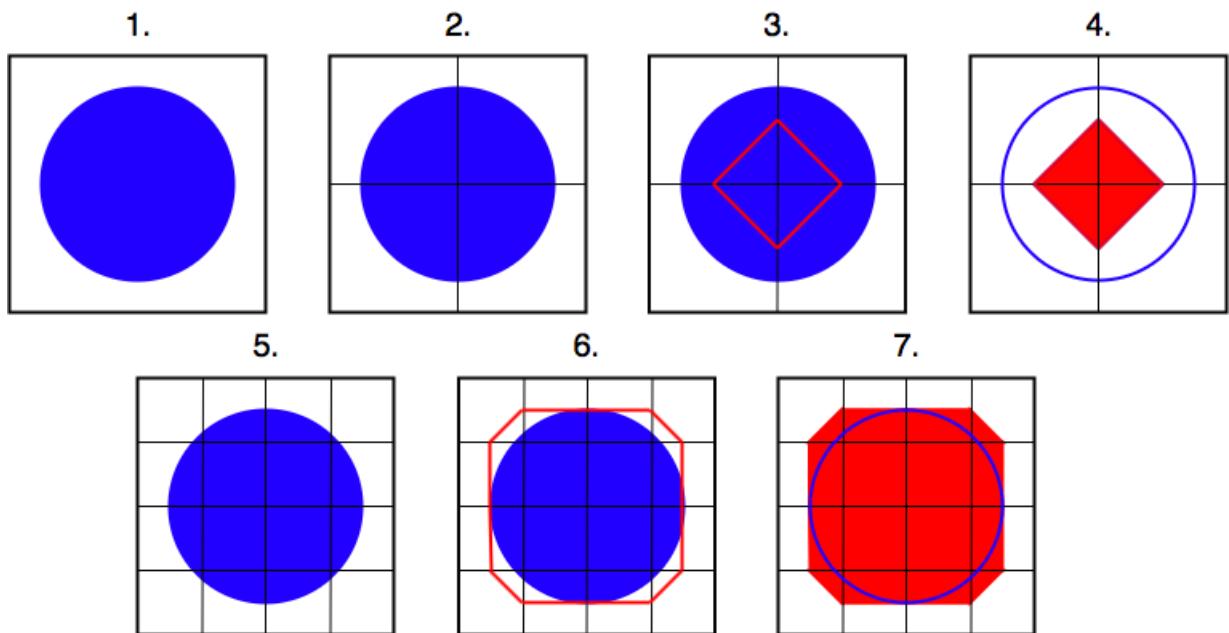
I figur 10 ser vi ett exempel på hur resultatet för bättras då kuberna minskar. I figuren är bilderna i 2D för enkelhetens skull men principen är den samma i 3D.

Nedan ges en kortfattad förklaring av stegen som utförs i figur 10.

1. Objektet och en yttre ram.

2. Dela in ramen i kvadrater.
3. Kontrollera vilket/vilka hörn som ligger i objektet och skapa en triangel för hörnet/hörnen.
4. Lägg ihop alla trianglar för att representera en yta för objektet.
5. Här delas ramen in i fler kvadrater.
6. Kontrollera vilket/vilka hörn som ligger i objektet och skapa en triangel för hörnet/hörnen.
7. Lägg ihop alla trianglar för att representera en yta för objektet.

I punkt 4 representeras ett ytobjekt men inte särskilt bra, i punkt 7 börjar det närligare sig ett mer likt ytobjekt. Hade kvadraterna gjorts mindre hade såklart ytobjektet blivit mer likt objektet men som nämnts ovan fungerar det bara att minska rutnätet till en viss nivå sedan blir kostnaderna för beräkningarna för stora.



Figur 10: Exempel på konstruktion av ytor med stora och små kvadrater. Inspirerad av Ben Anderson [2]

3.1.2 Interpolation av kantlinjer

Interpolation är ytligare ett sätt att förbättra marching cubes algoritmen. Det hela bygger på att istället för att skära kantlinjen på mitten interpolerar man fram var kantlinjen skärs. Detta görs med hjälp av ekvation 8.

$$P = P_1 + \frac{(isovalue - V_1)(P_2 - P_1)}{V_2 - V_1} \quad (8)$$

Där P_1 och P_2 är hörnpunkterna för den kantlinje som skärs och V_1 och V_2 är de skalära värdena på respektive hörnpunkt. P är punkten där kantlinjen skärs och $isovalue$ är isovärdet.

Åter till exemplet från avsnitt 3, figur 8. Nu interpoleras det fram var kantlinjerna skulle skäras om hörnen 0, 2, 3 och 7 skulle tilldelas slumpmässiga värden för respektive V och ett värde för *isovalue* mellan dessa V .

Hörn **0** = 5

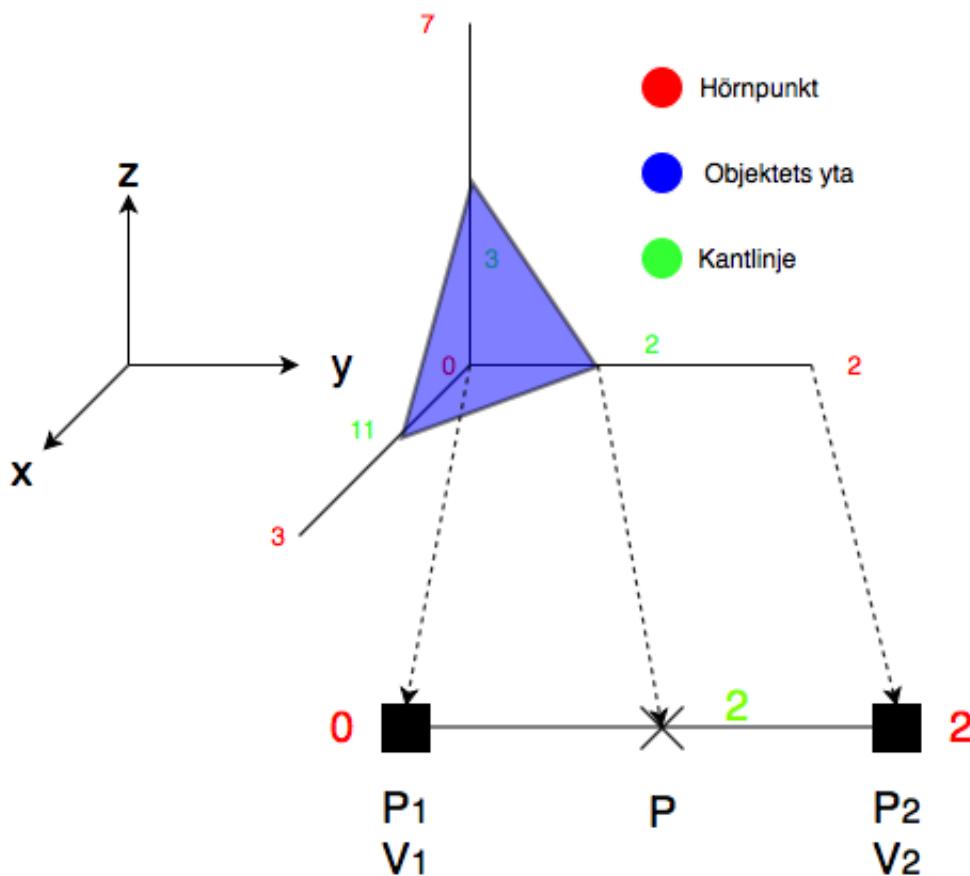
Hörn **2** = 3

Hörn **3** = 4

Hörn **7** = 1

Isovärdet *isovalue* = 2

I figur 11 ges ett förtysligande av beräkningen av P i y-led.

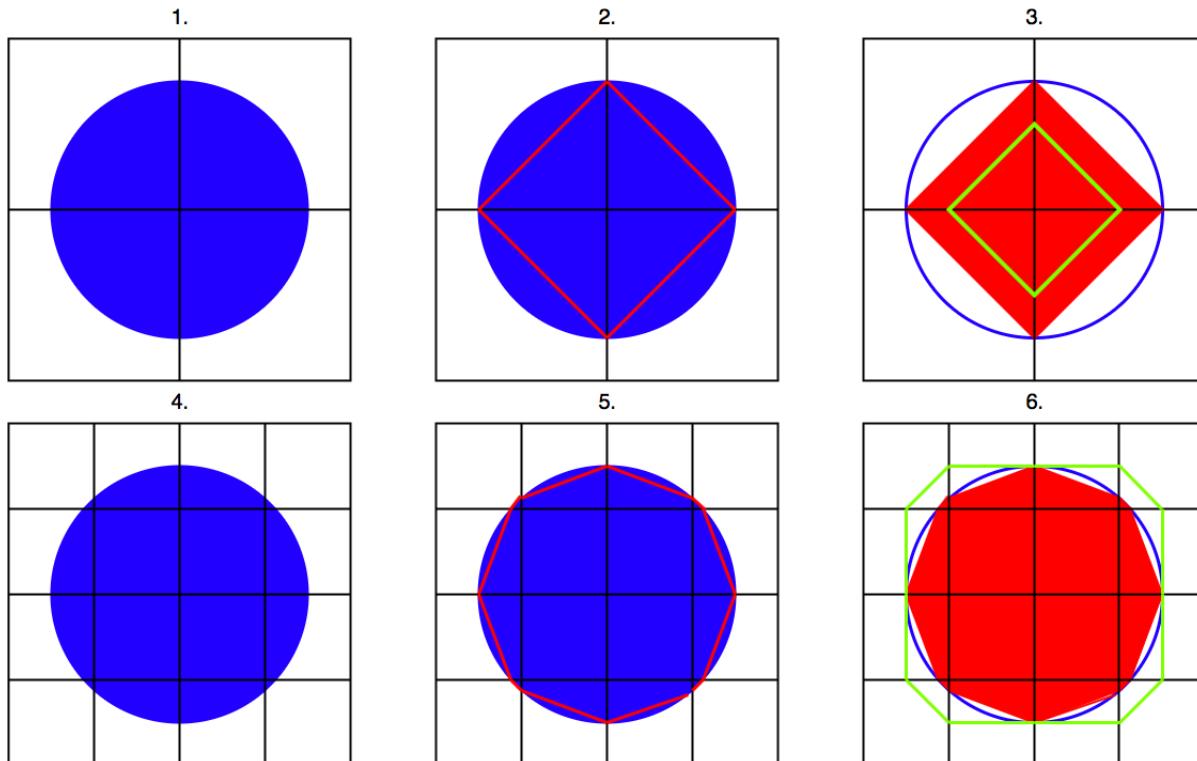


Figur 11: Exempel på interpolation av kantlinje i y-led.

- $P_1 = (0, 0, 0)$
- $P_2 = (0, 1, 0)$
- $V_1 = 5$
- $V_2 = 3$
- *isovalue* = 2

Sätts värdena ovan in i ekvation 8 skärs kantlinje 2 vid $P = (0, \frac{3}{2}, 0)$. På samma sätt beräknas det fram att kantlinje 3 skärs vid $P = (0, 0, \frac{4}{3})$ och kantlinje 11 vid $P = (3, 0, 0)$. Triangeln i figur 11 ska nu med hjälp av interpolering fått ett utseende som är mer likt den del av det faktiska objektet som finns i kuben.

Åter till 2D-exemplet från avsnitt 3.1.1 för att se hur interpolering faktiskt förbättrar resultatet. I figur 12, bild 3 ses hur interpolering ger yta som är mer lik objektet än utan interpolering. Den gröna inre linjekvadraten visar ytan utan interpolering. Kombineras interpolering och minskning av kvadraterna blir resultatet hyfsat nära originalet och en förbättring gentemot att bara minska av kvadraterna. Det gröna linjeobjektet visar ytan utan interpolering, se figur 12, bild 6.



Figur 12: Exempel på konstruktion av ytor med interpolation och kombination av interpolation och små kvadrater. Inspirerad av Ben Anderson [2]

4 Prestandamätning

Det här avsnittet presenterar mätningar gjorda i Inviwo för att jämföra prestandan hos volymsrendering med prestandan hos arching cubes.

4.1 Metod

Mätningarna utförs med hjälp av ett tidtagarur och manuell triggning av ombehandling av datan i Inviwo. Mätningarna utförs på en dator med följande specifikationer:

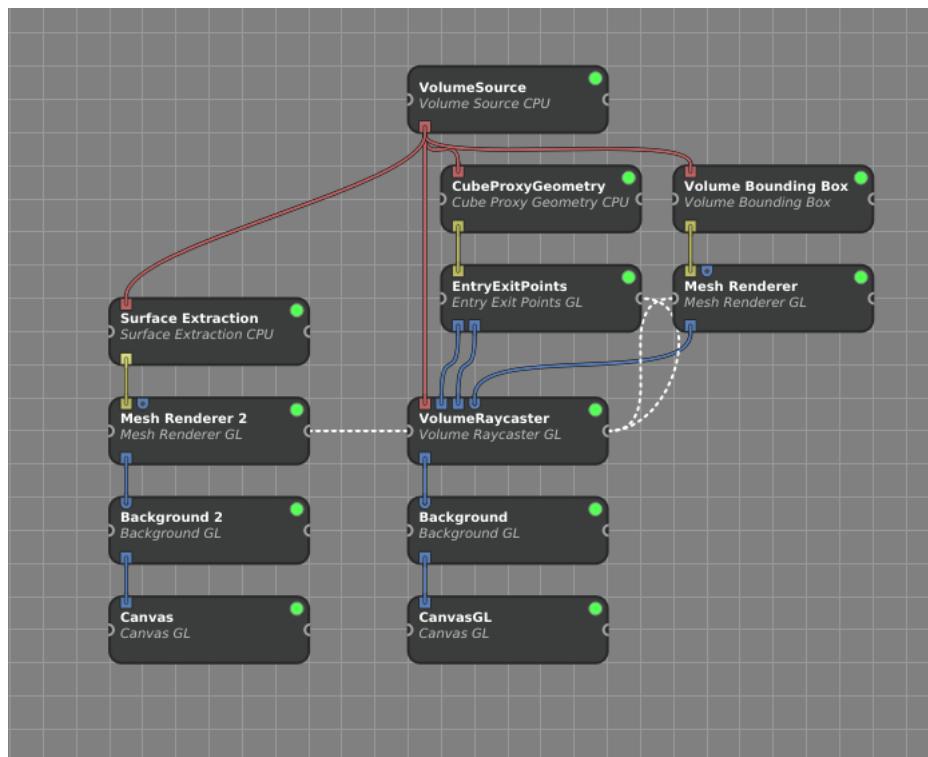
- Processor: Intel(R) Xeon(R) CPU E3-1225 v5 vid 3,30 GHz
- Grafikkort: NVIDIA GM107GL (Quadro K620) revision a2

- Primärminne: 16 GB DDR4

Versionen av Inviwo som används är v0.9.9.

4.1.1 Inviwonätverk

För att jämföra prestandan mellan marching cubes och volymsrendering används nätverket i figur 13. Den vänstra delen bestående av *Surface Extraction*, *Mesh Renderer 2*, *Background 2* och *Canvas* är ansvariga för att utföra marching cubes och rendera resultatet. *VolumeSource* läser in exempeldataten *boron.dat* som innehåller en volym på $150 \times 150 \times 150$ värden. Övriga processorer i nätverket är ansvariga för att utföra volymsrenderingen.



Figur 13: Inviwonätverk för prestandajämförelse.

Processorn *VolumeRaycaster* som är ansvarig för själva volymsrenderingen är inställt med följande inställningar:

- Classification: Transfer function
- Composition: Direct Volume Rendering
- Gradient computation: Central Differences

Dessa inställningar ger en direkt volymsrendering med en enkel färgkodad överföringsfunktion. Sampling rate, som påverkar hur många punkter på en stråle som skall sampelas, ändras för att jämföra prestandan.

Processorn *Surface Extraction* som är ansvarig för att utföra marching tetrahedon, vilket är en variant av marching cubes men i grund och botten samma princip, får ett isovärde satt som varieras för att jämföra olika tider för olika värden [9].

Volymen som visualiseras är densamma för både volymsrenderingen och marching cubes. Den är en kub bestående av $150 \times 150 \times 150 = 3\,375\,000$ punkter.

4.2 Resultat

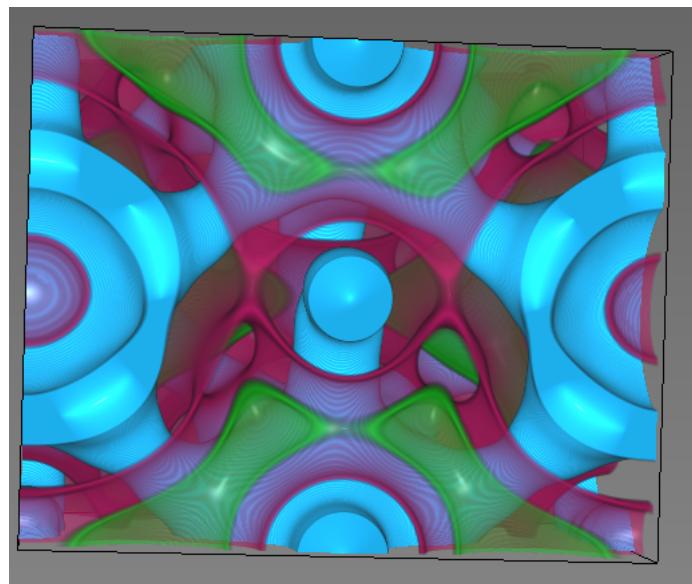
4.2.1 Volymsrendering

I tabell 4.2.1 finns resultaten från mätningarna gjorda för volymsrendering med olika antal sampel per stråle. I samtliga fall är kolumnen tid markerad med ett bindestreck då renderingen gick fortare än det var möjligt att manuellt starta och stoppa tidtagningen.

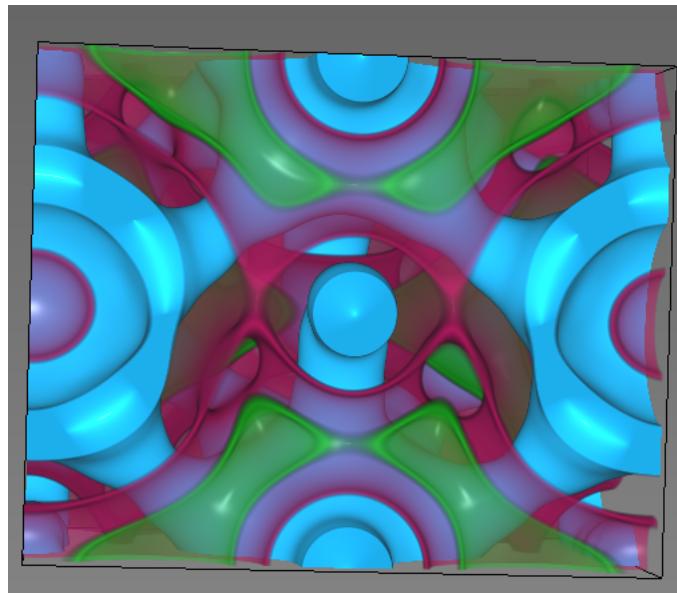
Tabell 1: Mätdata från tidtagning av volymsrendering.

Sampling rate [sampel per stråle]	tid [s]
1	-
8	-
10	-

Även om det inte gick att mäta med tidtaggar var det märkbart att det tog längre tid att rendera med högre antal sampel per stråle då programmet hackade om bilden roterades med 10 sampel per stråle, men inte med en sampel per stråle. Det var också en märkbar skillnad i kvalitet mellan bilden renderad med en sampel per stråle jämfört med bilden som gjordes med 10 sampel per stråle, se figur 14 och 15. Tydliga vågmönster kan ses i figur 14 som inte är närvarande i 15.



Figur 14: Volymsrendering med en sampel per stråle.



Figur 15: Volymsrendering med 10 sampel per stråle.

4.2.2 Marching tetrahedon

Marching tetrahedon tog betydligt längre tid att köras än volymsrenderingen på samma volym, vilket kan ses av resultaten i tabell 4.2.2.

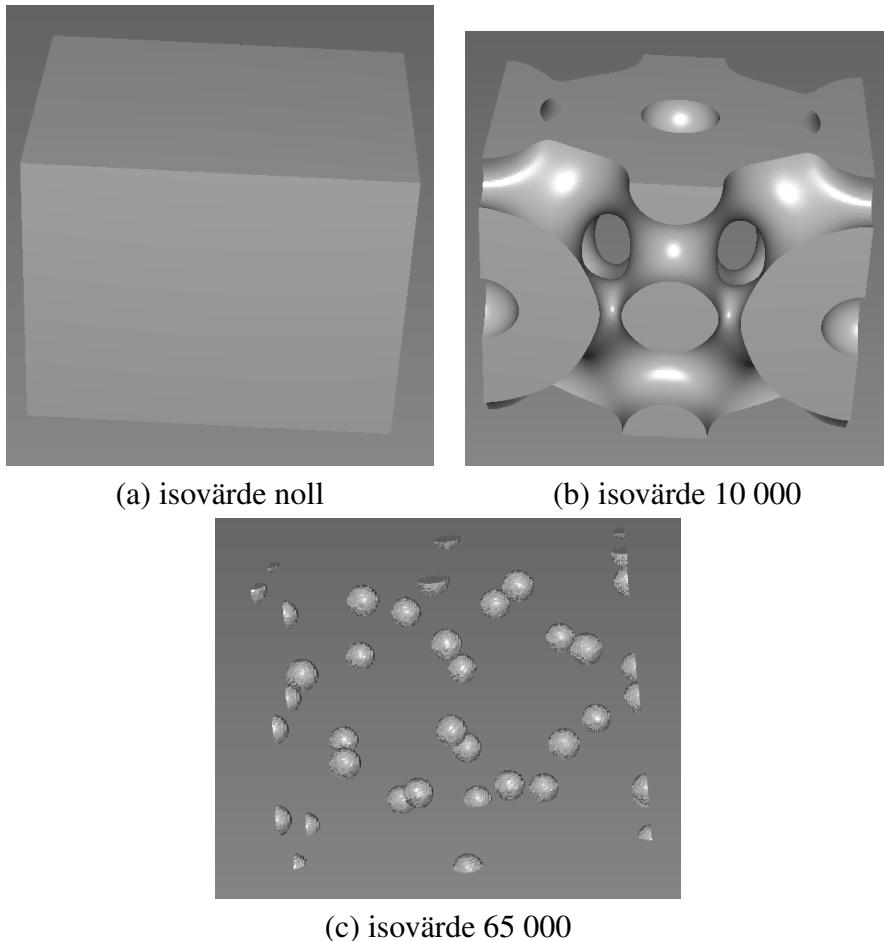
Tabell 2: Mätdata från tidtagning av Marching tetrahedon.

isovärde	tid [s]
0	13
1 000	14
10 000	25
20 000	20
40 000	6
65 000	4

När algoritmen körs med isovärdet noll fylls hela volymen upp och ytan i figur 16 (a) är resultatet. Processen tar i det fallet 13 sekunder, enligt tabell 4.2.2. Då isovärdet skruvas upp stiger först beräkningstiden och detaljer framträder, men vid isovärdet 10 000 nås en brytpunkt och tiden sjunker igen. Under mätningarna var den längsta tiden vid ett isovärde på 10 000 med 25 sekunder och den kortaste tiden med ett isovärde på 65 000 som tog 4 sekunder att köra, se tabell 4.2.2.

Skillnaden i grafiskt resultat åskådliggörs i figur 16 (a-c). isovärdet noll får som synes hela volymen att fyllas, 10 000 skapar en mindre yta och 65 000 resulterar i nästan ingen yta alls.

Till skillnad mot volymsrenderingen märks inget hackande i programmet när den konstruerade ytan roteras eller manipuleras på annat sätt. Någonting som är oberoende av vilket isovärde som valts. Marching tetrahedon-algoritmen och dess relativt långa körtid startar endast om indatan ändras eller någon inställning för algoritmen, t.ex. isovärdet, ändras.



Figur 16: Isoytor genererade med tre olika isovärden.

4.3 Diskussion

Den fösta, kanske mest uppenbara skillnaden, mellan volymsrendering och marching tetrahedon är tiden de tar att utföra. Volymsrenderingen går mycket fortare för en enskild bildruta, medan marching tetrahedon tar flera sekunder på sig att skapa en yta. I gengäld så går ytan skapad av marching tetrahedon mycket fortare att rendera när den väl är klar då algoritmen inte behöver köras om. Här kan en avvägning göras. Önskas t.ex. en rotation av ett väldigt stort objekt kan det hända att volymsrenderingen är för långsam och leder till långa väntetider mellan varje rotation, medan marching tetrahedon i det fallet tänkbart skulle kunna lämnas på över natten för att generera en yta som senare kan vridas i realtid.

Marching tetrahedon är i mätningarna betydligt längsammare än volymsrenderingen, men det är värt att notera att marching cubes eller marching tetrahedon måste gå igenom hela volymen. I det här fallet bestod volymen av 3 375 000 voxlar och en mindre volym skulle ge en betydligt kortare beräkningstid. Volymsrenderingen är inte lika starkt beroende av volymens storlek då den skickar in strålar i volymen och samplar på ett begränsat antal punkter.

Resultatet att marching tetrahedon först ökar i beräkningstid med ett ökat isovärde för att där-efter gå fortare beror på att det tar längre tid att fylla ett volymselement än att lämna det tomt. Detta beror på att ett volymselement som inte är tomt måste, för marching cubes, matchas mot de olika möjligheter som ses i figur 9 och motsvarande för marching tetrahedon. Den beräkning som tog längst tid var för isovärdet 10 000, vilket skulle kunna förklaras av att den har många

kurviga ytor. När en yta är väldigt kurvigt leder det till att de mer komplexa fallen, de fall där många kantlinjer skärs, i figur 9 matchar och då måste mer interpolering utföras, vilket i sin tur ökar beräkningstiden.

Utöver prestandan kan det, från bilderna i resultatdelen, ses att volymsrenderingen och marching tetrahedron löser något olika problem. I projektet för elektronvisualisering behandlas bl.a. elektrontäthet i kristaller och då är det önskvärt med en bild som visar elektrontätheten genom hela kristallens volym. Detta gör att volymsrendering passar bättre. Ett annat fall är visualiseringen av något som kallas Fermi-ytor och som namnet antyder är det en yta som ritas utifrån en kristall. Datat för dessa Fermi-ytor är fortfarande en uppsättning voxlar i en volym, vilket medför att marching tetrahedon möjligtvis är att föredraga.

Både marching tetrahedon och volymsrendering har sina tillämpningsområden. Beroende på vad som skall visualiseras och den prestanda som önskas kan vilken som helst av de två väljas och de verkar generellt ge goda resultat.

Referenser

- [1] *5.11 CHG file.* 29 mars 1999. URL: <http://cms.mpi.univie.ac.at/vasp/guide/node63.html#SECTION0007110000000000000000> (hämtad 2018-04-25).
- [2] Ben Anderson. *An Implementation of the Marching Cubes Algorithm.* URL: http://www.cs.carleton.edu/cs_comps/0405/shape/marching_cubes.html#1 (hämtad 2018-04-17).
- [3] Paul Brouke. *Polygonising a scalar field.* URL: <http://paulbourke.net/geometry/polygonise/> (hämtad 2018-04-17).
- [4] Computer Sweden. *Uppslagninstabell.* URL: <https://it-ord.idg.se/ord/uppslagningstabell/> (hämtad 2018-03-19).
- [5] Computer Sweden. *Volumetrisk.* URL: <https://it-ord.idg.se/ord/volumetrisk/> (hämtad 2018-03-19).
- [6] Xu D och Zhang Y. *Generating triangulated macromolecular surfaces by euclidean distance transform.* 2009. URL: https://zhanglab.ccmb.med.umich.edu/papers/2009_3.pdf (hämtad 2018-05-23).
- [7] Matthew Fisher. *Marching Cubes.* URL: <https://graphics.stanford.edu/~mdfisher/MarchingCubes.html> (hämtad 2018-04-17).
- [8] James D. Foley m. fl. *Computer Graphics: Principles and Practice.* Addison-Wesley, 1990. ISBN: 0201121107.
- [9] Charles D. Hansen och Chris R. Johnson. *Visualization Handbook.* Academic Press, 2004. ISBN: 012387582X.
- [10] Milan Ikits m. fl. *GPU GEMS. Chapter 39.2.* URL: https://developer.nvidia.com/gpugems/GPUGems/gpugems_ch39.html (hämtad 2018-03-09).
- [11] Milan Ikits m. fl. *GPU GEMS. Chapter 39.* URL: https://developer.nvidia.com/gpugems/GPUGems/gpugems_ch39.html (hämtad 2018-03-09).
- [12] Inviwo. *Features.* URL: <http://www.inviwo.org/user-documentation/features/> (hämtad 2018-03-09).
- [13] Barthold Lichtenbelt och Randy Crane. *Introduction to Volume Rendering (Hewlett-Packard Professional Books).* Prentice Hall, 1998. ISBN: 0138616833.
- [14] Jörgen Malmquist, Staffan Cederblom och Susanne Wikman. *Nationalencyklopedin. magnetisk resonanstomografi.* URL: <http://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/magnetisk-resonanstomografi> (hämtad 2018-04-25).
- [15] Mathworks. *Overforingsfunktion.* URL: <https://se.mathworks.com/discovery/transfer-function.html> (hämtad 2018-03-19).
- [16] Nationalencyklopedin. *gradient.* URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/gradient> (hämtad 2018-03-19).
- [17] Nationalencyklopedin. *implicitfunktion.* URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/implicit-funktion> (hämtad 2018-04-25).
- [18] Nationalencyklopedin. *interpolation.* URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/interpolation> (hämtad 2018-04-25).

-
- [19] *Nationalencyklopedin. sampling.* URL: <http://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/sampling> (hämtad 2018-04-26).
 - [20] *Nationalencyklopedin. Tetraeder.* URL: <https://svenska.se/tre/?sok=tetraeder&pz=2> (hämtad 2018-04-27).
 - [21] Richard C. Penney. *Linear algebra. ideas and applications.* Hoboken, New Jersey : Wiley, 2016., 2016. ISBN: 9781118909584.
 - [22] Anders Ramgard. *Vektoranalys.* Teknisk högskolelitteratur i Stockholm AB, 2000.
 - [23] *SciVis Group. Inviwo.* URL: <http://scivis.itn.liu.se/research/inviwo/> (hämtad 2018-03-09).
 - [24] *Svensk ordbok. skalär.* URL: <https://svenska.se/tre/?sok=skal%C3%A4r&pz=1> (hämtad 2018-04-25).

E Gruppkontrakt

Gruppkontrakt - Visualisering av elektronstrukturer

Om gruppen har beslutat att träffas skall alla deltagare vara på plats på förutbestämd tid. Vid sjukdom eller dylikt skall gruppmedlemmen meddela övriga om detta.

Tider som enbart är specificerade med timme medför att akademisk kvart tillämpas.

Varje möte skall inledas med en genomgång av vad varje person har gjort. Detta så att alla har koll på hur resten av gruppen ligger till.

I slutet av varje gruppmöte skall en genomgång av vad som skall göras inför nästa möte hållas.

Protokoll från mötet skall skickas ut av mötessekreteraren för att komma de andra i gruppen till handa.

Gruppen skall ses en gång i veckan under en lunch, om inte annat beslutas, för att stämma av hur arbetet fortskridet.

Arbetsbördan under projektets gång skall delas rättvist av gruppmedlemmarna.

Har någon medlem problem med gruppen bör detta tas upp på ett av de schemalagda gruppmötena, eller vid annat lämpligt tillfälle.

Gruppen strävar efter att göra ett så bra arbete som möjligt inom de 1 000 timmar som budgeterats åt projektet.

F Kravspecifikation

Kravspecifikation

Redaktör: Andreas Kempe

Version 1.1

Status

Granskad	Anders Rehult	2018-05-18
Godkänd	Rickard Armiento	2018-05-21

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz		073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	iv
1 Inledning	1
1.1 Parter	1
1.2 Syfte och Mål	1
1.3 Användning	1
1.4 Bakgrundsinformation	1
1.5 Definitioner	2
2 Utförande	3
3 Översikt av systemet	4
3.1 Grov beskrivning av produkten	4
3.2 Produktkomponenter	4
3.3 Beroenden till andra system	4
3.4 Ingående delsystem	4
3.5 Avgränsningar	4
3.6 Designfilosofi	5
3.7 Generella krav på hela systemet	6
4 Delsystem - Datakonvertering	9
5 Delsystem - Datahantering	10
6 Delsystem - Visualisering	11
7 Krav på vidareutveckling	12
8 Ekonomi	13
9 Leveranskrav och delleveranser	14
10 Dokumentation	15
11 Utbildning	16
Referenser	16

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-01-24	Första utkast.	Projektgruppen	Marian Brännvall
0.2	2018-01-26	Andra utkast.	Projektgruppen	Anders Rehult
0.3	2018-01-29	Tredje utkast.	PL och DOK	Anders Rehult
1.0	2018-01-30	Slutgiltig version.	PL	Anders Rehult
1.1	2018-05-18	Krav 5 omförhandlat.	Projektgruppen	Anders Rehult

1 Inledning

I detta dokument beskrivs alla krav med en tabellrad enligt nedan. Kravnummer är löpande genom hela dokumentet. Kolumn 2 anger om kravet är ett originalkrav eller om kravet har reviderats. Vid revidering finns en hänvisning till beslut. I kolumn 3 finns själva lydelsen av kravtexten. I kolumn 4 finns dess prioritet beskriven.

Prioritet *Ska* betyder att kravet är ett av baskraven som projektet måste uppfylla. Prioritet *Bör* betyder att kravet inte är avgörande för projektets fullbordande, men att kravet bör uppfyllas. Prioritet *Kanske* är den lägsta prioritetens, och betyder att kravet inte är avgörande för projektets fullbordande och inte kommer arbetas med förrän alla *Bör-* och *Ska*-krav är uppfyllda.

Krav nr x	Förändring	Kravtext för krav nr x	Prioritet
-----------	------------	------------------------	-----------

1.1 Parter

Rickard Armiento är beställare av detta projekt med Johan Jönsson som handledare och Anders Rehult som projektledare. Övriga gruppmedlemmar är Viktor Bernholtz, Andreas Kempe och Marian Brännvall.

1.2 Syfte och Mål

Målet med projektet är att utveckla ett system för visualisering av resultatet av elektronstrukturberäkningar. Detta ska göras i visualiseringsverktyget Inviwo och systemets funktionalitet ska demonstreras genom att använda det för att illustrera resultat från befintliga beräkningar. I och med att den framtagna mjukvaran ämnas användas i forskningssammanhang måste projektet hålla en hög vetenskaplig och teknisk kvalitet.

Utöver det konkreta målet med framtagandet av mjukvara för visualisering ska även projektet ge projektmedlemmarna erfarenhet av att arbeta i projekt och utöka deras förmåga till analytiskt och fysikaliskt tänkande för att ge värdefull erfarenhet inför arbetslivet.

1.3 Användning

Inom teoretisk fysik är elektronstrukturberäkningar ett viktigt verktyg för att förstå hur materials och molekylers egenskaper utifrån kvantmekaniska principer. Denna produkt kommer användas vid Linköpings universitet för att analysera data från sådana beräkningar.

1.4 Bakgrundsinformation

Detta projekt genomförs som en del i kursen TFYA75 vid Linköpings universitet. Visualisering av data från beräkningar kan i vissa fall förenkla, och ofta vara nödvändigt, för att kunna analysera datan och förstå materials och molekylers egenskaper.

Inviwo gör det möjligt att styra visualisering programmatiskt och att konstruera användargränsnitt för interaktiv visualisering.

1.5 Definitioner

Python är ett programmeringsspråk som används i Inviwo för att knyta samman processorer.

C++ är ett programmeringsspråk som är baserat på programspråket C. I Inviwo används det för att skriva programkod till processorer.

Inviwo (Interactive Visualization Workshop) är programvara för visualisering som tillhandahåller en nätverksredigerare för designen av dataflödesnätverk. Noderna i dessa dataflödesnätverk kallas processorer. Indata till nätverket behandlas i dessa processorer och utdata genereras.

BSD 2 är en licens för öppen källkod.

Git är ett versionshanteringsprogram.

Lisam är en lärplattform som används i projektet för att dela och samla information.

HDF5 är ett binärt filformat.

GUI (Graphical Use Interface) är ett grafiskt användargränssnitt.

LIPs är en modell med regler, instruktioner och mallar för att bedriva projekt. Detta projekt utformas utifrån LIPsmodellen.

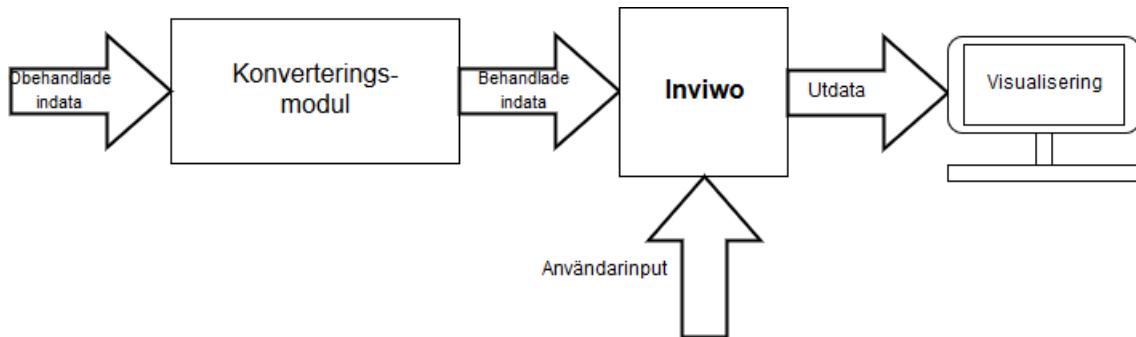
API (Application Programming Interface) är en specifikation av hur olika applikationer kan använda och kommunicera med en specifik programvara. Detta utgörs oftast av ett dynamiskt länkat bibliotek.

2 Utförande

Här listas krav för utförandet av projektet.

Krav nr 1	Original	Projektet ska drivas enligt LIPS-modellen.	Ska
Krav nr 2	Original	Vid begäran ska projektgruppen skicka en statusrapport till beställaren.	Ska

3 Översikt av systemet



Figur 1: Grov skiss av systemet

Systemet tar in data från elektronstrukturberäkningar och visualiseras egenskaper valda av användaren. Detta illustreras i figur 1.

3.1 Grov beskrivning av produkten

Produkten är ett verktyg för att visualisera viktiga egenskaper från elektronstrukturberäkningar.

3.2 Produktkomponenter

Produkten ska bestå av API:er för att programmatiskt utföra visualisering samt, eventuellt, ett grafiskt användargränssnitt. En demonstration av funktionalitet och en teknisk dokumentation ingår även i slutleveransen, se sektion 8.

3.3 Beroenden till andra system

Projektet använder sig av Inviwo för att behandla indata i form av resultat från elektronstrukturberäkningar och visualisera relevant data på ett interaktivt sätt.

3.4 Ingående delsystem

Systemet består av tre delsystem, ett som hanterar konvertering av data till format som resten av systemet kan arbeta med, ett som hanterar den konverterade datan och ett som hanterar visualiseringen av denna data. Datat som ska visualiseras kommer att kunna väljas av användaren.

3.5 Avgränsningar

Projektet innehåller att visualisera två egenskaper från listan i krav 5, varav den ena som väljs ska vara en som redan påbörjats av 2017 års projektgrupp och den andra ska göras från grunden.

3.6 Designfilosofi

Systemet kommer utvecklas som en påbyggnad av 2017 års projektgrupp. Kod som inte fungerar med den aktuella versionen av Inviwo kommer att uppdateras.

Projektet kommer att drivas med hjälp av versionshanteringssystemet Git och koden kommer vara licensierad med BSD 2, men även utvecklas under Inviwos utvecklaravtal för att, om önskvärt, kunna officiellt integreras i programvaran.

All design av systemet kommer att utgå från en designspecifikation som löpande kommer att uppdateras under projektets gång. I slutändan kommer detta att resultera i teknisk dokumentation som beskriver systemet.

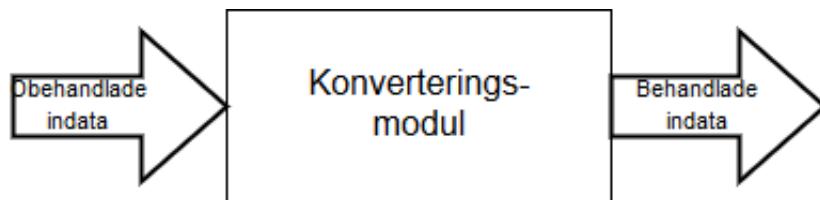
3.7 Generella krav på hela systemet

Här listas krav för det generella systemet.

Krav nr 3	Original	Källkoden i systemet ska vara BSD 2-clauselicensierad.	Ska
Krav nr 4	Original	Kod som integreras med Inviwo ska tillgängliggöras under Inviwos utvecklaravtal.	Ska
Krav nr 5	Omförhandlat 2018-05-17	Projektgruppen ska utöka alternativt implementera visualisering av minst en av nedanstående egenskaper, samt uppdatera minst en redan existerande implementation av någon av de nedanstående egenskapserna: <ul style="list-style-type: none"> • Elastiska konstanter • Fermi-ytor • ELF (Electron Localization Function) • Krafter på atomer • Bandstruktur • Total DOS (Density Of States) • Parkorrelationsfunktionen • Illustration av partiell elektrondensitet 	Ska
Krav nr 6	Original	Projektgruppen ska undersöka och lära sig om samtliga egenskaper ur listan i kravet ovan.	Ska
Krav nr 7	Original	Tillhandahållna python-moduler ska kunna anropas med enkla funktionsanrop.	Ska
Krav nr 8	Original	Tillhandahållna python-moduler ska kunna hantera indata.	Ska
Krav nr 9	Original	En beskrivning av vilka indata en tillhandahållen python-modul kräver ska kunna erhållas.	Ska
Krav nr 10	Original	En beskrivning av vilka utdata en tillhandahållen python-modul producerar ska kunna erhållas.	Ska
Krav nr 11	Original	En beskrivning av vad en tillhandahållen python-modul gör ska kunna erhållas.	Ska
Krav nr 12	Original	Användaren ska kunna ändra indata till python-moduler.	Ska
Krav nr 13	Original	Användaren ska kunna välja vilken typ av visualisering som ska visas.	Ska

Krav nr 14	Original	Användaren ska kunna länka samman olika python-moduler.	Ska
Krav nr 15	Original	Systemet ska implementeras i Inviwo.	Ska
Krav nr 16	Original	Tillhandahållna python-moduler ska kunna ta emot en eller flera typer av indata.	Ska
Krav nr 17	Original	Tillhandahållna python-moduler ska kunna ge utdata.	Ska
Krav nr 18	Original	Tillhandahållna python-moduler ska inte fortskrida som vanligt vid indata av fel typ.	Ska
Krav nr 19	Original	Tillhandahållna python-moduler ska vid felaktiga indata varna användaren.	Ska
Krav nr 20	Original	Tillhandahållna python-modulers egenskaper bör kunna ändras.	Bör
Krav nr 21	Original	Systemet bör tillhandhålla ett grafiskt gränssnitt (GUI) för vanligt förekommande visualiseringsuppgifter.	Bör
Krav nr 22	Original	Uppstart av systemet bör vara enkelt för användaren.	Bör
Krav nr 23	Original	Installation av systemet bör vara enkelt för användaren.	Bör

4 Delsystem - Datakonvertering



Figur 2: Skiss av hur systemet hanterar indata

Här listas krav för hur systemet ska hantera indata och konvertera de till format som resten av systemet kan arbeta med. Datakonverteringsmodulen tar in obehandlade indata från något beräkningsprogram, behandlar dem så att resten av systemet kan hantera dem, och skickar sedan vidare dem. Processen illustreras i figur 2.

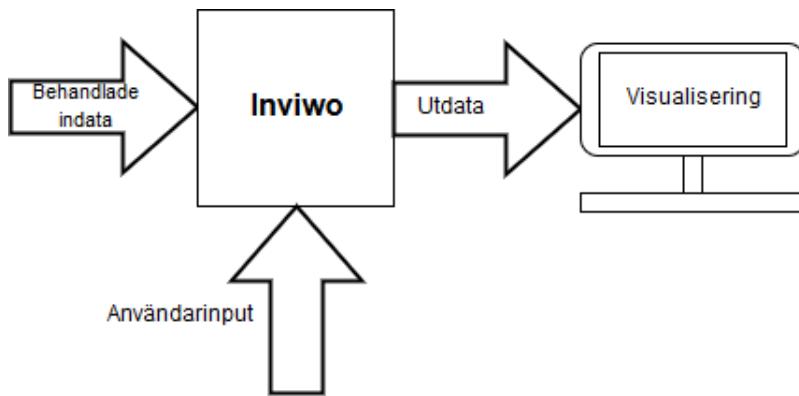
Krav nr 24	Original	Systemet ska kunna läsa in resultat från VASP.	Ska
Krav nr 25	Original	Systemet ska kunna konvertera data från kristallstrukturberäkningar.	Ska
Krav nr 26	Original	Systemet ska kunna konvertera data från elektronstrukturberäkningar.	Ska
Krav nr 27	Original	Systemet ska kunna konvertera data från tillståndstähetsberäkningar.	Ska
Krav nr 28	Original	Systemet ska översätta input-filer i textformat till det binära filformatet HDF5.	Ska
Krav nr 29	Original	Systemet bör kunna läsa in resultat från något annat beräkningsprogram, t.ex. Elk.	Bör
Krav nr 30	Original	Systemet bör utnyttja befintlig kod för hantering av inläsning av datafiler.	Bör

5 Delsystem - Datahantering

Här listas krav för hur systemet ska kunna hantera data.

Krav nr	Original	Systemet ska kunna hantera data från kri-	Ska
Krav nr 31	Original	Systemet ska kunna hantera data från kri-	Ska
Krav nr 32	Original	Systemet ska kunna hantera data från	Ska
Krav nr 33	Original	Systemet ska kunna hantera data från till-	Ska
Krav nr 34	Original	Systemet bör utnyttja befintlig kod för han-	Bör
Krav nr 35	Original	Systemet bör effektivt kunna hantera stora	Bör
		filer.	

6 Delsystem - Visualisering



Figur 3: Skiss av hur visualiseringen sker

Här listas krav för hur systemet ska visualisera data. Visualiseringssmodulen tar in indata behandlade av datakonverteringsmodulen, behandlar dem, och ritar sedan upp utdata på skärmen, se figur 3.

Krav nr	Original	Systemet ska visualisera projicerad tillståndstäthet härrörande till varje separat atom i en kristalls enhets-cell.	Ska
Krav nr 37	Original	Systemet ska visualisera kristallstruktur som atompositioner i enhetscellen.	Ska
Krav nr 38	Original	Systemet ska kunna visualisera den elektrontäthet som resulterar från en beräkning.	Ska
Krav nr 39	Original	Användaren ska, vid volymsrendering, kunna reglera en brytpunkt, i form av något specifikt värde av någon egenskap, för vilken full transparens inträder.	Ska
Krav nr 40	Original	Användaren ska, vid volymsrendering, kunna reglera ett intervall av värden av någon egenskap, där full transparens fås för alla värden inom intervallet.	Ska
Krav nr 41	Original	Användaren ska, vid volymsrendering, kunna ändra opaciteten/transparensnivån för valfria värden av någon egenskap.	Ska
Krav nr 42	Original	Systemet bör tillåta dynamisk visualisering baserad på en serie atompositioner från utdatafiler.	Bör
Krav nr 43	Original	Systemet bör tillåta att visualisering av egenskaper tillhörande atomer bara visas på vissa atomer, som kan väljas dynamiskt med musklick.	Bör

7 Krav på vidareutveckling

Här listas krav för hur systemet ska kunna vidareutvecklas.

Krav nr	Original	Innehåll	Ska
Krav nr 44	Original	All kod ska vara välkommenterad alternativt självförlarande.	
Krav nr 45	Original	Koden ska fungera för Inviwo 0.9.9 eller någon efterkommande version.	
Krav nr 46	Original	Den tekniska dokumentationen ska vara tydlig så att det är möjligt att bygga vidare på systemet utifrån denna.	

8 Ekonomi

Projektgruppen har total 1000 timmar för att slutföra projektet, dessa är fördelade jämnt över gruppens medlemmar.

Krav nr 47	Original	Efter godkänd projektplan (BP2) ska projektet maximalt ta 1000 arbets timmar att slutföra.	Ska
Krav nr 48	Original	Projektgruppen ska utföra kontinuerlig tidsredovisning.	Ska
Krav nr 49	Original	Tidsredovisning ska skickas till beställaren inför varje beslutspunkt.	Ska

9 Leveranskrav och delleveranser

Tisdag 30/1 Kravspecifikationen ska vara klar och godkänd (BP1).

Tisdag 13/2 Första version av projektplan, tidplan och systemskiss ska vara inlämnad till beställaren.

Fredag 16/2 Slutgiltig version av projektplan, tidplan och systemskiss ska vara inlämnad till beställaren, efter detta hålls beslutmöte BP2.

Fredag 2/3 Första version av designspecifikationen ska vara inlämnad till handledaren.

Senast 8/3 Designspecifikationen ska vara godkänd av handledaren vid beslutsmöte BP3.

Senast 11/4 Nuvarande design ska vara presenterad för och godkänd av handledaren vid ett beslutsmöte BP4.

Senast 14/5 Färdig kappa för slutrappart ska vara inskickad till TEMA.

Senast 23/5 Kraven ska vara verifierade (BP5).

Senast 28/5 Teknisk dokumentation, resultat av teknisk/naturvetenskaplig undersökning, användarhandledning och slutrappart för kandidatarbetet ska vara godkända.

Tisdag 5/6 Efterstudien ska vara inlämnad och inkluderad i slutrapparten. Gruppens källkod ska ha lagrats i ett system för källkodshantering och lämnats in i denna form.

Fredag 8/6 All utrustning och lånat material ska vara återlämnat.

En uppdaterad tidrapport ska lämnas till beställare inför varje beslutspunkt.

Här listas krav för leveranser och delleveranser.

Krav nr 50	Original	Vid Slutleverans ska det finnas ett fungerande interaktivt visualiseringssystem.	Ska
Krav nr 51	Original	Vid Slutleverans ska det finnas en teknisk/naturvetenskaplig rapport.	Ska
Krav nr 52	Original	Projektets delleveranser ska ske senast vid de datum som specificeras på kursens hemsida.	Ska

10 Dokumentation

Dokumentation av projektet ska utgå från LIPS-mallar. Syftet med dokumentationen är att ha en grund som utgås ifrån när arbetet sätter igång. Krav på systemet specificeras, designen specificeras och en projektplan, tidplan och systemskiss görs. Se tabell nedan för information om vilka dokument som ska skrivas.

Krav nr 53	Original	Projektgruppen ska ta fram en tidsplan.	Ska
Krav nr 54	Original	Dokumentationen ska sparas på kursens hemsida.	Ska
Krav nr 55	Original	Projektets slutleverans ska ske senast vid det datum som finns specificerat på kursens hemsida.	Ska
Krav nr 56	Original	Vid slutleverans ska det finnas en slutförslag.	Ska
Krav nr 57	Original	Vid slutleverans ska det finnas teknisk dokumentation med användaranvisning.	Ska
Krav nr 58	Original	Samtliga dokument som projektgruppen tar fram ska godkännas enligt listan med leveranser, se avsnitt 9.	Ska
Krav nr 59	Original	Designspecifikationen ska godkännas av handledaren.	Ska
Krav nr 60	Original	Projektgruppen ska ta fram en kravspecifikation.	Ska
Krav nr 61	Original	Projektgruppen ska ta fram en systemskiss.	Ska
Krav nr 62	Original	Projektgruppen ska ta fram en projektplan.	Ska
Krav nr 63	Original	Projektets dokument ska utgå ifrån LIPS-mallar.	Ska

11 Utbildning

Gruppens medlemmar kommer att utbildas i visualiseringsverktyget Inviwo och programmeringsspråket Python. Föreläsningar om relevanta fysikaliska fenomen kommer dessutom att hållas.

Referenser

- [1] *LIPS – nivå 1. Version 1.0.* Tomas Svensson och Christian Krysander. Kompendium, LiTH, 2002.

Projektdirektiv

Visualisering av elektronstruktur

Personer

- Beställare: Rickard Armiento
- Handledare: Johan Jönsson
- Expert: Peter Steneteg

Bakgrund

Elektronstrukturberäkningar är ett viktigt verktyg inom teoretisk fysik för att förstå hur materials och molekylers egenskaper kan härledas från kvantmekaniska effekter. För att förstå dessa egenskaper är det viktigt att kunna analysera data från sina beräkningar, något som i vissa fall förenklas genom visualisering och ofta helt och hållet kräver att man kan visualisera sin data. Inviwo är ett kraftfullt forskningsverktyg som utvecklas av Visualiseringscenter i Norrköping. Inviwo gör det möjligt att styra visualisering med programmering och att konstruera användargränssnitt för interaktiv visualisering.

Projektidé

Projektet går ut på att skapa ett verktyg för att visualisera viktiga egenskaper från elektronstrukturberäkningar. Verktyget ska bestå av APIer för att programmatiskt utföra visualisering samt eventuellt ett grafiskt användargränssnitt för dessa APIer.

Mjukvara ifrån tidigare projektomgång finns tillgänglig för projektgruppen och får användas för att underlätta uppfyllandet av kraven.

Syfte

Projektet syftar till att utveckla kreativiteten samt att ge färdigheter i fysikaliskt tänkande och analys av teoretiska resultat. Projektet bedrivs så realistiskt som möjligt för att vara en träning inför det kommande yrkeslivet. Resultatet av projektarbetet ska hålla hög vetenskaplig och teknisk kvalité och

baseras på moderna kunskaper, dokumenteras i form av projekt-och tidsplan, krav-och designs specification samt i en teknisk/vetenskaplig rapport, presenteras muntligt, demonstreras och följas upp i en efterstudie.

Mål

Att i visualiseringssverktyget Inviwo utveckla ett system för visualisering av resultatet av elektronstrukturberäkningar. Att demonstrera systemets funktionalitet genom att använda det till att illustrera några befintliga beräkningsresultat.

Krav på systemet

- Systemet ska implementeras i Inviwo.
- Källkoden i systemet bör licensieras med BSD 2-clause "simplified" licence.
- Kod som integreras med Inviwo ska tillgängliggöras under Inviwos utvecklaravtal.
- Tillhandahållna python-moduler ska vara användarvänliga och möjliggöra visualisering med kommandon på hög nivå.
- Systemet bör effektivt kunna hantera stora filer.
- Systemet bör översätta input-filer i textformat till det binära filformatet HDF5.
- Systemet bör tillhandahålla ett grafiskt gränssnitt (GUI) för vanligt förekommande visualiseringssuppgifter.
- Installation och uppstart av systemet bör vara enkel för användaren.
- Systemet bör utnyttja befintlig kod för hantering av inläsning av datafiler och hantering av bl.a. kristallstrukturer.
- Ska kunna läsa in resultat skapade med beräkningsprogrammet VASP.
- Bör kunna läsa inresultat från något annat beräkningsprogram, t.ex. Elk.
- Systemet ska kunna läsa indata direkt ifrån utdatafiler ifrån beräkningsprogram och visualisera detta.
- Ska visualisera kristallstruktur som atompositioner i enhetscellen.
- Systemet ska kunna visualisera den elektrontäthet som resulterat ifrån en beräkning.
- Visualiseringen ska utnyttja Inviwos funktionalitet för volymsrendering för partiell transparens.
- Visualiseringen ska tillåta interaktion i form av rotering, skalning, etc.
- Användaren ska kunna reglera en brytpunkt för vilken full transparensinträder för att kunna tydliggöra strukturer bättre.
- Systemet bör tillåta dynamisk visualisering baserad på en serie av atompositioner i utdatafiler.
- Ska visualisera projicerad tillståndstäthet härrörande tillvarje separat atom i en kristalls enhets-cell.

- Tillåta att visualisering tillhörande atomer bara visas på vissa atomer, som kan väljas dynamiskt med t.ex. musklick.
- Ska implementera (alternativt utöka befintlig implementation med) visualisering av minst två av följande egenskaper:
 - Elastiska konstanter.
 - Fermi-ytor.
 - ELF.
 - Krafter på atomer.
 - Bandstruktur.
 - Total DOS.
 - Parkorrelationsfunktionen.
 - Illustration av partiell elektrondensitet.

Slutgodkännande

För systemets godkännande krävs en interaktiv demonstration av systemets färdigheter som utförs i samband med leverans.

Leveranser

Se kursens Lisamsida.

Övriga krav

Projektet ska bedrivas enligt LIPS-modellen och samtliga dokument ska utgå från LIPS-mallar.

I förefasen ingår att projektgruppen ska ta fram en kravspecifikation, en systemskiss och en projektplan med tidplan. Samtliga dessa dokument ska godkännas av beställaren. Efter godkänd projektplan (BP2) får projektet ta maximalt 1500 arbetstimmar att slutföra.

Projektgruppen ska utföra kontinuerlig tidsredovisning som skickas till beställaren en gång per vecka. Vid begäran ska gruppen även skicka in en statusrapport. Vid slutleveransen ska det finnas ett fungerande interaktivt visualiseringssystem, teknisk dokumentation med användaranvisning, en teknisk/naturvetenskaplig rapport samt slutrapport. Projektets delleveranser

och slutleverans ska senast ske vid de datum som finns specificerade på kursens Lisamsida.
Även

formen för slutleveransen beskrivs på denna sida. Dokumentationen sparas på kursens Lisamsida.

H Projektplan

Projektplan

Redaktör: Anders Rehult

Version 1.0

Status

Granskad	VB	18-02-20
Godkänd		

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	v
1 Parter	1
2 Översiktlig beskrivning av projektet	1
2.1 Syfte och Mål	1
2.2 Leveranser	1
2.3 Begränsningar	2
3 Fasplan	2
3.1 Före-fasen	2
3.2 Under-fasen	3
3.3 Efter-fasen	3
4 Organisationsplan för hela projektet	3
4.1 Organisationsplan	3
4.2 Villkor för samarbetet inom projektgruppen	4
4.3 Definition av arbetsinnehåll och ansvar	4
5 Dokumentplan	4
6 Utvecklingsmetodik	5
7 Utbildningsplan	6
7.1 Egen utbildning	6
7.2 Kundens utbildning	6
8 Rapporteringsplan	6
9 Mötesplan	7
10 Resursplan	7
10.1 Personer	7
10.2 Material	7
10.3 Lokaler	7
10.4 Ekonomi	7

11 Milstolpar och beslutspunkter	7
11.1 Milstolpar	8
11.2 Beslutspunkter	8
12 Aktiviteter	9
12.1 Dokumentation	9
12.2 Möte	10
12.3 Utbildning	10
12.4 Presentation	11
12.5 Delsystem datakonvertering	11
12.6 Delsystem visualisering	13
12.7 Tester	15
12.8 Övrigt	16
13 Tidsplan	16
14 Förändringsplan	16
15 Kvalitetsplan	16
15.1 Granskningar	16
15.2 Testplan	17
16 Riskanalys	17
17 Prioriteringar	17
18 Projektavslut	17
Bilaga A Gruppkontrakt	18
Bilaga B Kravspecifikation	19
Bilaga C Projektdirektiv	38
Bilaga D Systemskiss	41
Bilaga E Tidsplan	51

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-02-13	Första utkast.	Projektgruppen	VB
0.2	2018-02-14	Andra utkast.	Projektgruppen	AK
0.3	2018-02-16	Tredje utkast.	Projektgruppen	AK
0.4	2018-02-20	Fjärde utkast.	Projektgruppen	VB
1.0	2018-02-20	Godkänd version.	Projektgruppen	VB

1 Parter

I Tabell 1 listas de olika parter som är inblandade i projektet. En schematisk bild över de ingående parternas relationer i projektet hittas i avsnitt 4.1.

Roll	Namn	E-post
Beställare	Rickard Armiento	rickard.armiento@liu.se
Expert	Rickard Englund	rickard.englund@liu.se
Handledare	Johan Jönsson	johan.jonsson@liu.se
Projektledare	Anders Rehult	andre449@student.liu.se

Tabell 1: Parter i organisationen.

2 Översiktlig beskrivning av projektet

I detta avsnitt beskrivs syfte och mål med projektet, leveranser i projektet samt projektets begränsningar.

2.1 Syfte och Mål

Målet med projektet är att utveckla ett system för visualisering av resultat av elektronstrukturberäkningar. Detta ska göras i visualiseringsverktyget Inviwo och systemets funktionalitet ska demonstreras genom att använda det för att illustrera resultat från befintliga beräkningar. I och med att den framtagna mjukvaran ämnas användas i forskningssammanhang måste projektet hålla en hög vetenskaplig och teknisk kvalitet.

Utöver det konkreta målet med framtagandet av mjukvara för visualisering ska även projektet ge projektmedlemmarna erfarenhet av att arbeta i projekt och utöka deras förmåga till analytiskt och fysikaliskt tänkande för att ge värdefull erfarenhet inför arbetslivet.

2.2 Leveranser

I projektet ingår ett antal leveranser, dessa listas nedan.

- **Tisdag 30/1** Kravspecifikationen ska vara klar och godkänd. Lämnas in till handledare och beställare.
- **Tisdag 13/2** Första version av projektplan, tidsplan och systemskiss ska vara inlämnad till beställaren.
- **Fredag 16/2** Slutgiltig version av projektplan, tidsplan och systemskiss ska vara inlämnad till beställaren.
- **Fredag 2/3** Första version av designspecifikation ska vara inlämnad till handledaren.
- **Senast 8/3** Designspecifikationen ska vara godkänd av handledaren.

- **Senast 11/4** Nuvarande design ska vara presenterad för och godkänd av handledaren.
- **Senast 14/5** Färdig kappa för slutrappport ska vara inskickat till TEMA.
- **Senast 23/5** Kraven ska vara verifierade av handledare och beställare.
- **Senast 28/5** Teknisk dokumentation, resultat av teknisk/naturvetenskaplig undersökning, användarhandledning och slutrappport för kandidatarbetet ska godkända av handledare och beställare.
- **Tisdag 5/6** Efterstudien ska vara inlämnad och inkluderad i slutrapporten, lämnas till handledare och beställare. Gruppens källkod ska ha lagrats i ett system för källkodshantering och lämnats in i denna form till handledare och beställare.
- **Fredag 8/6** All utrustning och lånat material ska vara återlämnat.
- **Inför varje beslutspunkt** En uppdaterad tidrapport lämnas till beställaren.
- **Preliminärt vecka 22** En slutleverans till beställaren.

2.3 Begränsningar

I projektet kommer visualiseringssverktyget Inviwo och programmeringspråken Python och C++ användas. Det kommer inte utredas om det är bättre att använda andra verktyg.

3 Fasplan

Projektet drivs enligt LIPS-modellen, som är en modell med regler, instruktioner och mallar för att bedriva projekt. Projektet delas upp i tre faser - före, under, och efter. Se avsnitt 11 för information om beslutspunkter och avsnitt 13 för en detaljerad tidsplan.

3.1 Före-fasen

Före-fasen är ämnad att undersöka om projektet bör genomföras och, om så är fallet, definiera och konkretisera projektets mål samt organisera arbetet som krävs för att uppnå dessa. Under före-fasen skrivs bland annat ett *projektdirektiv* (Bilaga C), en *kravspecifikation* (Bilaga B), en *systemskiss* (Bilaga D) och en projektplan.

Projektdirektivet skrevs innan projektgruppen skapades och gavs till projektgruppen vid projektets start.

Projektgruppen har under före-fasen skrivit ett *gruppkontrakt* (Bilaga A), en kravspecifikation, en systemskiss samt denna projektplan.

3.2 Under-fasen

Under under-fasen utförs det arbete som leder till att projektets krav uppfylls. Projektgruppen skriver en *designspecifikation* som beskriver hur systemet ska konstrueras i mer detalj än systemskissen. Projektgruppen arbetar sedan med att, utgående från denna designspecifikation, konstruera och testa systemet. En *teknisk dokumentation* skrivas parallellt med systemets konstruktion för att reflektera den faktiska implementationen av idéerna beskrivna i designspecifikationen.

3.3 Efter-fasen

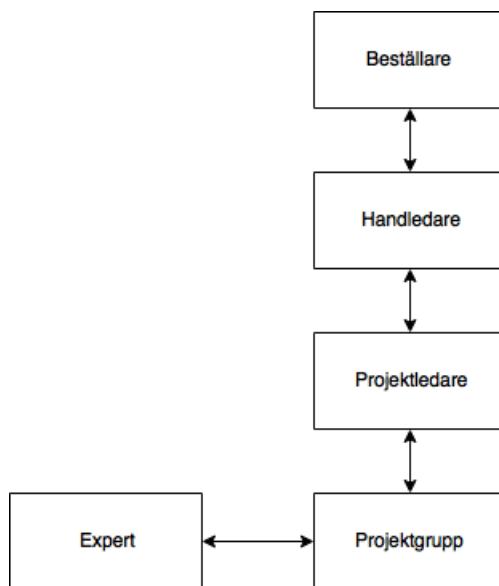
Under efter-fasen överförs projektresultatet till beställaren och projektet avslutas. Projektgruppen skriver en *slutrapport* som bifogas vid slutleveransen av produkten. Vid slutleveransen ska även en demonstration hållas som visar att produkten uppfyller kraven ställda på den. Efter produkten levereras skriver projektgruppen en *etterstudie*.

4 Organisationsplan för hela projektet

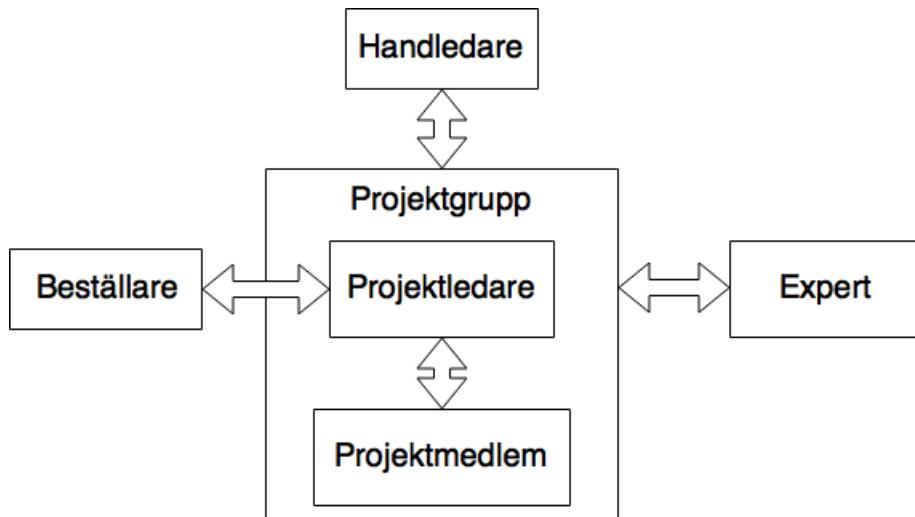
Detta avsnitt behandlar hur organisationen för projektet ser ut.

4.1 Organisationsplan

Figur 1 visar en schematisk bild över projektets hierarki. Figur 2 visar kontakten mellan de inblandade parterna. Beställaren har kontakt med projektledaren som i sin tur har kontakt med projektets medlemmarna. Både handledaren och experten har kontakt med projektgruppen.



Figur 1: Projektets organisation



Figur 2: Projektets organisation

4.2 Villkor för samarbetet inom projektgruppen

Projektgruppen har skrivit ett gruppkontrakt som hittas i Bilaga A.

4.3 Definition av arbetsinnehåll och ansvar

Anders Rehult är projektledare och har den huvudsakliga kontakten med beställaren. Marian Brännvall är dokumentansvarig och har ansvar för att färdigställda dokument finns i grupprummet på Lisam. Andreas Kempe är sekreterare och Viktor Bernholtz är gruppmedlem.

5 Dokumentplan

Detta avsnitt behandlar de dokument som ingår i projektet. Dokumenten lagras dels på Lisam och dels sparar dokumentansvarig dem på sin dator. Utskrivna dokument sparas som fysiska exemplar av dokumentansvarig.

Dokumenten skrivs på svenska eller engelska.

Samtliga gruppmedlemmar har tillgång och möjlighet att redigera samtliga dokument med undantag för den sammanställda tidrapporten som endast projektledare redigerar.

Samtliga dokument ska ha en dokumenthistorik där versionsnummer, datum för förändring, utförd förändring, ansvarig för förändring samt granskare finns angiven. Versionsnummer är 0.x innan dokumentet är godkänt och godkänt dokument får versions nummer 1.0. Förändringar efter godkännande får alltså nummer 1.x.

Dokument	Ansvarig/godkänns av	Syfte	Distribueras till	Färdig datum
Gruppkontrakt	Projektgrupp	Sätta upp villkor för sammarbetet i gruppen.	Projektgrupp	2018-01-25
Krav-specifikation	Projektgrupp/beställare och handledare	Specificera krav för projektet.	Projektgrupp, beställare och handledare	2018-01-30
Systemskiss	Projektgrupp/beställare och handledare	Beskriva de olika delsystemen som systemet består av.	Projektgrupp, beställare och handledare	2018-02-16
Projektplan	Projektgrupp/beställare och handledare	Beskriva hur projektet ska utföras, tidsåtgång och ansvarsområden.	Projektrupp, beställare och handledare	2018-02-16
Tidsplan	Projektgrupp/beställare och handledare	Planera olika aktiviteter och tidsåtgång för dessa.	Projektgrupp, beställare och handledare	2018-02-16
Design-specifikation	Projektgrupp/handledare	Specificera designen för systemet utifrån systemskiss och kravspecifikation.	Projektgrupp och handledare	2018-03-08
Teknisk-dokumentation	Projektgrupp/beställare och handledare	Beskrivning av hur systemet/produkten är konstruerad.	Projektgrupp, beställare och handledare	2018-05-28
Slutrapport	Projektgrupp/beställare och handledare	Sammanställa arbetet i en rapport.	Projektgrupp, beställare och handledare	2018-05-28
Teknisk/vetenskaplig rapport	Projektgrupp/beställare och handledare	Två rapporter, en på Fermi-ytor och en på volymrendering.	Projektgrupp, beställare och handledare	2018-05-28
Användar-handledning	Projektgrupp/beställare och handledare	Beskriva hur systemet används.	Projektgrupp, beställare och handledare	2018-05-28
Efterstudie	Projektgrupp/beställare och handledare	Samla erfarenheter från projektet	Projektgrupp, beställare och handledare	2018-06-05
Mötesprotokoll	Projektgrupp	Dokumentation från möten under projektets gång.	Projektgrupp	Löpande

6 Utvecklingsmetodik

Projektet kommer genomföras i steg enligt *tidsplanen* (Bilaga E). Enhetstester kommer, i största möjliga mån, löpande skapas för den kod projektgruppen skriver.

7 Utbildningsplan

Detta avsnitt behandlar de utbildningar personer i projektet kommer genomgå.

7.1 Egen utbildning

Projektgruppen erhåller utbildning från universitet i form av föreläsningar och laborationer. Nedan finns en lista över de utbildningsmoment som skall utföras:

- Introduktionsföreläsning med utdelning av projektdirektiv och information om LIPS-projektmodellen.
- Föreläsning om valt projekt.
- Tre stycken etikföreläsningar.
- Två etikseminarier.
- Föreläsning om LIPS-modellens kravspecifikation.
- Föreläsning om simuleringsmjukvaran Inviwo.
- Föreläsning om materialfysik.
- Inspirationsföreläsning om solceller.
- Föreläsning LIPS-modellens tidsplanering.
- Föreläsning om Python-programmering.
- Python-laboration.
- Laboration i beräkningsfysik.
- Föreläsning om beräkningsfysik.
- Föreläsning om LIPS-modellens designspecifikation.
- Seminarium om rapportgranskning.
- Föreläsning om sluttolkning enligt LIPS-modellen.

7.2 Kundens utbildning

Vid en demonstration kommer kunden få utbildning i den färdiga produkten. Dessutom ska en användarhandledning framställas.

8 Rapporteringsplan

All tidrapportering sker till projektledaren som i sin tur för in det i tidsplanen. Projektledaren rapporterar sedan till beställaren inför varje beslutspunkt. Statusrapportering sker kontinuerligt inom projektgruppen vid projektmöten. Statusrapport skickas till beställaren på begäran.

9 Mötesplan

Projektmöten bokas gemensamt av gruppen och hålls minst en gång i veckan. Sekreteraren väljs på plats och ser till att alla gruppmedlemmar tillhandahålls mötesprotokoll.

10 Resursplan

Detta avsnitt behandlar de resurser projektgruppen har tillgång till. Vilka personer som är inblandade, lokaler och utrustning samt hur ekonomin för projektet ser ut.

10.1 Personer

Rickard Armiento är beställare/expert i detta projekt med Johan Jönsson som handledare och Anders Rehult som projektledare. Övriga gruppmedlemmar är Viktor Bernholtz, Andreas Kempe och Marian Brännvall. Projektgruppen kommer i största möjliga mån arbeta under ordinarie kontorstid. Expert/beställare och/eller handledare kan nås via mail eller på deras kontor under ordinarie kontorstid.

10.2 Material

Projektet kräver tillgång till programvaran Inviwo. Två datorer med programvaran finns till projektgruppens förfogande.

10.3 Lokaler

Projektgruppen har dygnet runt tillgång till lokalen Saturnus, H 303 i Fysikhuset på Campus Valla vid Linköpings universitet. I lokalen finns två datorer som enbart får användas av projektgruppen.

10.4 Ekonomi

Projektgruppen har totalt 1 000 timmar för att slutföra projektet, dessa är fördelade jämnt över gruppens medlemmar.

11 Milstolpar och beslutspunkter

Detta kapitel specificerar projektets milstolpar och beslutspunkter. Milstolparna är delmål för projektet som ska vara mätbara.

11.1 Milstolpar

Milstolparna är numrerade, har en kort beskrivning samt datum då milstolpen ska vara nådd.

Nr	Beskrivning	Datum
1.	Designspecifikationen godkänd och klar.	2018-03-08
2.	Kunna konvertera kristallstrukturer i VASP-format till HDF5-format.	vecka 10
3.	Kunna konvertera elektronräntähet i VASP-format till HDF5-format.	vecka 16
4.	Kunna konvertera tillståndstäthet VASP-format till HDF5-format.	vecka 13
5.	Kunna konvertera Fermi-ytor i VASP-format till HDF5-format.	vecka 17
6.	Kunna konvertera data från Elkberäkningar till HDF5-format	vecka 18
7.	Kunna visualisera en atom och justera färg- och transparensinställningar.	vecka 10
8.	Kunna visualisera kristallstrukturer.	vecka 13
9.	Kunna visualisera elektronräntähet.	vecka 17
10.	Kunna visualisera tillståndstäthet.	vecka 15
11.	Kunna justera färg- och transparensinställningar för kristallstrukturer.	vecka 16
12.	Kunna justera färg- och transparensinställningar för elektronräntähet.	vecka 16
13.	Kunna justera färginställningar för tillståndstäthet.	vecka 16
14.	Kunna visualisera Fermi-ytor.	vecka 20
15.	Kunna justera färg- och transparensinställningar för Fermi-ytor	vecka 20

11.2 Beslutspunkter

Beslutspunkterna är numrerade, har en kort beskrivning samt datum för beslutpunktsmöte.

Nr	Beskrivning	Datum
0.	Projektgruppen formad och projektuppgift vald.	2018-01-22
1.	Godkännande av kravspecifikation.	2018-01-30
2.	Godkännande av systemskiss, projektplan och tidsplan.	2018-02-16
3.	Godkännande av designspecifikation.	2018-03-08
4.	Presentation och godkännande av nuvarande design.	2018-04-11
5.	Verifiering av kraven.	2018-05-28

12 Aktiviteter

Detta avsnitt behandlar de aktiviteter som kommer genomföras under projektets gång. Aktiviteter utan angiven tidsåtgång har ägt rum innan BP2 och räknas därför inte med i de 1 000 timmar som projektet har att tillgå.

12.1 Dokumentation

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
1.	Designspecifikation	Designspecifikationen är ett dokument som beskriver den fullständiga designen av leveransen, produkten, programvaran och all teknisk dokumentation. Designspecifikationen är ett förtydligande av systemskissen och kravspecifikationen.	70	
2.	Slutrapport/kappa	Slutrapporten ska ge en komplett överblick över hela det genomförda arbetet och inkludera följande delar: <ul style="list-style-type: none"> • Problemformulering • Kunskapsbas • Genomförande • Resultat och slutsatser En färdig kappa för slutrapporten ska skickas till TEMA.	30	
3.	Teknisk dokumentation	Den tekniska dokumentationen ska innehålla en beskrivning av hur produkten är konstruerad.	60	
4.	Efterstudie	En efterstudie med projektmedlemmarnas samlade erfarenheter från projektet.	8	
5.	Renskrivning	Renskrivning av protokoll	8	
6.	Manual	En användarhandledning ska framställas.	12	

12.2 Möte

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
7.	Projektmöte	Projektgruppen har möte minst en gång i veckan.	32	
8.	BP3	Designspecifikationen ska vara godkänd av handledaren.	4	1
9.	BP4	Nuvarande design ska vara presenteras och godkänd av handledaren.	4	
10.	BP5	Kraven ska vara verifierade.	8	

12.3 Utbildning

Kursens föreläsningar ingår inte i timmarna räknade till projektet.

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
11.	Föreläsning	Introduktionsföreläsning med utdelning av projektdirektiv och information om LIPS-projektmallen.		
12.	Föreläsning	Föreläsningar om etik.		
13.	Föreläsning	Föreläsningar om etik.		
14.	Föreläsning	Föreläsningar om valt projekt.		
15.	Föreläsning	Föreläsningar om etik.		
16.	Föreläsning	Föreläsning om LIPS-modellen planering.		
17.	Föreläsning	Föreläsning om LIPS-modellens kravspecifikation.		
18.	Föreläsning	Föreläsning om simuleringsmjukvaran Inviwo.		
19.	Föreläsning	Föreläsning om materialfysik.		
20.	Föreläsning	Inspirationsföreläsning om solceller.		
21.	Föreläsning	Föreläsning om LIPS-modellens designspecifikation.		
22.	Föreläsning	Föreläsning om python.		
23.	Laboration	Laboration i python.		
24.	Föreläsning	Föreläsning om beräkningsfysik.		
25.	Laboration	Laboration i beräkningsfysik.		
26.	Föreläsning	Föreläsning om slutdokumentation enligt LIPS-modellen.		

12.4 Presentation

Etikseminarierna ingår inte i timmarna räknade till projektet.

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
27.	Seminarium	Seminarium om etik.		
28.	Seminarium	Seminarium om etik.		
29.	Seminarium	Seminarium om rapportgranskning.		
30.	Slutpresentation	En presentation av projektet ska förberedas och hållas för beställaren.	20	
31.	Opponering	Projektgruppen ska förbereda och utföra en opponering på en annan grups arbete.	20	2, 3

12.5 Delsystem datakonvertering

Detta dokument skiljer inte på att projektgruppen implementerar lösningar och att projektgruppen sätter sig in i redan tillgängliga lösningar från föregående års projektgrupp.

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
32.	Förstå utdata från VASP-beräkningar: kristallstruktur	Projektgruppen behöver förstå hur kristallstrukturer definieras i VASP.	8	
33.	Förstå utdata från VASP-beräkningar: elektrontäthet	Projektgruppen behöver förstå hur elektrontäthet presenteras i VASP.	8	
34.	Förstå utdata från VASP-beräkningar: tillståndstäthet	Projektgruppen behöver förstå hur tillståndstäthet presenteras i VASP.	8	
35.	Förstå utdata från VASP-beräkningar: Fermi-ytor	Projektgruppen behöver förstå hur energi-eigenvärden presenteras i VASP.	8	
36.	Konvertera kristallstrukturdata från VASP-format till HDF5-format.	Kristallstrukturdata behöver läsas in och konverteras till HDF5-format så att systemet kan behandla dem.	16	32
37.	Konvertera elektrontäthetsdata från VASP-format till HDF5-format.	Elektrontäthetsdata behöver läsas in och konverteras till HDF5-format så att systemet kan behandla dem.	16	33
38.	Konvertera tillståndstäthetsdata från VASP-format till HDF5-format.	Tillståndstäthetsdata behöver läsas in och konverteras till HDF5-format så att systemet kan behandla dem.	16	34
39.	Konvertera data från VASP-beräkningar av Fermi-ytor till HDF5-format.	Data från beräkningar av Fermi-ytor behöver läsas in och konverteras till HDF5-format så att systemet kan behandla dem.	16	35
40.	Konvertera data från Elk-beräkningar till HDF5-format	Data behöver läsas in och konverteras till HDF5-format så att systemet kan behandla dem.	12	32, 33, 34, 35

12.6 Delsystem visualisering

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
41.	Grundläggande förståelse av Inviwo	Projektgruppen behöver förstå sig på hur Inviwo används, och till viss mån, fungerar.	16	
42.	Grundläggande visualisering av atom	Visualisera en atom genom att rita ut en sfär.	4	41
43.	Transparens-inställning	Transparensen av allting som ritas ut i 3D-grafer ska kunna ändras med avseende på någon egenskap. I det första grundläggande fallet, utritande av en atom, behöver transparensen av atomen kunna ställas in godtyckligt, då inga andra egenskaper än dess form ritas ut.	2	42
44.	Färginställning för atom	Färgen av atomen ska kunna ställas in godtyckligt.	2	42
45.	Rotation	3D-grafer ska kunna roteras.	2	41
46.	Visualisering av kristallstruktur	Visualiseringsmodulen behöver kunna använda sig av data skickat från datakonverteringsmodulen för att rita upp kristallstrukturer, alltså illustrera hur atomerna i kristallens enhetscell är placerade i förhållande till varandra.	20	41, 36
47.	Val av elektrontäthetsvisualiseringssätt	Projektgruppen behöver lära sig vad elektrontäthet är och komma överens om hur det ska visualiseras.	8	
48.	Visualisering av elektrontäthet	Sannolikheten att hitta elektroner i diverse områden i kristallens enhetscell ska kunna illustreras.	16	41, 37
49.	Transparens-inställning, brytpunkt och intervall	En brytpunkt ska kunna väljas för vilken full transparens inträder för det grafiska objekt som representerar elektrontätheten. Full transparens ska även kunna fås inom ett intervall. Transparensnivån ska, utöver detta, kunna ändras för valfria värden.	4	48

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
50.	Färginställning, flera saker som ska färgläggas	Atomerna ska kunna ha en färg, de olika värdena på elektrontäthet en eller flera andra.	4	48
51.	Sätta sig in i tillståndstäthet	Projektgruppen behöver sätta sig in i tillståndstäthet.	8	
52.	Tillståndstäthet	Tillståndstäthet ska illustreras i en 2D-graf med energi på x-axeln och antal tillstånd på y-axeln.	32	38
53.	Val av visualiseringssätt för Fermi-ytor	Projektgruppen behöver komma överens om hur Fermi-ytor ska visualiseras.	8	
54.	Fermi-ytor	Fermi-ya, eller ytan för godtycklig energinivå under Fermi-energin, ska ritas ut i k-rummet.	80	39

12.7 Tester

Testerna är inte automatiserade tester utan görs manuellt efter att en uppgift är slutförd.

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
55.	Testa konvertera data från VASP-beräkningar	För att säkerställa att konvertering till HDF5-format från VASP-beräkningar fungerar görs tester av detta.	2	36, 37, 38, 39
56.	Testa konvertera data från Elk-beräkningar	För att säkerställa att konvertering till HDF5-format från Elk-beräkningar fungerar görs tester av detta.	2	40
57.	Testa visualisering av atom	Atomen ska kunna visualiseras och inställningar för färg och transparens ska vara möjliga.	2	42
58.	Testa visualisering av kristallstruktur	Kristallstrukturer ska kunna visualiseras och inställningar för färg och transparens ska vara möjligt samt rotation.	2	46
59.	Testa visualisering av elektrontäthet	Elektrontäthet ska kunna visualiseras och inställningar för färg och transparens ska vara möjligt samt rotation.	2	48
60.	Testa visualisering av tillståndstäthet	Tillståndstäthet ska kunna visualiseras och inställningar för färg ska vara möjlig.	2	52
61.	Testa visualisering av Fermi-ytor	Fermi-ytor ska kunna visualiseras och inställningar för färg och transparens ska vara möjligt samt rotation.	2	54

12.8 Övrigt

Nr	Aktivitet	Beskrivning	Beräknad tid (h)	Beroende aktivitet nr
62.	Kodstädning	Snygga till kod och kommentarer.	24	
63.	Beskrivningar	Skriv sammanfattande beskrivningar till alla processorer och python-moduler.	8	
64.	Förståelse av legacy-kod	Projektgruppen behöver förstå föregående års kod för att veta vad som kan återanvändas och byggas vidare på.	40	
65.	Uppdatering av legacy-kod	De delar av föregående års kod som ska användas i årets projekt behöver med största sannolikhet uppdateras för att bli kompatibel med aktuell version av Inviwo.	80	64
66.	Fördjupningsarbete	Projektgruppen ska i grupper om två skriva ett fördjupningsarbete relaterat till projektet.		
67.	Buffert	Antal timmar av de ursprungliga, 1 000 timmarna, projektgruppen har kvar att tillgå.	244	

13 Tidsplan

Se Bilaga E.

14 Förändringsplan

Vid eventuella förändringar i projektet kommer projektgruppen möjligtvis vara tvungen att ändra i tidsplan, milstolpslista, aktivitetslista eller kravspecifikation. Om krav behöver omförhandlas bokar gruppen ett möte med beställare.

15 Kvalitetsplan

I detta avsnitt beskrivs hur kvaliteten på leveranser ska kontrolleras med hjälp av granskningar och tester.

15.1 Granskningar

Samtliga dokument som framställs kommer att granskas av någon i projektgruppen och detta anges även på det granskade dokumentet.

15.2 Testplan

Tester av systemet kommer att skrivas parallellt med att kod skrivs och utföras kontinuerligt.

16 Riskanalys

Inga större risker har identifierats med projektet.

17 Prioriteringar

Kraven i kravspecifikationen är av typen ska-krav, bör-krav eller kanske-krav. Ska-kraven kommer att prioriteras vid förseningar och tidsbrist för att säkerställa att projektet slutförs vid sluttid.

18 Projektavslut

I en efterstudie ska projektmedlemmarna samla erfarenheterna från projektarbetet. All utrustning och lånat material ska återlämnas. Dokumentationen förvaras i grupprummet på Lisam så att samtliga projektmedlemmar har åtkomst till den.

I Systemskiss

Systemskiss

Redaktör: Andreas Kempe

Version 1.0

Status

Granskad	Viktor Bernholtz	2018-02-16
Godkänd	Johan Jönsson	2018-02-20

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	iv
1 Inledning	1
1.1 Syfte och Mål	1
2 Översikt av systemet	1
2.1 Ingående delsystem	2
3 Delsystem datakonvertering	3
3.1 Inläsning	3
3.2 Konvertering	3
4 Delsystem visualisering	4
4.1 HDF5-inläsning	4
4.2 Användarindata	5
4.3 Utritning	6
4.4 Interaktivitet	6
Referenser	6

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-02-07	Första utkast.	Projektgruppen	VB
0.2	2018-02-08	Andra utkast.	Projektgruppen	AK
0.3	2018-02-09	Tredje utkast.	Projektgruppen	VB
1.0	2018-02-16	Godkänd version.	Projektgruppen	VB

1 Inledning

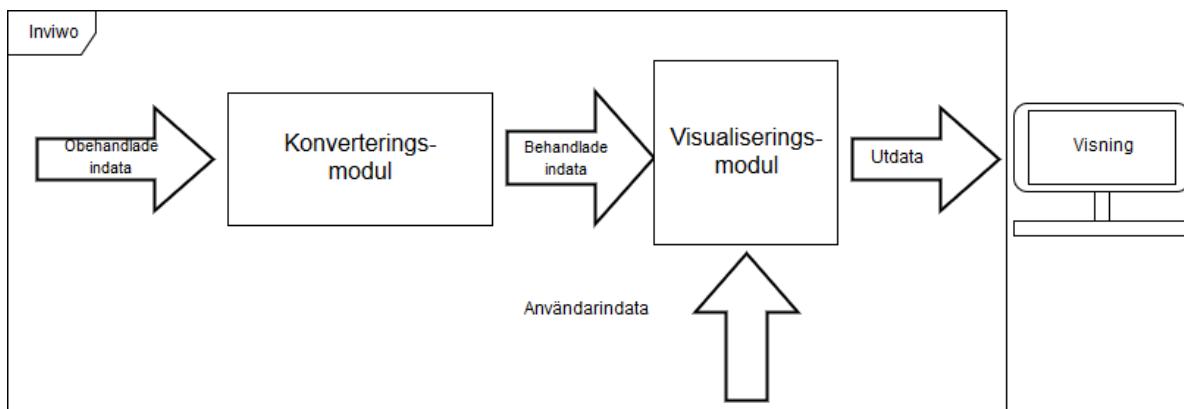
Detta dokument ämnar ge en bild av ett system för visualisering av elektronstrukturdata som erhålls från kvantmekaniska beräkningar. Systemet byggs kring mjukvaran Inviwo som utvecklas av forskargruppen Scientific Visualization vid Linköpings universitet för att underlätta visualisering av olika sorters data.

Projektet utförs för IFM (Institutionen för Fysik, Kemi och Biologi) vid Linköpings universitet som ett kandidatprojekt.

1.1 Syfte och Mål

Projektet skall ta fram mjukvara för visualisering av data från kvantmekaniska kristallstrukturberäkningar. Detta då det är svårt att dra slutsatser utifrån rådata utan verktyg som underlättar i analysen av densamma.

2 Översikt av systemet



Figur 1: Grov skiss av systemet

Systemet som helhet ska ta in data från VASP- och Elk-beräkningar, konvertera detta till HDF5-format och sedan visualisera dessa resultat med hjälp av visualiseringssverktyget Inviwo. Användaren ska kunna välja vad som ska visualiseras och, när bilden renderats, även kunna modifiera den utritade bilden. I Figur 1 visas detta flöde och rutorna representerar en aktivitet i systemet, medan pilarna representerar dataflödet genom systemet.

Systemet ska kunna:

- Läsa in data från elektronstrukturberäkningar gjorda i VASP och Elk.
- Konvertera inkommende obehandlad data från textfil till HDF5-format.
- Ta in användaridata som avgör vad som ska visualiseras.
- Visualisera behandlad data utifrån inskickad användaridata.
- Hantera interaktivitet dvs. användaren ska kunna modifiera den renderade bilden i efterhand.

2.1 Ingående delsystem

Systemet består av två övergripande delsystem. Dessa är datakonvertering och visualisering.

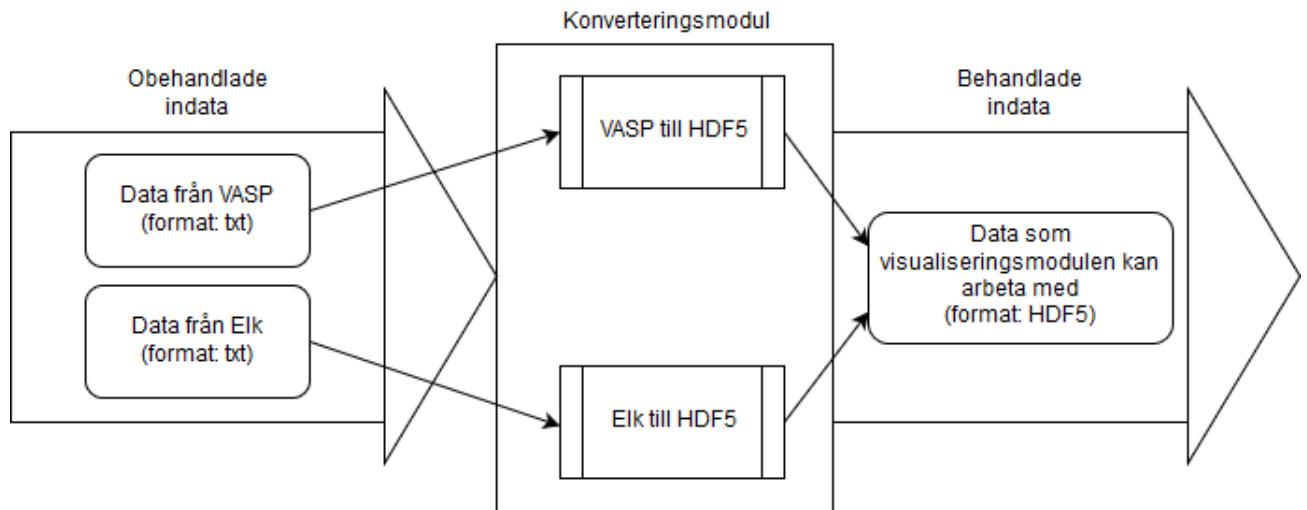
Datakonverteringen hanterar inkommande, obehandlad data och skickar ut data konverterad till HDF5-format. HDF5 är ett dataformat som lämpar sig bättre att hanteras av Inviwo än originaldatan som kommer i textfiler. Detta då det är designat att hantera stora datamängder på ett flexibelt sätt och tillhandahåller programmatiska gränssnitt för interaktion med datan [1].

Visualiseringen skapar en grafisk representation av indata i form av 3D- eller 2D-grafer som sedan visas för användaren. Modulen tillåter sedan manipulation av bilden i form av rotation, skalning, ändrande av färgläggning och så vidare.

I avsnitten för datakonvertering och visualisering beskrivs delsystemen och deras funktioner mer ingående.

3 Delsystem datakonvertering

Figur 2 beskriver hur delsystemet för datakonvertering är uppbyggt. Obehandlad indata representeras av en stor pil och rutorna inuti pilen representerar textfiler med data från beräkningar gjorda i beräkningsprogrammen VASP och Elk. Konverteringsmodulen representeras av en ruta i mitten och rutorna inuti är konverteringen för data från respektive beräkningsprogram. Slutligen skickas konverterad data ut vilket representeras av en stor pil.



Figur 2: Skiss av datakonverteringsmodulen

I detta delsystem läses resultat från VASP- eller Elk-beräkningar in, i form av textfiler, och konverteras till det binära filformatet HDF5. Dessa data kan komma från beräkningar på kristallstrukturer, elektronstrukturer eller tillståndstäthet.

3.1 Inläsning

Modulen ska kunna läsa in textfiler som innehåller resultatet av beräkningar gjorda i VASP eller Elk. Denna data ska sedan konverteras för att ytterligare kunna behandlas i Inviwo.

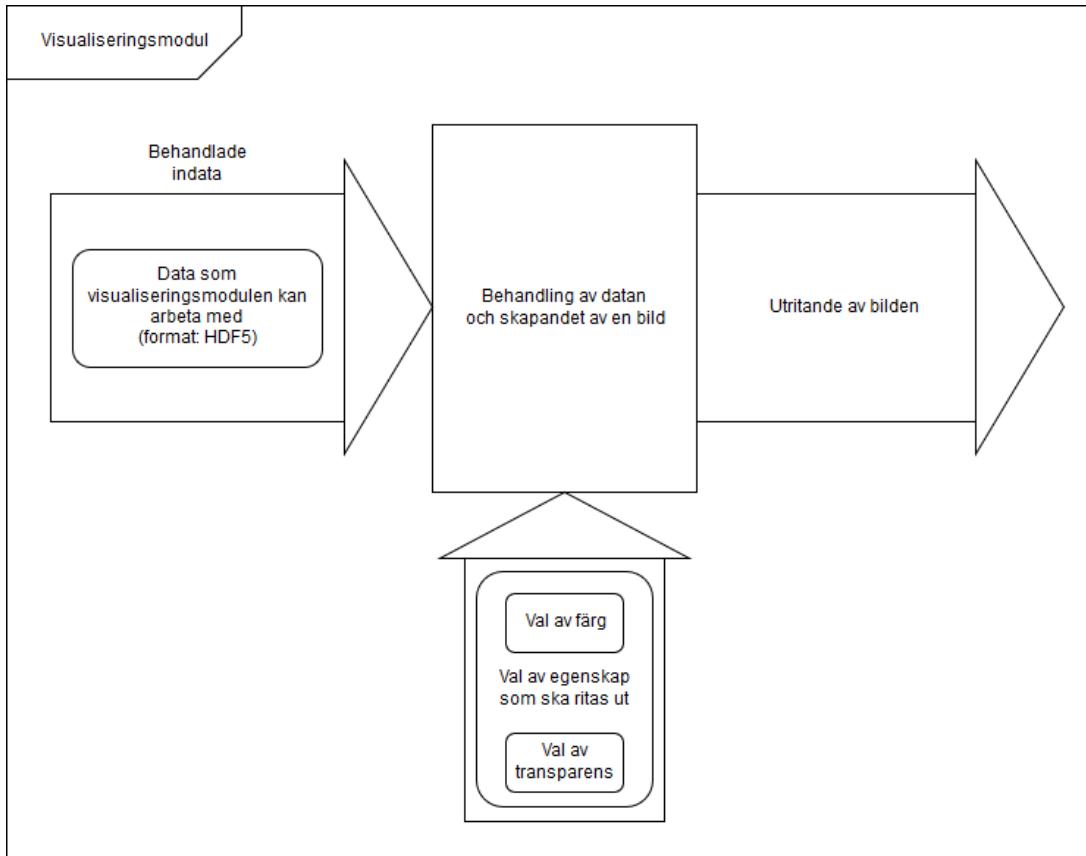
Filerna som läses in kan vara för stora för att rymmas i datorns primärminne. I sådana fall måste de hanteras genom att den relevanta datan strömmas från disk eller att ett subset av datan väljas ut och visualiseringen begränsas till att bara rita ut en del av modellen i taget.

3.2 Konvertering

Data ska läsas in och konverteras till HDF5-datastrukturer som tack vare tidigare års arbete redan finns tillgängliga.

4 Delsystem visualisering

Figur 3 beskriver hur delsystemet för visualisering är uppbyggt. Behandlad data skickas in, vilket i figur 3 representeras av en stor pil. Dessutom skickas användarindata in vilket också beskrivs med en stor pil, användarindata består av val av färg, val av egenskap som ska ritas ut, val av transparens etc. Dessa behandlas sedan i visualiseringsmodulen som skapar en bild utifrån dem. Den utritade bilden skickas sedan ut från modulen.



Figur 3: Skiss av visualiseringsmodulen

I detta delsystem visualiseras den behandlade datan. Visualiseringsdelsystemet består av flera moduler för olika typer av visualisering, exempelvis visualisering av:

- total tillståndstätthet
- Fermi-ytor

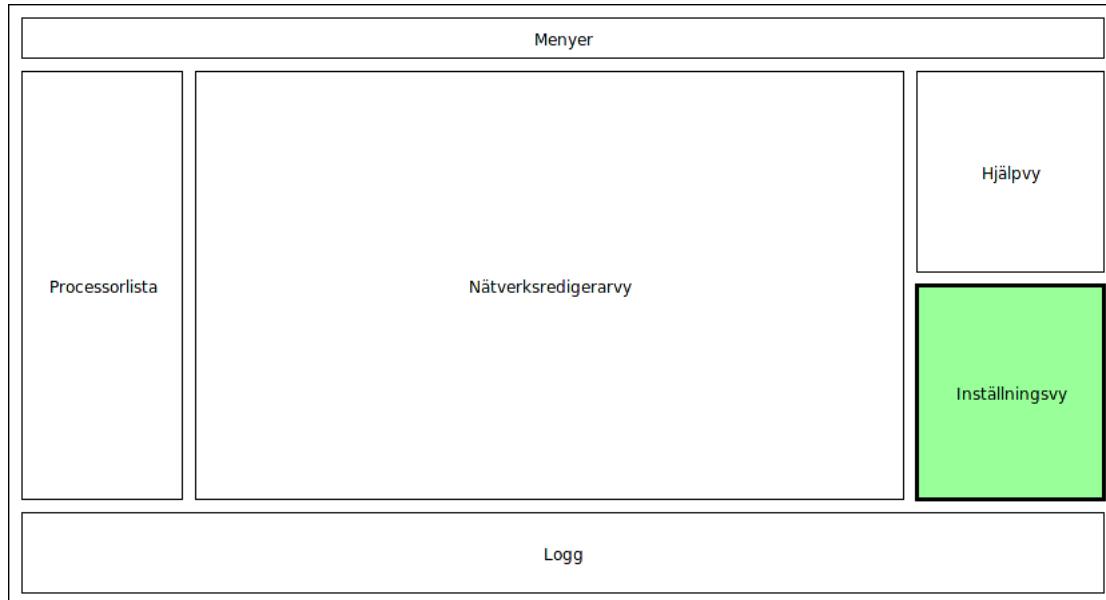
För att visualisera vissa typer av egenskaper kommer kristallstrukturer behöva visualiseras. Kristallerna ritas upp genom att de enskilda atomerna i en enhetscell visualiseras som en liten sfär. Samtliga visualiseringsmoduler implementeras i Inviwo.

4.1 HDF5-inläsning

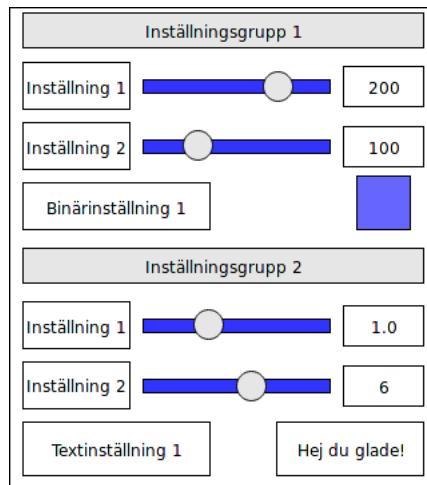
Systemet ska läsa in data på HDF5-format som skickats ut av datakonverteringsmodulen. I Inviwo version 0.9.9 finns en modul som kan importera data i HDF5-format [2], detta kan vara ett möjligt sätt att utföra HDF5-inläsning.

4.2 Användaridata

Användaren ska kunna bestämma vilka egenskaper som ska visualiseras. Detta görs genom enkla funktionsanrop eller genom val i ett användargränssnitt.



Figur 4: Skiss av användargränssnittet i Inviwo.



Figur 5: Skiss av inställningsvyn i Inviwo.

Figur 4 är en skiss över användargränssnittet för Inviwo. Den fetmarkerade rutan med grön bakgrund är vyn där inställningar görs. I Figur 5 är inställningsvyn skissad mer detaljerat. Numeriska värden kan ställas in genom direkt inmatning i värderutorna till höger, eller genom dragreglagen. En binär inställning ställs in genom att rutan för inställningen antingen klickas så att den är ifylld, inställningen på, eller så att den är tom, inställningen av. Slutligen har vi textinställningar som ställs in genom att texten skrivas in direkt i värderutan.

4.3 Utritning

Systemet ska rita ut bilder utefter den behandlade data. Utritningen skall ge en visualisering av den egenskap som modulen behandlar och kan till exempel vara en volymrendering eller en vanlig 2D-graf, beroende på vilket som är mest lättolkat.

Utritningen görs via Inviwos inbyggda funktionalitet för att rendera via OpenGL. Grafisk behandling görs också med Inviwo och OpenGL, där möjligheten finns att både använda färdig funktionalitet och att utveckla nya renderingsfunktioner.

4.4 Interaktivitet

Användaren ska kunna modifiera visualiseringen genom att reglera ett intervall av värden för någon egenskap, där full transparens fås för alla värden inom intervallet, rotera 3D-bilder etc.

Ny användaridata, som skickas in efter att den första bilden har ritats upp, skickas tillbaka till visualiseringsmodulen för att utföra en ny rendering.

Referenser

- [1] The HDF Group, (2017). *What is HDF5?* [online] Tillgänglig på: <https://support.hdfgroup.org/HDF5/whatishdf5.html> [Hämtad 2018-02-08].
- [2] Inviwo, (2017). *Inviwo 0.9.9 Released* [online] Tillgänglig på: <http://www.inviwo.org/2017/10/04/inviwo-0-9-9-released/> [Hämtad 2018-02-07].

Tidplan

Basplan

Projekt:

Projektgrupp: Visualisering av elektronstrukturer
 Beställare: Rickard Armiento
 Kurs: TFY475

Datum: 2018-01-16
 Version: 0.3
 Utfördare: PG

Granskad:
 Marian Brännvall

AKTIVITETER		TID	VEM	TIDPLAN (när, veckonummer)																									
Nr	Beskrivning	timmar	Initialer	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30			
1	Skriva designspecifikation.	70		50	20																							70	
2	Skriva slutrappart/kappa.	30																										30	
3	Skriva teknisk dokumentation.	60																12	6	20	18	4						60	
4	Göra en efterstudie.	8																										8	
5	Renskrivning av protokoll.	8			1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	8	
6	Skriva en användarmanual.	12																										12	
7	Projektmötet	32		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	32		
8	Beslutspunkt 3.	4																4										4	
9	Beslutspunkt 4.	4																	4									4	
10	Beslutspunkt 5.	8																										8	
11																												0	
12																												0	
13																												0	
14																												0	
15																												0	
16																												0	
17																												0	
18																												0	
19																												0	
20																												0	
21																												0	
22																												0	
23																												0	
24																												0	
25																												0	
26																												0	
27																												0	
28																												0	
29																												0	
30	Slutpresentation	20																										20	
31	Oppnering	20																										20	
32	Förstå utdata från VASP-beräkningar: kristallstruktur	8															8											8	
33	Förstå utdata från VASP-beräkningar: elektronräntähet	8																	8									8	
34	Förstå utdata från VASP-beräkningar: tillståndstäthet	8																		8									8
35	Förstå utdata från VASP-beräkningar: Fermi-ytor	8																											8
36	Konvertera VASP-data, kristallstruktur	16														8	8												16
37	Konvertera VASP-data, elektronräntähet	16																											16
38	Konvertera VASP-data, tillståndstäthet	16																	8	8								16	
39	Konvertera VASP-data, Fermi-ytor	16																											16
40	Konvertera data från Elk-beräkningar till HDF5-format	12																											12
41	Grundläggande förståelse för Inviwo	16													8	8													16
42	Grundläggande visualisering för väteatom	4															4												4
43	Transparensinställning för väteatom	2																2											2
44	Färginställning för väteatom	2																	2										2
45	Rotera 3D-grafer	2																	2										2
46	Visualisera kristallstrukturer	20																	10	10									20
47	Välja elektronräntäthsvisualiseringssätt	8																											8
48	Visualisera elektronräntähet	16																											16
49	Transparensinställning, brytpunkt och intervall	4																											4
50	Fler färginställningar	4																											4
51	Sätta sig in i tillståndstäthet	8																	8										8
52	Visualisera tillståndstäthet	32																		16	16								32
53	Välja visualiseringssätt för Fermi-ytor	8																											8
54	Visualisera Fermi-ytor	80																											80
55	Manuell test av konvertering från VASP-data till HDF5	2																											2
56	Manuell test av konvertering från Elk-data till HDF5	2																											2
57	Manuell test av visualisering av väteatomkäma	2																											2
58	Manuell test av visualisering av kristallstruktur	2																											2
59	Manuell test av visualisering av elektronräntähet	2																											2
60	Manuell test av visualisering av tillståndstäthet	2																											2
61	Manuell test av visualisering av Fermi-ytor	2																											2
62	Kodstädning	24																											24
63	Skriva beskrivningar av processorer och python-moduler	8																											8
64	Förståelse av legacy-kod	40														20	20											40	
65	Uppdatering av legacy-kod	80																	16	16	16	16						80	
66	Skriva fördjupningsarbete																												0
	MILSTOLPAR																												0
1	Designspecifikation klar																												0
2	Konvertera VASP-data kristallstrukturer																												0
3	Konvertera VASP-data elektronräntähet																												0
4	Konvertera VASP-data tillståndstäthet																												0
5	Konvertera VASP-data Fermi-ytor																												0
6	Konvertera Elk-data																												0
7	Visualisera väteatomkäma																												0
8	Visualisera kristallstrukturer																												0
9	Visualisera tillståndstäthet																												0
10	Färg-och transparensinställningar, kristallstrukturer																												0
11	Färg-och transparensinställningar, elektronräntähet																												0
12	Färg-och transparensinställningar, tillståndstäthet																												0
13	Färg-och transparensinställningar, Fermi-ytor																												0
14	Visualisera elektronräntähet																												

Tidsbuffert:

244

K Teknisk dokumentation**Teknisk dokumentation**

Redaktör: Anders Rehult

Version 0.2

Denna tekniska dokumentation är baserad på 2017 års tekniska dokumentation för visualiseringssertyget ENVISIoN.

Status

Granskad	PL	18-05-25
Godkänd		

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	v
1 Inledning	1
1.1 Parter	1
1.2 Syfte och mål	1
1.3 Användning	1
1.4 Begränsningar	1
1.5 Definitioner	2
2 Produkten	3
3 Översikt av systemet	4
3.1 Resurser	4
3.2 Ingående delsystem	4
3.3 Utvecklingsfilosofi	5
4 Delsystem 1 - System för parsning	5
4.1 VASP	5
4.2 HDF5-struktur för VASP-data	5
4.3 Modul för skrivning till HDF5	6
4.4 Moduler för parsning	8
4.4.1 Incarparser	9
4.4.2 Volymparser	9
4.4.3 Tillståndstäthetsparser	10
4.4.4 Enhetscellsparser	10
4.4.5 parse_all	11
5 Delsystem 2 - System för visualisering	11
5.1 Nätverk	11
5.1.1 Nätverk för visualisering av volymdata	11
5.1.2 Nätverk för visualisering av enhetscell	17
5.1.3 Nätverk för visualisering av DOS	18
5.1.4 Sammankoppling av nätverk	20
5.1.5 Färg-och-transparensinställning	21
5.2 Datastrukturer	22
5.2.1 Point	22

5.2.2	Function	22
5.3	Processorer	22
5.3.1	Kristallstruktur	22
5.3.2	HDF5	23
5.3.3	2D	26
5.4	Properties och widgets	28
5.4.1	IntVectorproperty	28
5.4.2	IntVectorPropertyWidget	28
6	Utvecklingsmöjligheter	29
Referenser		30
Bilaga A	HDF5-datastruktur	31
Bilaga B	HDF5-datastruktur i två delar	32

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-05-18	Första utkast.	Projektgruppen	PL
0.1	2018-05-25	Andra utkast (figur 11 uppdaterad).	Projektgruppen	PL

1 Inledning

Dokumentet är en teknisk dokumentation för kandidatprojektet i visualisering av elektronstrukturer. Visualisering av elektronstrukturer är ett av projekten i kursen TFYA75 vid Linköpings universitet. Den tekniska dokumentationen beskriver detaljerat hur systemet är implementerat.

1.1 Parter

Rickard Armiento har beställt systemet som är beskrivet i denna tekniska dokumentation. Medlemmarna i projektgruppen, som är listade under rubriken projektidentitet ovan, är mottagare av denna beställning och har haft i uppgift att implementera systemet. Projektgruppens handledare är Johan Jönsson.

1.2 Syfte och mål

Inom materialfysik är elektronstrukturberäkningar ett viktigt teoretiskt verktyg. Detta för att få en så bra förståelse som möjligt av hur olika material är uppbyggda, bland annat vad gäller kristallers egenskaper sett ur ett kvantmekaniskt perspektiv. Det kan exempelvis handla om att man vill ta reda på olika materials egenskaper vad gäller värmeförståelse, strömsättning, förmåga etc.

För att öka förståelsen samt för att enklare kunna presentera resultat från forskning inom området är det viktigt att kunna göra olika typer av visualiseringar baserat på elektronstrukturberäkningar.

I många av de system som används för att utföra dessa beräkningar, exempelvis VASP, ges ingen eller begränsad möjlighet till detta. Vidare är de system som idag finns tillgängliga för visualisering förhållandevis ineffektiva, dåligt standardiserade samt har få visualiseringarfunktioner.

Syftet och målet för projektet har varit att uppdatera och utöka det modifierbara verktyg för visualisering av resultaten från elektronstrukturberäkningar som togs fram av 2017 års projektgrupp. Utöver det konkreta målet med utvecklingen av mjukvara för visualisering ska även projektet ge projektmedlemmarna erfarenhet av att arbeta i projekt och utöka deras förmåga till analytiskt och fysikaliskt tänkande för att ge värdefull erfarenhet inför arbetslivet.

1.3 Användning

Denna produkt kommer användas vid Linköpings universitet för att analysera data från elektronstruktursberäkningar.

1.4 Begränsningar

I projektet kommer visualiseringssverktyget Inviwo och programmeringspråken Python och C++ användas. Det kommer inte utredas om det är bättre att använda andra verktyg.

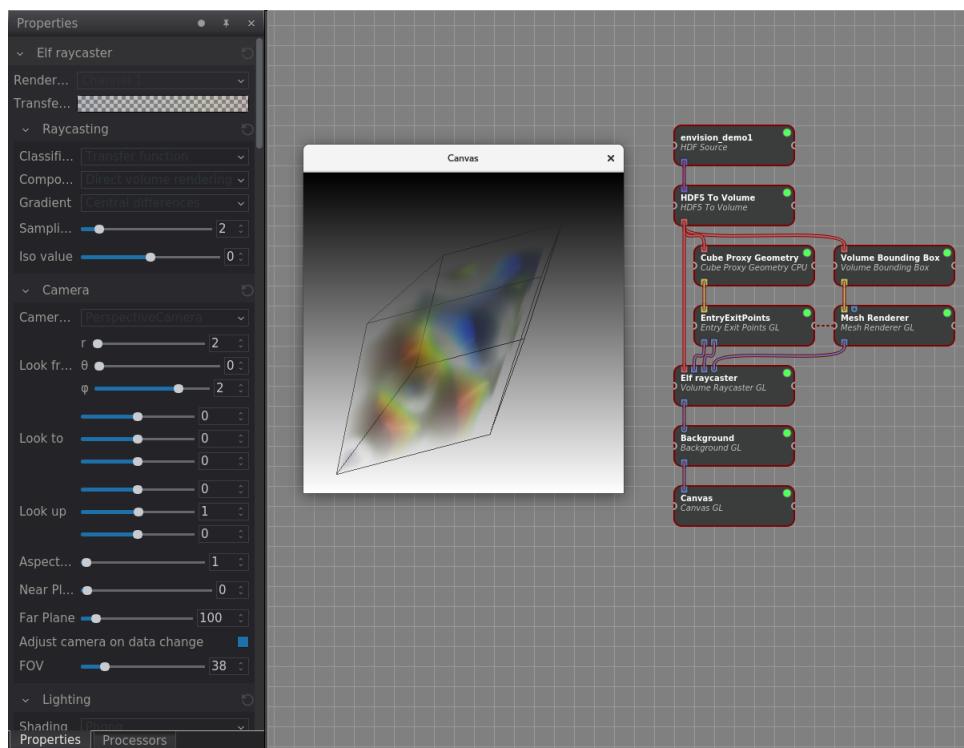
1.5 Definitioner

- **Inviwo:** Ett forskningsverktyg som utvecklas vid Linköpings universitet och ger användaren möjlighet att styra visualisering med hjälp av programmering i Python 3 eller grafiskt. Det tillhandahåller även användargränssnitt för interaktiv visualisering.
 - **Processor:** Benämningen på ett funktionsblock i Inviwos nätverksredigerare som tar emot indata och producerar utdata. I detta dokument avser en processor alltid en inviwoprocesor om inte annat anges.
 - **Ports:** Kanaler som processorer använder för att utbyta data av specifika typer.
 - **MultiImports:** Ports som kan ta emot flera datastrukturer av samma typ.
 - **Properties:** Variabler som bestämmer en processors tillstånd.
 - **Länkar:** Kanaler som processorer använder för att länka samman properties av samma typ så att deras tillstånd synkroniseras.
 - **Inviwo-nätverk:** Ett antal processorer sammankopplade via portar och länkar.
 - **vec3:** Vektor med tre komponenter.
 - **Mesh:** Polygonyta.
 - **Volymdata:** Tredimensionella data.
 - **API** (Application Programming Interface): En specifikation av hur olika applikationer kan använda och kommunicera med en specifik programvara. Detta utgörs oftast av ett dynamiskt länkat bibliotek. [8]
 - **BSD2** är en licens för öppen källkod. [3]
 - **C++** är ett programmeringsspråk. [4]
I Inviwo används C++ för att skriva programkod till processorer.
 - **Python** är ett programmeringsspråk. [12]
I Inviwo används Python för att knyta samman processorer.
 - **Fermi-energi** defineras som en energinivå där antalet tillstånd som har en energi lägre än Fermi-energin är lika med antalet elektroner i systemet. [2]
 - **Fermi-yta** är, för elektroners k-punkter i reciproka rymden, isoytan där elektronernas energi är lika med Fermi-energin. [9]
 - **Git** är ett decentraliserat versionshanteringssystem. [5]
 - **GUI** (Graphical User Interface) är ett grafiskt användargränssnitt. [10]
 - **HDF5** är ett filformat som kan hantera stora mängder data [6]. Alla HDF5-objekt har en rotgrupp som äger alla andra objekt i datastrukturen. Denna grupp innehåller i sin tur all övrig data i form av andra grupper, länkar till andra grupper eller dataset. Dataset innehåller rådata av något slag. Rådata kan i sammanhanget vara bilder, utdata från beräkningar, programdata, etc. [7, s. 4-5]
- De övriga objektstyperna gås inte igenom i detalj i detta dokument, men finns väl beskrivna i *High Level Introduction to HDF5* [7].
- **VASP** är ett program för modellering på atomnivå, för t.ex. elektronstruktursberäkningar och kvantmekanisk molekyldynamik. [1]
 - **Rastergrafik** är en typ av data som innehåller en matris av färgvärden, med bestämd och informationsbegränsad upplösning. [11]

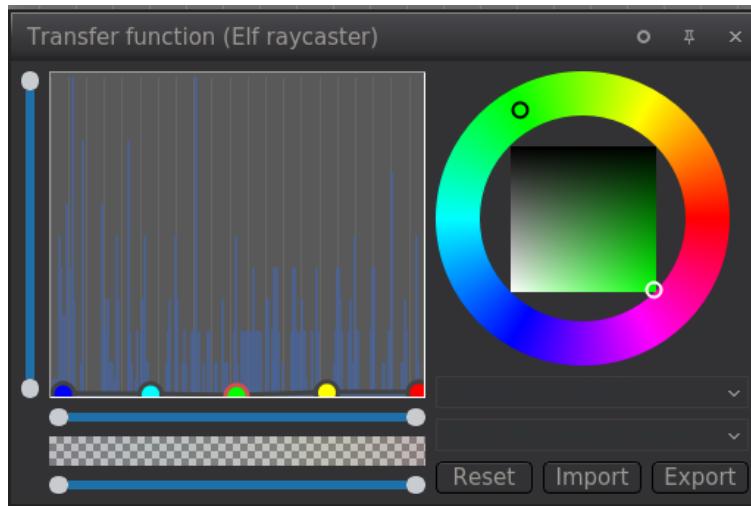
2 Produkten

Visualiseringssverktyget kan visualisera elektrontäthet, ELF (Electron localization function), kristallstruktur som atompositioner i enhetscellen samt total och partiell tillståndstäthet.

Figur 1 visar vad användaren ser när ELF-data för diamant visualiseras. Längst till vänster i figuren ses ett fönster med Properties, i detta fall för processorn ELF raycasting som kan hittas i nätverket längst till höger i figuren. Properties för de olika processorerna fås när en processor är markerad. I mitten av figuren ligger den utritade bilden. Bilden kan roteras genom att klicka och dra i den. I fönstret Properties finns en Property vid namn Transfer function, med hjälp av vilken användaren kan ändra transparens/opacitet samt färg, se figur 2.



Figur 1: Bild av vad användaren ser när ELF visualiseras, i detta fall för diamant.



Figur 2: Transfer function property där transparens och färg kan ställas in.

3 Översikt av systemet

Systemet är ett visualiseringssverktyg som kan användas för att visualisera utdata från elektronstrukturberäkningar i beräkningsprogrammet VASP. Programmet är modulärt programmerat och styrs via ett API.

3.1 Resurser

Följande resurser har använts för att utveckla systemet:

Mjukvara:

- Inviwo
- h5py
- Qt Creator
- CMake
- Git

Programmeringsspråk:

- Python 3
- C++ 14

3.2 Ingående delsystem

Delsystem 1 - System för parsning

Detta delsystem har i uppgift att läsa in data från VASP och omvandla till HDF5-formatet.

Delsystem 2 - System för visualisering

Detta delsystem har i uppgift att visualisera data som parsats av delsystem 1.

3.3 Utvecklingsfilosofi

Projektet har drivits med hjälp av versionshanteringssystemet Git och koden är licensierad med BSD 2, men även utvecklad under Inviwos utvecklaravtal för att, om önskvärt, kunna officiellt integreras i programvaran.

4 Delsystem 1 - System för parsning

Delsystemet 1 för parsning består av ett pythonbibliotek av moduler för att kunna hantera de olika utdatafilerna från VASP. Varje modul används för att läsa in specifika data från varje enskild utdatafil. Systemet har också en modul som anropas av samtliga andra moduler för att skriva HDF5-filer. Delsystemet parsar alltså först utdatafiler från VASP och strukturerar sedan om denna data till HDF5-format. Detta dataformat är delsystem 2 implementerat för att ha som indata, bland annat därfor att data i detta format har en enklare struktur.

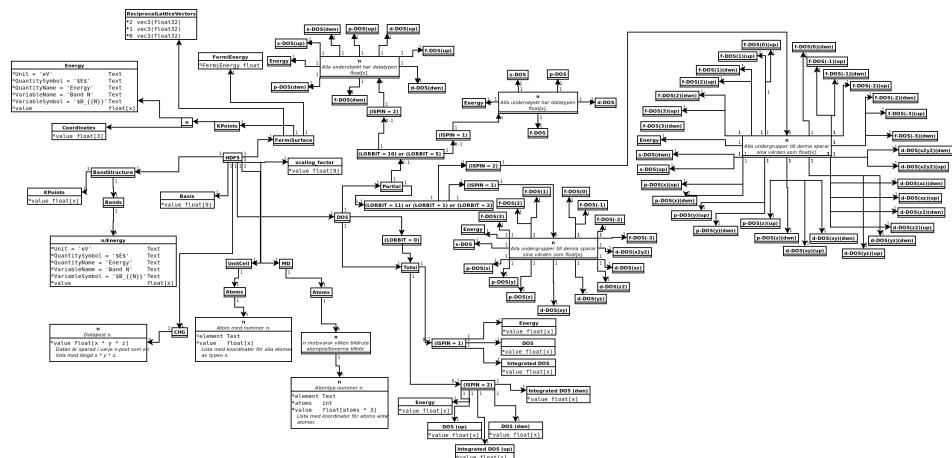
4.1 VASP

Från beräkningsprogrammet VASP fås en rad olika utdatafiler och dessa listas nedan.

- POSCAR innehåller data för enhetscellen samt atompositionsdata.
- CHG innehåller laddningstäthetsdata.
- DOSCAR innehåller tillståndstäthetsdata. Se kapitel 4.4.3 för mer information.
- EIGENVAL innehåller data för alla energier i k-rummet.
- POTCAR innehåller data om atomtyper.
- OUTCAR innehåller all utdata.
- XDATCAR innehåller data om enhetscell, atompositionsdata för varje beräkningssteg och även atomtyp.
- CONTCAR på samma format som POSCAR men CONTCAR fylls med information om atompresentationer uppdaterats.

4.2 HDF5-struktur för VASP-data

För att enkelt kunna läsa data i Inviwo följer HDF5-filerna som skapas av parsersystemet filstruktur enligt figur 3 nedan. Figur 3 beskriver det HDF5-formatet som VASP-datan ska läsas in till.



Figur 3: Dataformatet som används när VASP konverteras till HDF5. Se större bild i Bilaga A och B.

En ruta i diagrammet motsvarar en del i sökvägen i HDF5-objektet, om inte titeln är inom paranteser, då det innebär att rutans barnnoder blir åtkomliga om jämförelsen inom parenteserna är sann. Om en ruta har namngivna fält så innebär fältet *value* den data som är sparad i ett dataset, medan övriga datafält är attribut. En ruta som heter *n*, eller har det i namnet, innebär att det finns flera dataset och att de är numrerade med heltal.

På grund av platsbrist saknar figur 3 attribut för DOS-datan, men alla fält som innehåller DOS-data i diagrammet, till exempel p-DOS, d-DOS(xy), Energy, etc., har attribut som beskriver formatet på datan i fältet. En lista över attributen följer nedan:

- **VariableName** är fältets namn.
- **VariableSymbol** är en symbol som representerar variabeln.
- **QuantityName** är ett för en människa läsligt namn på fältet.
- **QuantitySymbol** är symbol som representerar storheten.
- **Unit** är storhetens fysikaliska enhet.

Den högra texten i varje ruta avser alltid datatypen för en datapost. I de fall datatypen följs av hakparanteser, till exempel *float[x]*, betyder det att data är sparad i en lista där värdet inom hakparanteserna utgör listans längd.

Som exempel så koms tillståndstätheten för den totala tillståndstätheten åt via */DOS/Total/DOS* oberoende av LORBIT. Ett till exempel är första energin för första bandet i bandstrukturen som nås via */BandStructure/Bands/0/Energy*.

4.3 Modul för skrivning till HDF5

Denna modul består av en pythonfil med namnet h5writer innehållandes funktionerna *_write_coordinates*, *_write_basis*, *_write_md*, *_write_steps*, *_write_bandstruct*, *_write_dos*, *_write_volume* samt *_write_incar*. Dessa funktioner skapar grupper och dataset enligt figur 3 ovan. Nedan följer en kort beskrivning av varje funktion.

_write_coordinates

Denna funktion skriver koordinater för atompositioner där varje atomslag tilldelas ett eget dataset. Attribut sätts för respektive grundämnesbeteckning per dataset.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- atom_count: Lista med antalet atomer av de olika atomslagen.
- coordinates_list: Lista med koordinater för samtliga atomer.
- Elements: = None eller lista med atomslag

Returnerar:

- None.

_write_basis

Denna funktion skriver gittervektorerna i ett dataset med namn basis.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- basis: Lista med basvektorerna.

Returnerar:

- None.

_write_bandstruct

Denna funktion skriver ut data för bandstruktur i en grupp med namn Bandstructure. Inom denna grupp tilldelas specifika K-punkter, energier samt bandstrukturer egna dataset. Diverse attribut sätts även för bl.a. specifika energier.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- band_data: Lista med bandstrukturdata.
- kval_list: Lista med K-punkter för specifika bandstrukturdata.

Returnerar:

- None.

_write_dos

Denna funktion skriver ut DOS-data i en grupp med namn DOS där total och partiell DOS tilldelas grupper med namn Total respektive Partial. Inom gruppen Total tilldelas energin samt specifika DOS egna dataset och inom gruppen Partial tilldelas varje partiell DOS egna grupper där energin samt specifika DOS tilldelas egna dataset.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- total: En lista med strängar av de olika uträkningarna som har utförts av VASP för total DOS.
- partial: En lista med strängar av de olika uträkningarna som har utförts av VASP för partiell DOS.
- total_data: En lista med alla beräkningar för total DOS för varje specifik atom.
- partial_list: En lista med alla beräkningar för partiell DOS för varje specifik atom.
- fermi_energy: Fermi-energin för den aktuella uträkningen.

Returnerar:

- None.

_write_volume

Denna funktion skriver ut elektrontäthetsdata och elektronlokaliseringefunktionsdata (ELF) till grupper med namn Charge respektive Elf. Inom dessa grupper tilldelas varje iteration ett dataset.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- i: Skalär som anger numret på iterationen.
- array: Array med parsad data för respektive iteration.
- data_dim: Lista som anger dimensionen av data för respektive iteration.
- hdfgroup: En textsträng med namnet på vad man vill kalla gruppen i HDF5-filen.

Returnerar:

- None.

_write_incar

Denna funktion skriver ut parsad data från INCAR i ett dataset med namn Incar där varje datatyp tilldelas egna dataset.

Parametrar:

- h5file: Sökväg till HDF5-fil.
- incar_data: Datalexikon med all data från INCAR-filen.

Returnerar:

- None.

4.4 Moduler för parsning

Samtliga moduler för parsning består av diverse pythonfunktioner som utför själva parsningen av data. Dessa innefattar parsers som endast parsar specifika data, en som parsar INCAR-data samt en funktion som anropar samtliga parsers utom INCAR-parsern.

4.4.1 Incarparser

Incarparsern består av pythonfil med namnet incar som innehåller funktionerna, incar och parse_incar. Dessa funktioner läser in och sparar information från INCAR-filen samt anropar en separat pythonmodul som skriver en HDF5-fil. INCAR är en inputfil för VASP som anger hur VASP ska exekvera elektronberäkningar. VASP-användaren har initialt satt värden rad för rad för en mängd variabler.

Funktionen incar kontrollerar om HDF5-filen redan innehåller INCAR-data och anropar funktionen parse_incar om så inte är fallet. Existerar INCAR-filen i användarens VASP-katalog parsas data av funktionen parse_incar som då sparar ett dataset för varje datatyp och namnger dataseten därefter. Funktionen incar anropar sedan pythonmodulen som skriver HDF5-filen där varje enskilt dataset tilldelas en egen grupp (se delavsnitt 4.2 ovan för beskrivning av strukturen i HDF5-filen).

Funktionsanrop: `envision.parser.vasp.incar(h5file, vasp_dir)`

Parametrar:

- `h5file`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till VASP-katalog.

Returnerar:

- Lista med namn på data (dataseten) som parsats.
- Bool: True om parsning skett felaktigt, False annars.

4.4.2 Volymparser

Volymparsern består av en mängd funktioner i en pythonfil som används för parsning av CHG och ELFCAR. Den kan läsa in och spara data på HDF5-format från båda dessa filer genom att anropa en pythonmodul. Detta är för att CHG och ELFCAR har samma struktur och består av ett antal iterationer av volymdata från volymberäkningar. Således innehåller den sista iterationen data som är mest korrekt. Därför skapar volymparsern också en länk till den sista iterationen i HDF5-filen för att data av högst kvalitet lätt ska kunna plockas ut.

Funktionsanrop vid parsning av CHG-data: `envision.parser.vasp.charge(h5file, vasp_dir)`

Funktionsanrop vid parsning av ELFCAR-data: `envision.parser.vasp.elf(h5file, vasp_dir)`

Parametrar:

- `h5file`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till VASP-katalog.

Returnerar:

- Bool: True om parsning skett felaktigt, False annars.

4.4.3 Tillståndstäthetsparser

Tillståndstäthetsparsern består av en mängd funktioner i en pythonfil som används för parsning av DOSCAR. DOSCAR-filen består först av den totala tillståndstäheten och sedan partiell tillståndstähet för varje atom i kristallen. Beroende på vad som står i INCAR kan dock denna data se väldigt olika ut. Flaggorna ISPIN, RWIGS och LORBIT i INCAR-filen avgör vad som skrivs i DOSCAR-filen. ISPIN-flaggan informerar om spinn har tagits hänsyn till vid beräkningar, RWIGS-flaggan specificerar Wigner-Seitz-radien för varje atomtyp och LORBIT-flaggan (kombinerat med RWIGS) avgör om PROCAR- eller PROOUT-filer (som DOSCAR-filen refererar till) skrivs. Parsern läser därför från data givet av incarparsern i HDF5-filen för att se hur DOSCAR ska parsas. Parsern delar upp data i två grupper i HDF5-filen, total och partiell. I gruppen partiell finns det en grupp för varje atom. Ett dataset för varje undersökt fenomen skrivs sedan ut för varje atom under partiell, och för total tillståndstähet under total.

Funktionsanrop: `envision.parser.vasp.dos(h5file, vasp_dir)`

Parametrar:

- `h5file`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till VASP-katalog.

Returnerar:

- Bool: True om parsning skett felfritt, False annars.

Parametrar:

- `h5file`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till VASP-katalog.

Returnerar:

- Bool: True om parsning skett felfritt, False annars.

4.4.4 Enhetscellsparser

Enhetscellsparsern läser in gittervektorer, som multipliceras med skalfaktorn och skrivs till /basis i HDF5-filen. Atompositioner läses från POSCAR och om dessa är angivna med kartesiska koordinater räknas de om till koordinater med gittervektorerna som bas. Koordinaterna skrivs till HDF5-filen uppdelade efter atomslag och attribut sätts med respektive grundämnesbeteckning. Om dessa inte ges med parametern `elements` letar parsern i första hand i POTCAR och i andra hand i POSCAR.

Funktionsanrop: `envision.parser.vasp.unitcell(h5file, vasp_dir, elements = None)`

Parametrar:

- `h5file`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till VASP-katalog.
- `elements = None`: None eller lista med atomslag.

Returnerar:

- Bool: True om parsning skett felfritt, False annars.

4.4.5 parse_all

parse_all är en funktion för parsning av allt som finns i katalogen som ges som inparameter. Funktionen kollar på alla systemets parsers och skriver ut meddelande om vad som parsas och om parsningen gjordes eller ej. Den returnerar en lista med alla namn på grupper i HDF5-filen eller None om HDF5-filen ej skapats.

Funktionsanrop: `envision.parse_all(h5_path, dir)`

Parametrar:

- `h5_path`: Sökväg till HDF5-fil.
- `vasp_dir`: Sökväg till katalog med utdata-filer från beräkningsprogram.

Returnerar:

- Bool: True om parsning skett felfritt, False annars.

5 Delsystem 2 - System för visualisering

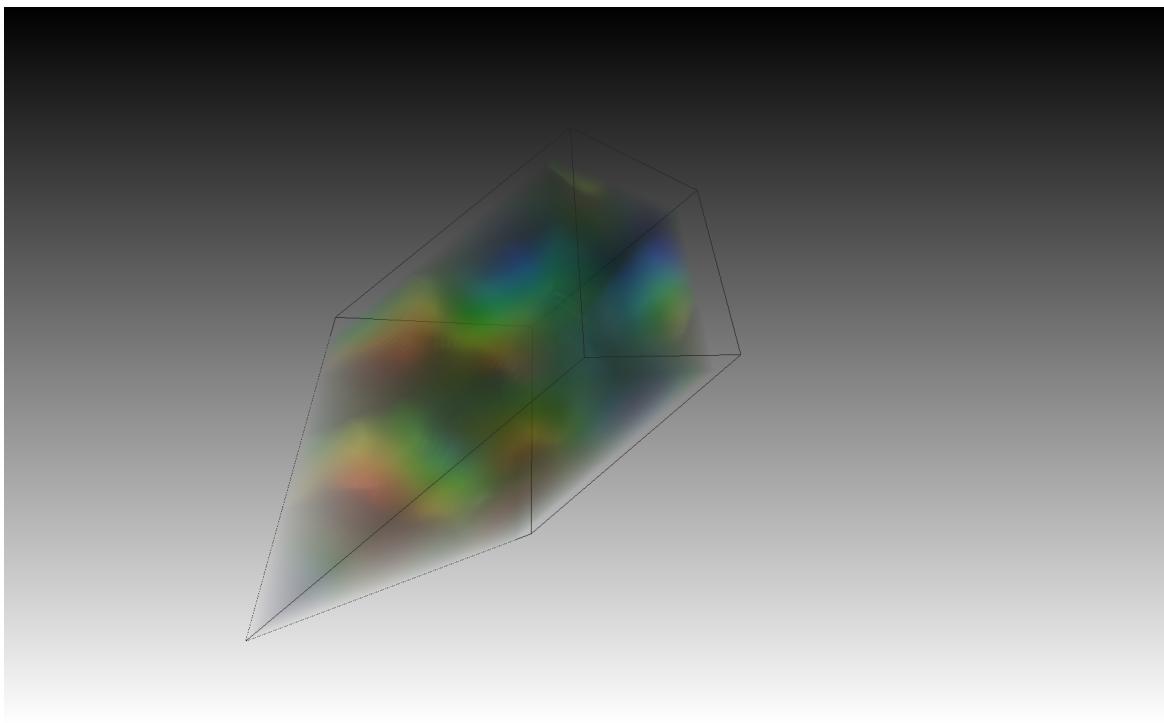
Visualiseringssystemet består av ett antal pythonmoduler för nätverksbygge i Inviwo samt c++-moduler implementerade i Inviwo av projektgruppen för att visualisera data.

5.1 Nätverk

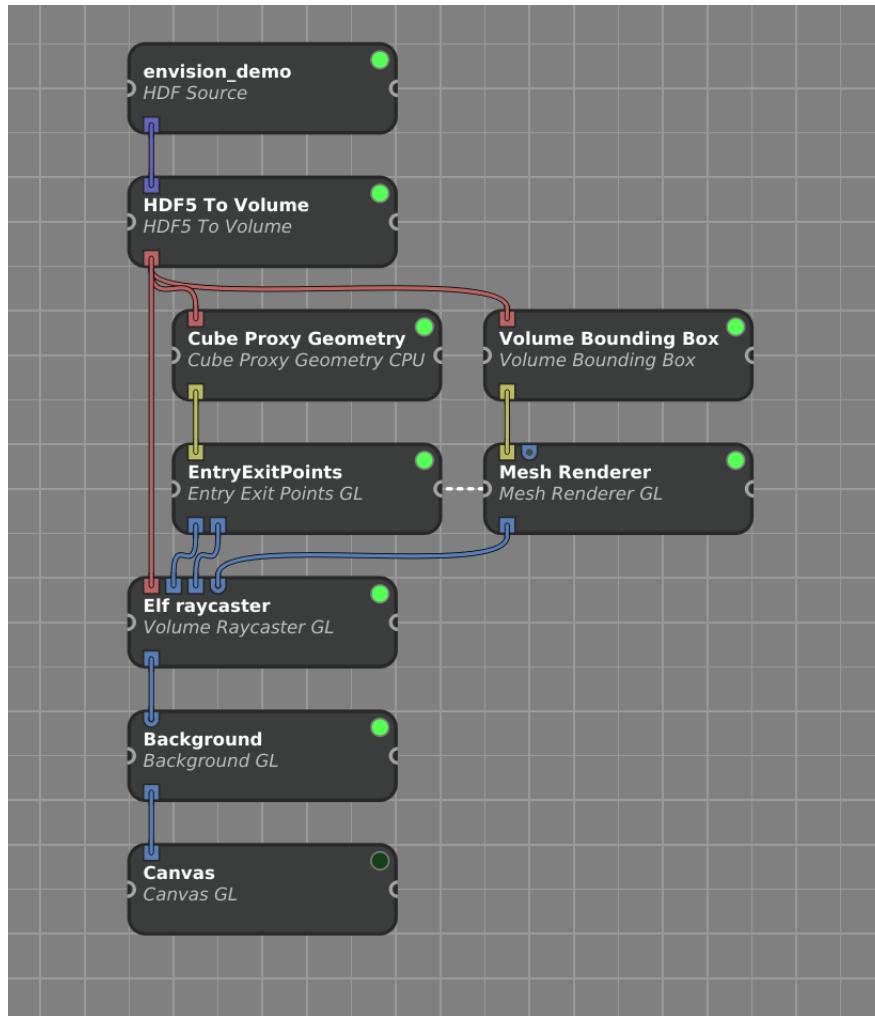
Processorer är de primära objekt i nätverken som användaren interagerar med. De består bland annat av importar och utportar som används vid utbyte av data av specifika typer. Nätverken mynnar ut i en så kallad Canvas, en processor som på sin import tar rastergrafikdata som den visualiseras i ett separat fönster. Samtliga pythonmoduler som bygger nätverk består i huvudsak av kod som anropar Inviwos egna interna funktioner för att bland annat importera, koppla, länka samt sätta värden på specifika parametrar i processorer.

5.1.1 Nätverk för visualisering av volymdata

Data som visualiseras med hjälp av volymnätverket är elektrontäthetsdata och elektronlokaliseringsfunktionsdata (ELF-data). Nätverket kan visualisera data med hjälp av tekniken volume ray casting eller iso ray casting. Se figur 4 nedan som visar en skärmbild från Inviwo när visualisering av ELF-data för diamant med iso ray casting görs. Det tillhörande nätverket syns i figur 5.

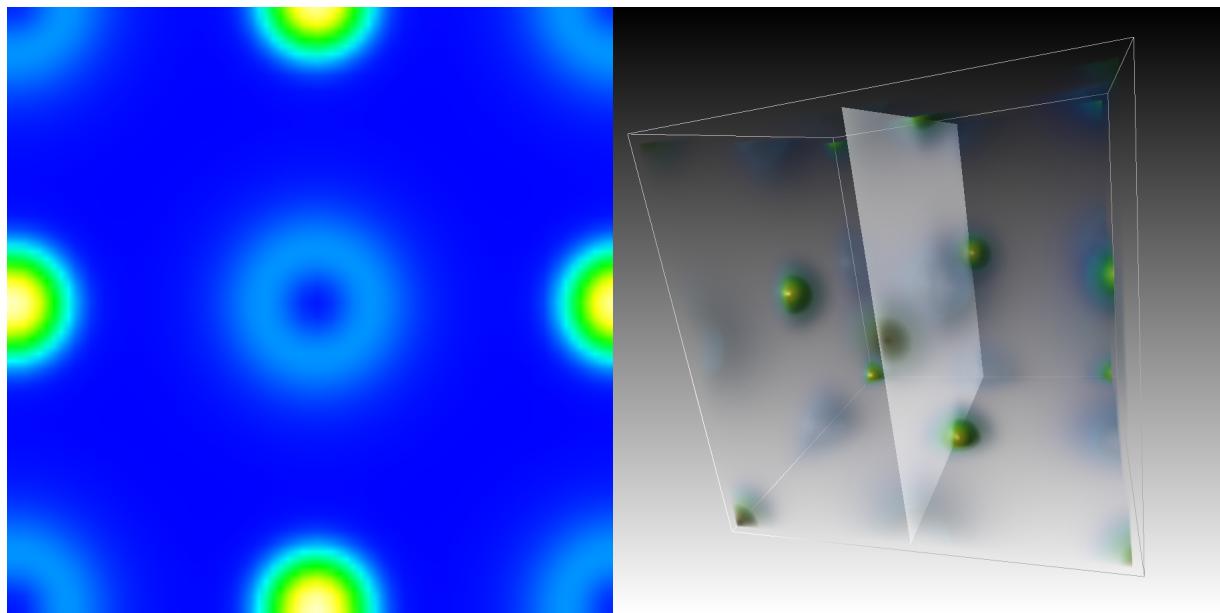


Figur 4: Visualisering av ELF-data för diamant i Inviwo.

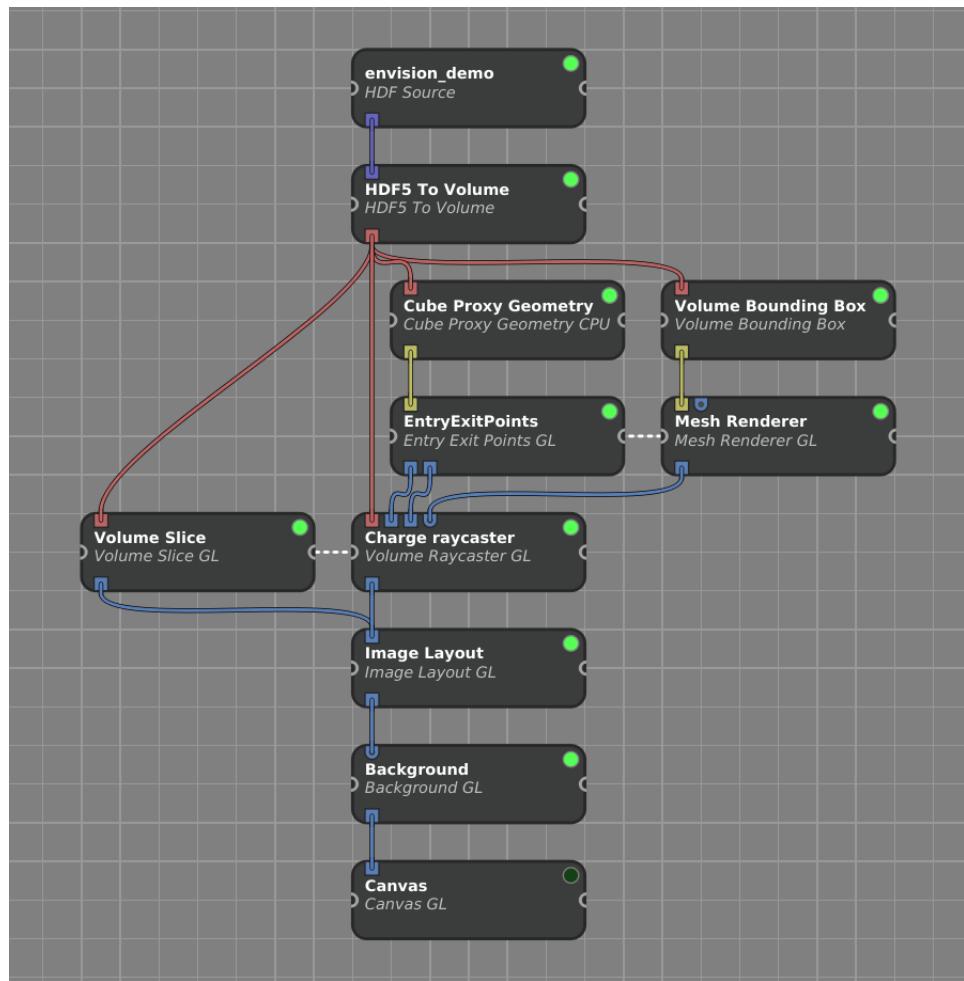


Figur 5: Nätverket för visualisering av ELF-data för diamant i Inviwo

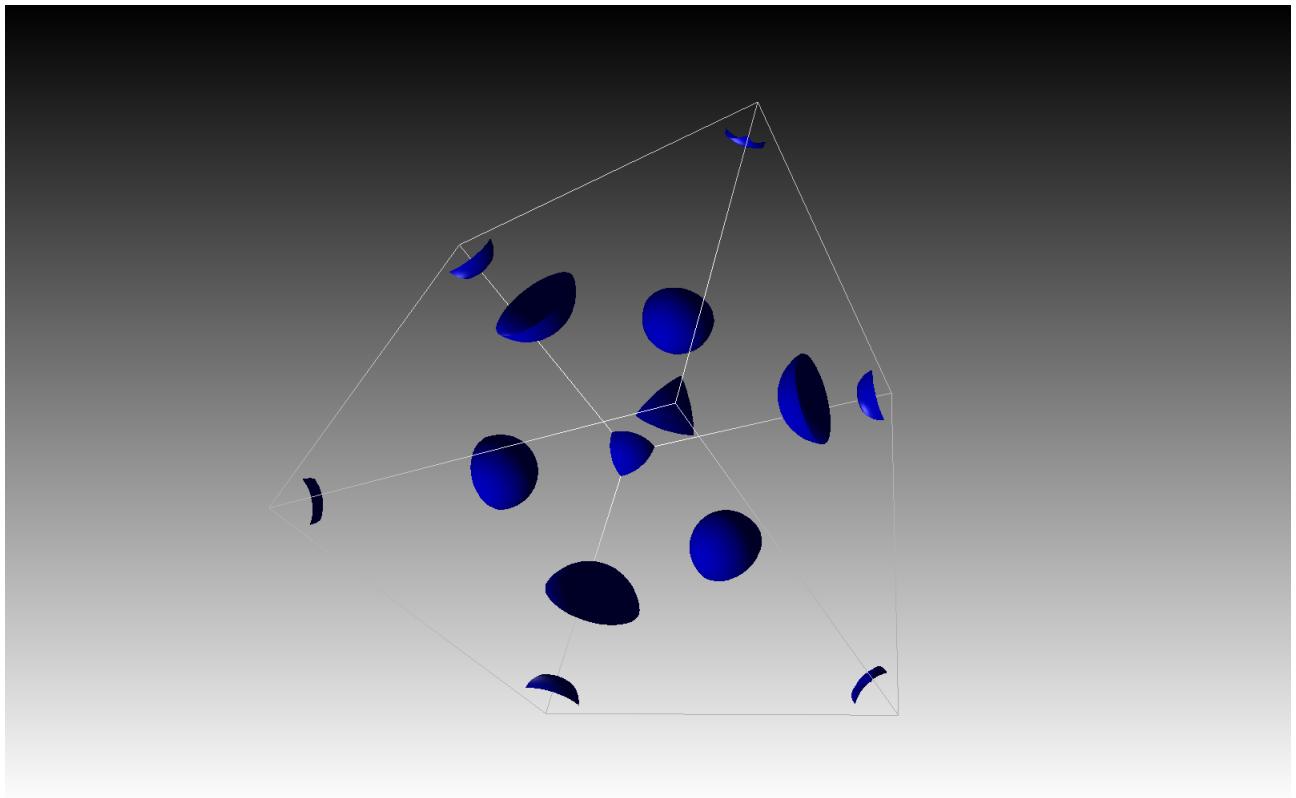
När tekniken volume ray casting används har användaren möjlighet att aktivera en slice-funktion som skapar ett plan som skär igenom volymen. Visualiseringen består då av två stycken bilder placerade bredvid varandra i samma Canvas. Högra bilden visar hela volymen tillsammans med ett plan, som användaren på eget behag kan flytta, och vänstra bilden visar utseendet av det specifika tvärsnittet av volymen som planet i varje enskild position skapar. Se figur 6 nedan som visar en skärmbild från Inviwo när visualisering av laddningstäthet för natriumklorid, NaCl, med volume ray casting och slice-funktion görs, och figur 7 för tillhörande nätverk. Försöker användaren visualisera data med iso ray casting och slice-funktion returneras meddelandet "Slice is not possible with ISO Raycasting, therefore no slice-function is showing." och en visualisering med iso ray casting utan slice-funktion görs. Figur 8 visar visualisering av laddningstäthet för NaCl med ISO ray casting, där en isoyta ritats upp för laddningstäthet lika med 0.15.



Figur 6: Visualisering av laddningstäthet för NaCl med slice-funktion i Inviwo.



Figur 7: Nätverket för visualisering av laddningstäthet för NaCl med slice-funktion i Inviwo.



Figur 8: Visualisering av laddningstäthet för NaCl med ISO raycasting.

Systemet som bygger volymnätverket består av en pythonfil med namnet volume innehållandes tre funktioner: charge, elf och volume_network.

Funktionerna charge och elf är de funktioner som användaren anropar och de tar vardera fem argument:

- h5file: Sökvägen till HDF5-filen där samtlig data finns.
- iso: Önskas iso-yta sätts denna till det specifika värdet iso-ytan ska ha annars till None. Defaultvärde = None.
- slice: Bool-variabel som sätts till True om slicefunktion önskas, annars till False.
- xpos: Den översta processorns x-koordinat. Default = 0.
- ypos: Den översta processorns y-koordinat. Default = 0.

Funktionsanrop charge:

```
envision.inviwo.charge(h5file, iso = None, slice = False, xpos = 0, ypos = 0)
```

Funktionsanrop ELF:

```
envision.inviwo.elf(h5file, iso = None , slice = False, xpos = 0, ypos = 0)
```

Dessa två funktioner anropar funktionerna volume_network som bygger själva nätverket. Denna tar argumenteten:

- h5file: Sökvägen till HDF5-filen där samtlig data finns.

- volume: En textsträng som anger om elektrontäthetsdata eller ELF-data ska visualiseras.
- iso: Önskas iso-yta sätts denna till det specifika värde iso-ytan ska ha annars till None.
- slice: Bool-variabel som sätts till True om slicefunktionen önskas, annars False.
- xstart_pos: Den översta processorns x-koordinat.
- ytart_pos: Den översta processorns y-koordinat.

Funktionen charge anropar användaren för att visualisera elektrontäthet och funktionen elf för att visualisera ELF-data. Båda dessa anger argumentet "h5file" i funktionen volume_network med det värde användaren gett som argument "h5file" till respektive funktion som. Argumentet "volume" anger funktionen charge som "Charge raycaster" och funktionen elf som "Elf raycaster". Resterande argument sätts, på samma sätt som för argumentet "h5file", till det värde användaren gett motsvarande argument till respektive funktion.

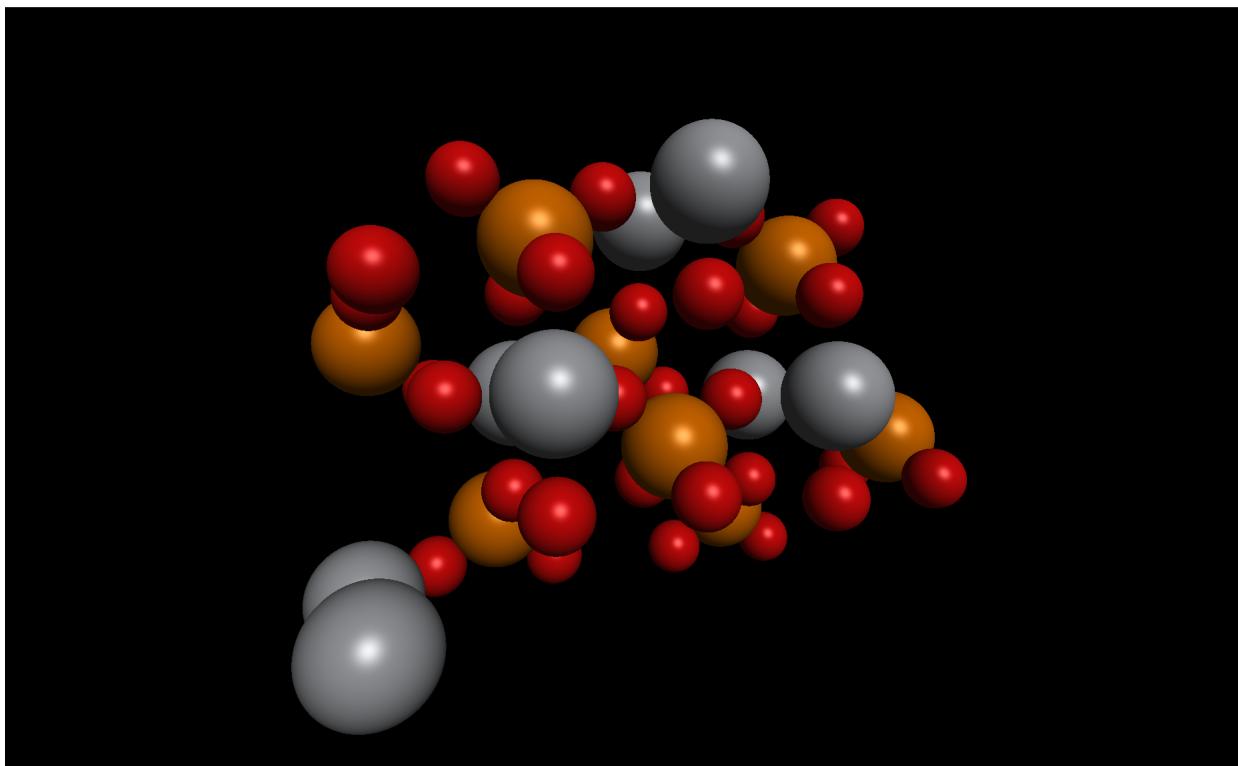
Funktionen volume_network laddar en HDF Source processor som ges samma namn som HDF5-filen. Dess utdata skickas till en HDF5 To Volume processor. HDF Source processorn anger vilken HDF5-fil och HDF5 To Volume processorn anger, baserat på argumentet "volume", vilken specifik grupp i HDF5-filen som data ska laddas från. Utdata från HDF5 To Volume processorn skickas till processorerna Cube Proxy Geometry, Volume Bounding Box samt till processorn för ray casting. Baserat på argumentet "iso" laddas processorn ISO Raycaster eller Volume Raycaster och denna ges namnet "Charge raycaster" eller "Elf raycaster" baserat på argumentet "volume". Processorn Cube Proxy Geometry skapar en proxy geometri i form av en mesh formad som en parallelepiped och skickar denna data till processorn Entry Exit Points. Denna processor beräknar start- och slutpunkt för respektive "stråle där data sampelas från" i processorn för ray casting och utgör således en del av denna processors indata. Beräkningen av start- och slutpunkt görs utgående från aktuell vinkel som den kamera som "ser in i" volymen data representerar har. Processorn Volume Bounding Box definierar geometrin för en mesh som omsluter volymen (utgör dess kanter) och skickar denna data till processorn Mesh Renderer som skapar meshen. Mesh Renderer skickar sedan denna data till processorn för ray casting.

Baserat på utdata från processorerna HDF5 To Volume, Entry Exit Points och Mesh Renderer skapar sedan processorn för ray casting en 2-dimensionell representation av volymen som data representerar. Denna data skickas, via processorn Background som lägger till en bakgrund, till Canvas-processorn. Se figur 5 som visar en skärmbild från Inviwo över nätverket när ELF-data för diamant visualiseras med volume ray casting.

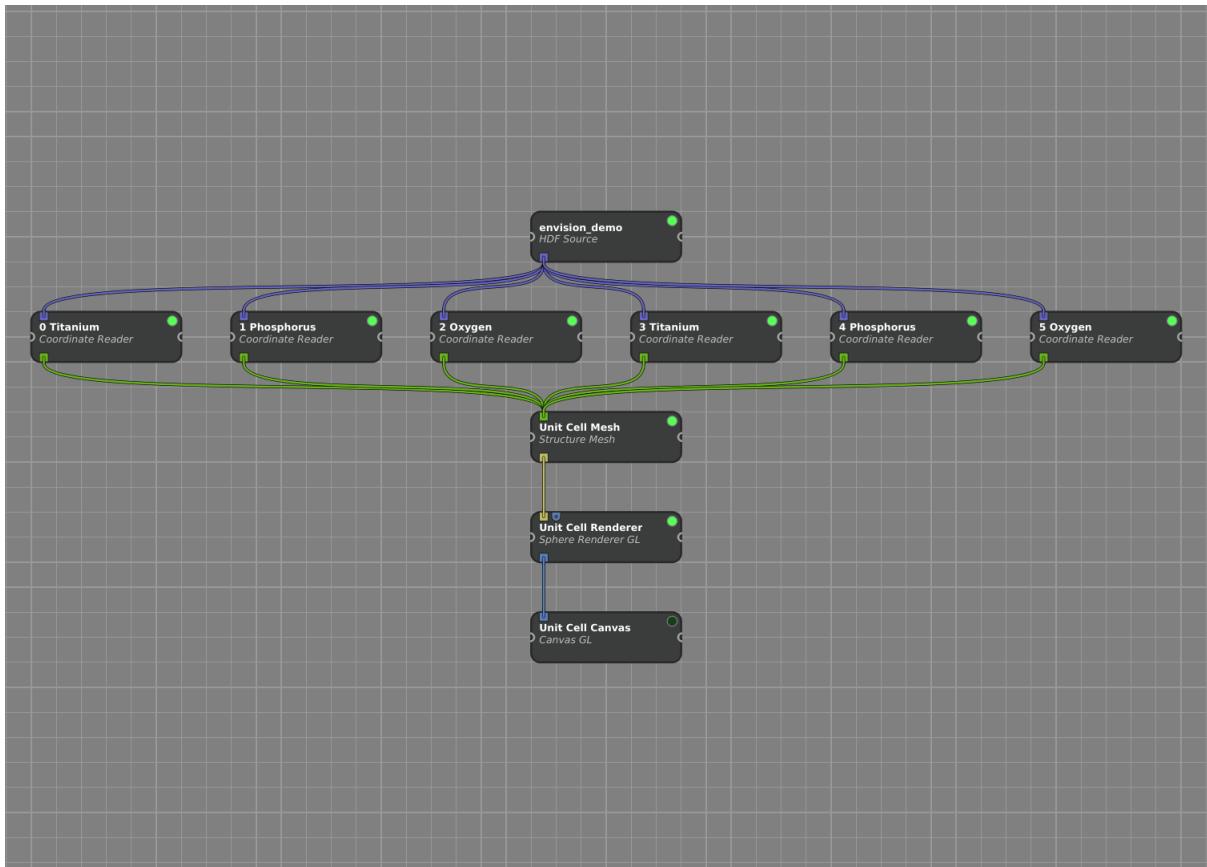
Om slicefunktionen används laddar även funktionen volume_network en Volume Slice processor och en Image Layout processor. Volume Slice processorn skapar det plan som skär volymen se figur 6 till höger. Indata till denna processor utgörs av utdata från HDF5 To Volume processorn och dess utdata går till processorn Image Layout. Processorn Image Layout skapar en delad vy över de båda bilderna se figur 6 ovan och dess indata utgörs, förutom av utdata från Volume Slice processorn, av utdata från processorn för ray casting. Denna data skickas sedan, via processorn Background som skapar en bakgrund, till Canvas-processorn. Se figur 7 som visar en skärmbild från Inviwo över nätverket när laddningstäthet för natriumklorid, NaCl, visualiseras med hjälp av volume ray casting och slice-funktion.

5.1.2 Nätverk för visualisering av enhetscell

Funktionen för visualisering av enhetscell placerar ut en koordinatläsarprocessor (Coordinate-Reader) per atomslag och kopplar dessa till en processor av typen HDF source, som läser den av användaren angivna HDF5-filen. Utdata från koordinatläsarna skickas till en StructureMesh-processor som placerar ut klot som representerar atomer. Färg och radie sätts enligt en fördefinierad lista, men kan ändras via Inviwos grafiska användargränssnitt. Utdata från StructureMesh skickas till processorn Sphere Renderer, som sköter själva renderingen. Utporten på denna processor kan kopplas till den blå importen på processorn Mesh Renderer i volymrenderingsnätverket, se delavsnittet 5.1.1 om detta nätverk ovan, så att enhetscellen visas tillsammans med elektronstrukturen. Se figur 9 nedan som vardera visar en skärmbild från Inviwo när visualisering av enhetscell för titanfosfat ($TiPO_4$) görs, och figur 10 för tillhörande nätverk.



Figur 9: Visualisering av $TiPO_4$ i Inviwo

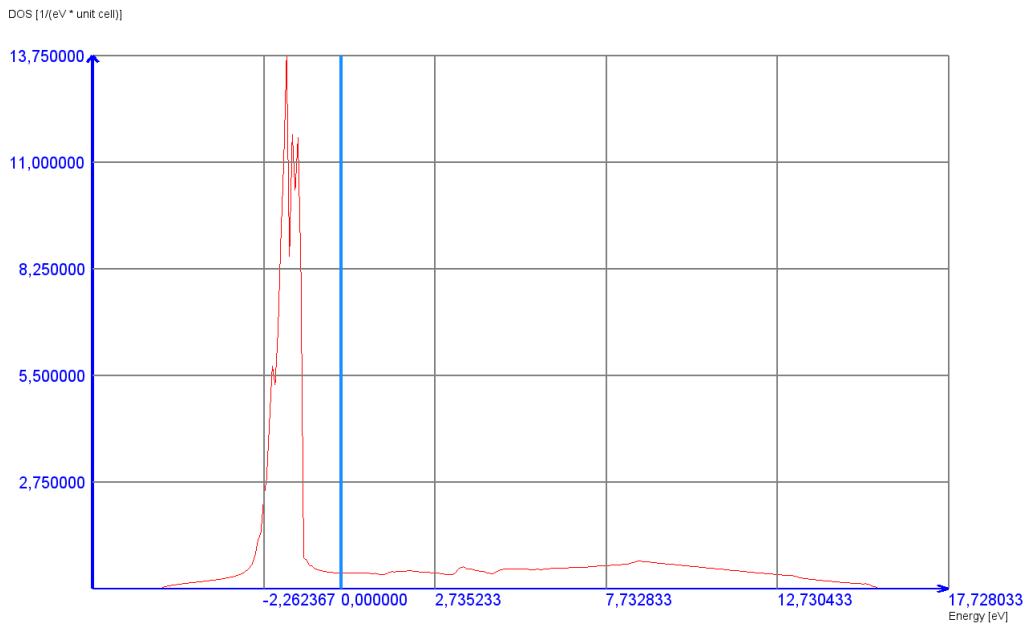


Figur 10: Nätverket för visualisering av TiPO4 i Inviwo.

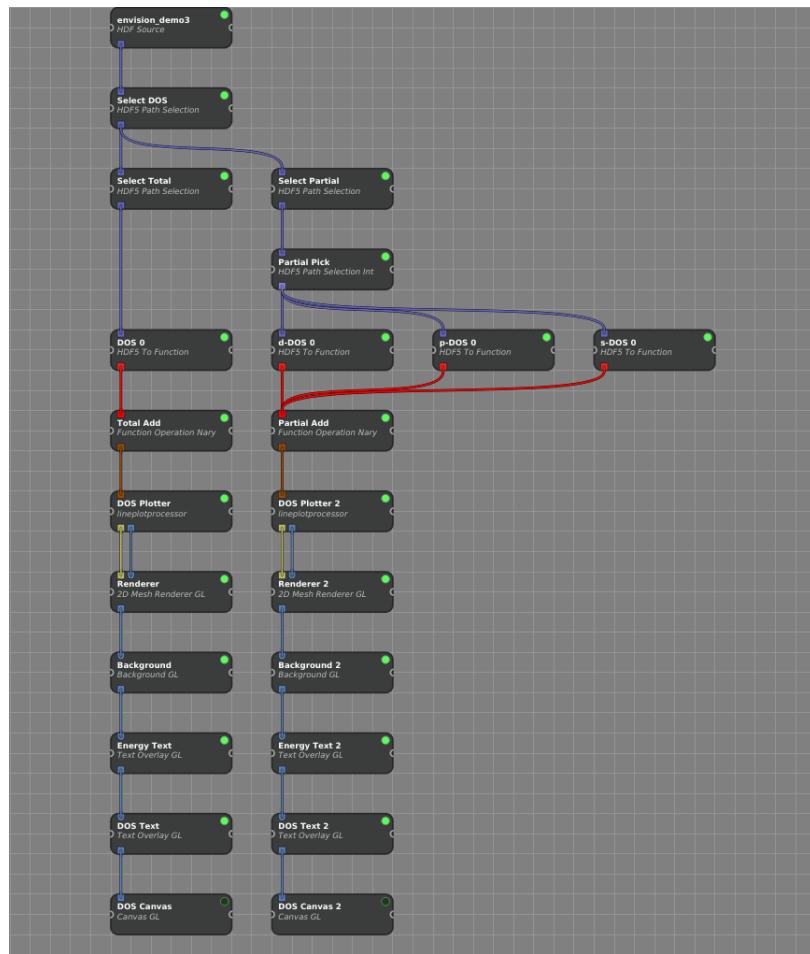
5.1.3 Nätverk för visualisering av DOS

Nätverket för visualisering av tillståndstäthetsdata laddar en HDF Source processor som anger HDF5-filen som data laddas från om HDF Source processor inte finns, annars kopplas en HDF5 Path Selection processor, som tar ut den givna HDF5-gruppens alla undergrupper (beskrivs närmare i kap. 5.3) direkt till den redan befintliga HDF Source processorn. Denna processor anger att data ska laddas från DOS-gruppen i HDF5-filen. Två till HDF5 Path Selection processorer laddas sedan som anger grupperna Total och Partial i HDF5-filen.

För Total-delen laddas sedan kontinuerligt HDF5 To Function processorer som gör funktioner av all data i Total-gruppen. För Partial-gruppen laddas en HDF5 Path Selection processor (beskrivs närmare i kap. 5.3.2) som tar ut dataset för en vald atom genom att välja den givna HDF5-filens relevanta undergrupp. Denna processor har namnet Partial Pick i nätverket. Därefter laddas HDF5 To Function processorer för alla dataset i grupperna under Partial-gruppen. All data matas sedan i en lineplot processor (beskrivs närmare i kap. 5.3.3) som gör en 2D-graf. Detta matas in i en Canvas-processor som visar själva grafen. Dessutom finns två textOverlay processorer som skriver ut text för x- och y-axeln. Figur 11 visar total tillståndstäthet för koppar, Cu. Skärmbilden i figur 12 är över nätverket som ger 2D-grafen i figur 11, nätverket ger även en 2D-graf av den partiella tillståndstätheten.



Figur 11: Visualisering av Total DOS för Cu.

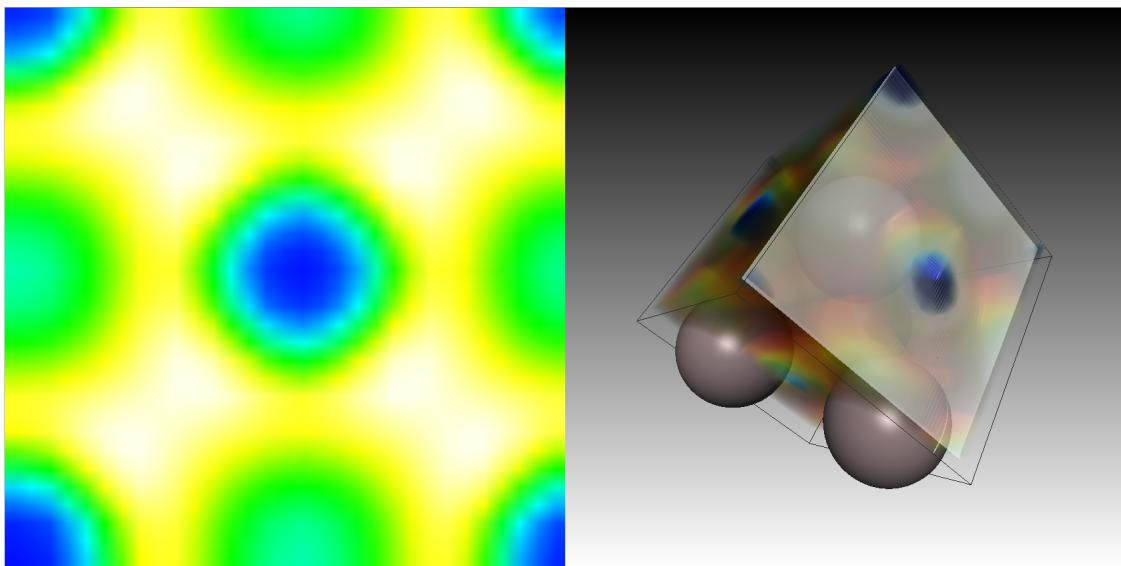


Figur 12: Nätverket för visualisering av total och partiell tillståndstäthet för Cu.

Det går att visualisera dos tillsammans med visualisering av kristallstruktur och med hjälp av en så kallad ”picking”-funktion välja enskilda atomer för att se tillhörande tillståndstäthet. Mer om detta i kap. 5.1.4.

5.1.4 Sammankoppling av nätverk

Alla de typer av visualisering som är listade ovan kan köras samtidigt, oberoende av varandra, så länge nödvändig data återfinns i de tillhandahållna VASP-filerna. Funktionalitet för att koppla ihop nätverk med varandra och visualisera flera egenskaper i samma bild finns för några kombinationer av egenskaper. Alla kombinationer av tredimensionell visualisering i det ”direkta” rummet (alltså inte det reciproka rummet) kan fås i samma bild. Ett exempel på detta är figur 13 som visar visualisering av enhetscellsdata och ELF-data för aluminium, Al, i samma bild.



Figur 13: Data för enhetscell och ELF för aluminium visualiseras i samma bild, med volume ray casting och slice-funktion. Slice-funktionens plan är här placerat alldeles intill enhetscellens sida.

DOS sammankopplat med enhetscell

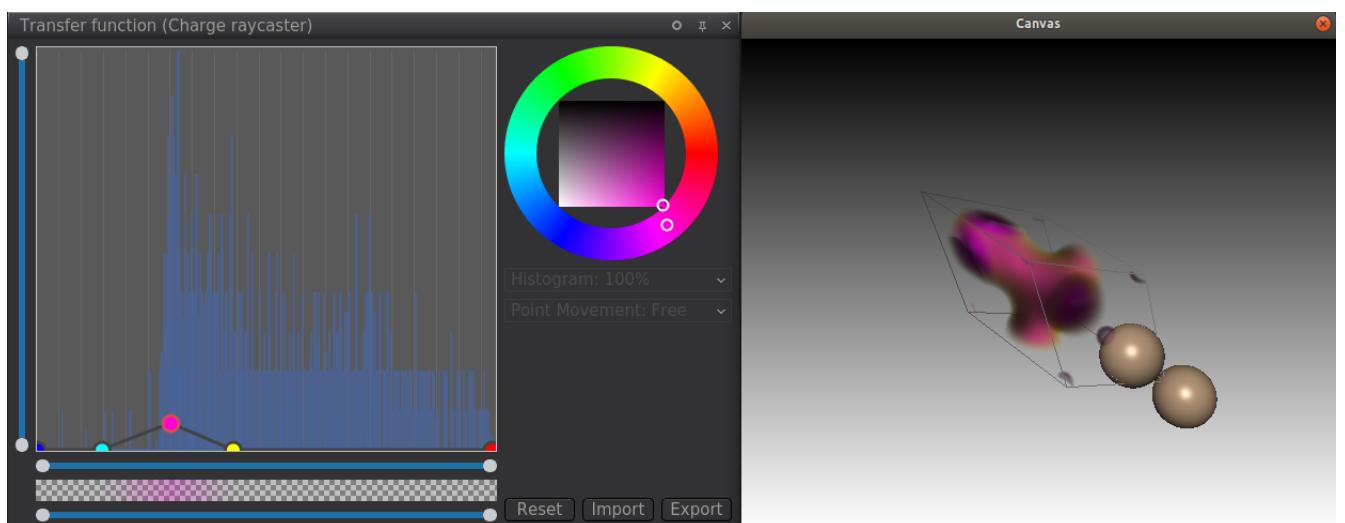
DOS och enhetscellen kan visualiseras samtidigt och ger då en bild med kristallstrukturen till vänster och tillståndstätheten åt höger. Då möjliggör en ”picking”-funktion att användaren kan trycka på enskilda atomer för att få ut den projicerade tillståndstätheten. Processorn Structure-Mesh har en BoolProperty med namnet enablePicking_ (se kap. 5.3.1), sätts denna till True så blir det möjligt att välja enskilda atomer i enhetscellen genom att klicka på dem.

Om en processor med namnet ”Unit Cell Mesh” hittas i nätverket så byts Partial Pick All mot Partial Pick och i Unit Cell Mesh sätts enablePicking_ till True efter att Partial Picks:s property intVectorProperty_ och Unit Cell Mesh:s property inds_ länkats samman.

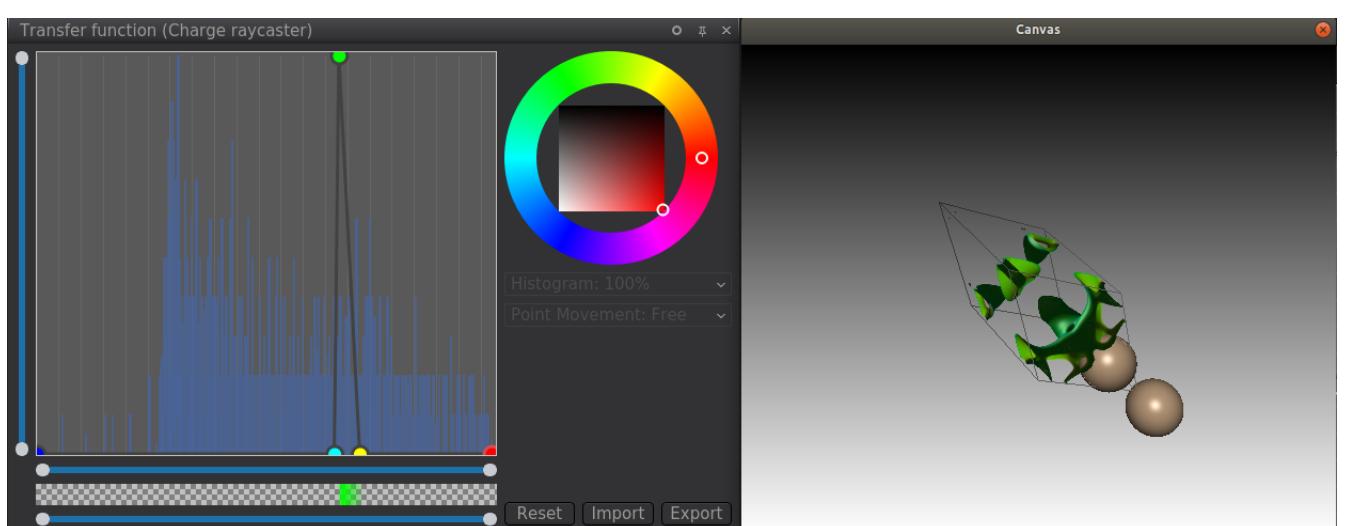
5.1.5 Färg-och-transparensinställning

Färger och transparens för samtliga typer av tredimensionell visualisering kan ändras godtyckligt av användaren.

- Enhetscellsvisualisering: Färg och transparens ändras i processorn *Unit Cell Mesh* i propertyn *color i*, där *i* är numret på den atomtyp som inställningarna ska ändras för.
- Visualisering av volymdata: Färg och transparens (ej för ISO raycasting, då isoytor ritas helt opaka) ändras i processorn *Charge Raycaster* eller *ELF Raycaster* för laddningstäthetsvisualisering respektive *ELF*-visualisering, i propertyn *Transfer function*. Här visas en interaktiv graf med volymsdatavärdet längs x-axeln och opacitet längs y-axeln. Med hjälp av punkter som placeras ut i detta plan kan användaren styra färgen och transparensen på vad som visualiseras. Se figur 14 och 15 för exempel på dessa inställningar för överföringsfunktionen för visualiseringen av laddningstäthet av kisel i diamantstruktur.



Figur 14: Laddningstäthetsvärdet mellan 0.15 och 0.45 markeras med en någorlunda genomskinlig lila yta.



Figur 15: Laddningstäthetsvärdet kring 0.7 markeras med en opak grön yta, nästan en isoyta.

Slice-funktioners plans färg och transparens kan även de ändras. Detta i processorn *Volume Slice*, i propertyn *Transfer function*.

5.2 Datastrukturer

Två datastrukturer, Point och Function, har introducerats. Dessa används i vissa av de implementerade processorerna.

5.2.1 Point

Denna datatyp representerar en reell 1D-punkt och inkapslar punktens värde (ett flyttal) samt variabel metadata.

5.2.2 Function

Denna datatyp representerar en reellvärd funktion av en reell variabel och inkapslar sampelvärdet och variabel-metadata för x- och y-axlarna.

5.3 Processorer

Ett antal processorer har implementerats, dessa kategoriseras och beskrivs nedan.

5.3.1 Kristallstruktur

Nedanstående processorer är relaterade till visualiseringen av kristallstrukturer.

CoordinateReader

Från en HDF5-fil läser denna processor koordinater för atompositioner. En sökväg till ett dataset sätts via en property kallad *StringProperty*. Denna *StringProperty* ska ha storleken $n*3$ och bestå av 32-bitars flyttal. Utdata från *CoordinateReader* är n stycken *vec3*.

Import:

- `Hdf5::Import import_`

Utpart:

- `DataOutport< std::vector<vec3> > outport_`

Properties:

- `StringProperty path_`

StructureMesh

Atompositionsdata kopplas ihop med rätt atomfärg och radie med StructureMesh-processorn. Dessutom hanterar processorn ”picking”-funktionen som gör det möjligt för användaren att välja atomer genom att klicka på dem. StructureMesh har en multiimport, dit en eller flera CoordinateReader-processorer kan kopplas in. Indata för StructureMesh är atompositionsdata i form av vec3 för varje atomslag. Till denna indata läggs properties för färg, radie och antal till för varje atomslag/processor som kopplas in. Den ger en mesh, som har buffrar för position, färg och radie.

Om picking-funktionen är påslagen får meshen även en pickingbuffer, som innehåller de globala picking id som tilldelas av PickingMappern. Då användaren vänsterklickar på en atom läggs dess lokala id i IntVectorPropertyn. Färgen ändras på alla valda atomer genom att alfaflagret sätts till 0,5.

Import:

- DataImport< std::vector<vec3>, 0> structure_

Utpart:

- MeshOutport mesh_

Properties:

- FloatProperty scalingFactor_
- FloatMat3Property basis_
- BoolProperty fullMesh_
- IntProperty timestep_
- std::vector< std::unique_ptr<FloatVec4Property> > colors_: vektor som innehåller färgproperty för varje atomslag
- std::vector< std::unique_ptr<FloatProperty> > radii_: vektor som innehåller radieproperty för varje atomslag
- std::vector< std::unique_ptr<IntProperty> > num_: vektor som innehåller antalet atomer per tidssteg för varje atomslag
- BoolProperty enablePicking_: sann då picking-funktionen är påslagen
- IntVectorProperty inds_: vektor med index på valda atomer

5.3.2 HDF5

Nedanstående processorer är ämnade att fungera väl med de HDF5-relaterade processorer som är inkluderade i Inviwo.

HDF5PathSelection*

Detta är en grupp av processorer som har funktionalitet liknande den inbyggda processorn

HDF5PathSelection. En eller flera av dessa processorer placeras med fördel mellan en HDF-Source och en eller flera HDF5To*.

Gemensamt för dessa processorer är att de på importen tar en Hdf5-grupp och på utporten skriver noll eller flera av dessa omedelbara undergrupper.

Nedan beskrivs de olika processorerna i denna grupp.

HDFpathSelectionInt

Denna processor väljer en HDF5-grupp med heltalsnamn, baserat på värdet på processorns intProperty_, eventuellt utökat med ledande nollar till bredden specificerat på processorns zeroPadWidthProperty_.

HDF5PathSelectionInt kan med fördel användas tillsammans med en OrdinalPropertyAnimator för att plocka ut relevant data ur en HDF5-fil.

Anledningen till att utdata ges som en vektor av HDF5-grupper, trots att processorn alltid skriver exakt en grupp på utporten, är att processorn ska följa samma mönster som, och fungera väl med, resterande processorer.

Inport:

- DataImport<hdf5::Handle> hdf5HandleImport_

Utpart:

- DataOutport< std::vector<hdf5::Handle> > hdf5HandleVectorOutport_

Properties:

- IntProperty intProperty_
- IntSizeTProperty zeroPadWidthProperty_

HDF5PathSelectionIntVector

Denna processor väljer noll eller flera HDF5-grupper med heltalsnamn, baserat på värdet på processorns intVectorProperty_, eventuellt utökat med ledande nollar till berdden specificerat av processorns zeroPadWidthProperty_.

HDF5PathSelectionIntVector kan med fördel användas tillsammans med ”picking” för att plocka ut relevant data ur en HDF5-fil.

Inport:

- DataImport<hdf5::Handle> hdf5HandleImport_

Utpart:

- DataOutport< std::vector<hdf5::Handle> > hdf5HandleVectorOutport_

Properties:

- IntVectorProperty intVectorProperty_

- IntSizeTProperty zeroPadWidthProperty_

HDF5PathSelectionAllChildren

Denna processor väljer den givna HDF5-gruppens alla undergrupper.

Inport:

- DataImport<hdf5::Handle> hdf5HandleImport_

HDF5To*

Detta är en grupp av processorer som har funktionalitet liknande den inbyggda processorn HDF5ToVolume. Processorerna placeras med fördel efter en HDFSource-processor, med en eller flera mellan liggande HDF5PathSelection*.

Gemensamt för dessa är att de som indata tar noll eller flera HDF5-grupper (baserat på *pathSelectionProperty_), plockar ut dataset för varje grupp och omvandlar dessa till relevanta objekt (Point eller Function) som sedan skrivs till utporten. Objektens variabel-metadata tas, om de finns tillgängliga, från attributen associerade med dataseten. Vidare kan, om så väljs med *namePrependParentsProperty_, metadat utökas med namnen på de grupper var i dataseten ligger.

Vilka dataset som kan väljas med *pathSelectionProperty_ uppdateras dynamiskt beroende på vilka grupper som ligger på importen. När ett lämpligt dataset valts kan *pathFreezeProperty_ användas för att stänga av denna dynamik, så att värdet sparas även om grupperna på importen (antagligen tillfälligt) ändras. Detta underlättar manuellt experimenterande samt användandet av processorer som tillfälligt ger noll grupper som utadat, t.ex. HDF5PathSelectionIntVector.

HDF5ToPoint

Denna processor konverterar HDF5-data till noll eller flera Point-objekt.

Inport:

- DataImport<hdf5::Handle, 0, true> hdf5HandleFlatMultiImport_

Utpart:

- DataOutport< std::vector<Point> > pointVectorOutport_

Properties:

- OptionPropertyString pathSelectionProperty_
- BoolProperty pathFreezeProperty_
- IntSizeTProperty namePrependParentsProperty

HDF5ToFunction

Denna processor konverterar HDF5-data till noll eller flera Function-objekt.

Normalt plockas två dataset per grupp ut, ett för x-axeln och ett för y-axeln. Om endast data för y-axeln finns tillgänglig kan implicitXProperty_ sättas, varvid processorn automatgenererar data för x-axeln.

Inport:

- DataImport<hdf5::Handle, 0, true> hdf5HandleFlatMultiImport_

Utpart:

- DataOutport< std::vector<Function> > functionVectorOutport_

Properties:

- BoolProperty implicitXProperty_
- OptionPropertyString xPathSelectionProperty_
- OptionPropertyString yPathSelectionProperty_
- BoolProperty xPathFreezeProperty_
- BoolProperty yPathFreezeProperty_
- IntSizeTProperty xNamePrependParentsProperty_
- IntSizeTProperty yNamePrependParentsProperty_

5.3.3 2D

Nedanstående processorer är ämnade att bearbeta och presentera 2D-data, närmare bestämt data av typen Point och Function.

FunctionOperationUnary

Denna processor implementerar en unär operator, antingen negation ($g_i(x) = -f_i(x)$) eller (multiplikativ) inversion ($g_i(x) = 1/f_i(x)$). Operatorn appliceras på funktioner på importen, en i taget, och skriver respektive resultat på utporten.

Import:

- DataFrameImport dataframeImport_

Utpart:

- DataFramOutport dataframOutport_

Properties:

- OptionPropertyString operationProperty_

FunctionOperationNary

Denna processor implementerar en operator med variabel aritet (engelska n-ary), antingen addition/summa ($g(x) = \sum_i f_i(x)$) eller multiplikation/produkt ($g(x) = \prod_i f_i(x)$). Operatorn appliceras på samtliga funktioner på importen och skrver resultatet på utporten.

Då funktionerna på importen kan vara samplade vid olika x-värden behöver processorn ta beslut om var ut-funktionen ska sampas. Processorn utgår från att sampla i samtliga x-värden för

samtliga in-funktioner. `sampleFilterEnableProperty_` kan sättas för att filtrera dessa. Då `sampleFilterEnableProperty_` är satt ser processorn till att sampelavståndet är minst det värde som anges i `sampleFilterEpsilonProperty_`. När processorn skapas är `sampleFilterEnableProperty_` satt och `sampleFilterEpsilonProperty_` är 0 vilket innebär att x-värden som är identiska filtreras bort.

Om ett värde behöver beräknas vid ett x-värde där en in-funktion inte är samplat används linjär interpolation om x-värdet ligger innanför funktionens definitionsintervall. Om x-värdet ligger utanför detta intervall används `undefinedFallbackProperty_` för att avgöra vilket värde som används istället. Detta kan antingen vara noll eller funktionens värde vid intervallets relevanta ändpunkt.

Import

- `org.envision.FunctionFlatMultiImport functionFlatMultiImport_`

Utpart:

- `DataFramOutport dataframOutport_`

Properties:

- `OptionsPropertyString operationProperty_`
- `OptionsPropertyString undefinedFallbackProperty_`
- `BoolProperty sampleFilterEnableProperty_`
- `FloatProperty sampleFilterEpsilonProperty_`

lineplotprocessor

lineplotprocessor tar en *DataFrame* som förväntas innehålla två kolumner med punkter, kallade X och Y. Den konstruerar en mesh som representerar en linjegraf och denna mesh renderas sedan förslagsvis med hjälp av en *2D Mesh Renderer*-processor för att generera en bild av grafen.

lineplotprocessor genererar även en utbild att lägga över grafen som innehåller axelgraderingen. Axelgraderingen kan också den skickas in i *2D Mesh Renderer*-processorn och kommer då läggas ovanpå grafen.

Inställningar som har *range* i namnet justerar minimum- och maximumvärden på koordinataxlarna. Inställningar med *width* eller *colour* justerar bredd respektive färg för olika linjer ritade i diagrammet.

label_number_ anger antalet divisioner på koordinataxlarna. Är värdet till exempel satt till tjugo innebär det att varje axel kommer ha tjugo divisioner och tjugo axelgraderingsetiketter.

font_ ställer in vilket typsnitt axelgraderingen skall ha.

enable_line_ aktiverar ritandet av en vertikal linje på x-koordinaten specificerad i *line_x_coordinate_*. Denna är avsedd att ge en visuell markering av var specifika x-värden finns på x-axeln.

Import:

- `DataFrameImport dataFrameImport_`

Uports:

- MeshOutport meshOutport_
- ImageOutport labels_

Properties:

- FloatVec4Property colour_
- FloatVec2Property x_range_
- FloatVec2Property y_range_
- FloatProperty scale_
- BoolProperty enable_line_
- FloatProperty line_x_coordinate_
- FloatVec4Property line_colour_
- FloatVec4Property axis_colour_
- FloatProperty axis_width_
- FloatVec4Property grid_colour_
- FloatProperty grid_width_
- FontProperty font_
- FloatVec4Property text_colour_
- IntProperty label_number_

5.4 Properties och widgets

5.4.1 IntVectorproperty

Denna property består av en vektor av int-värden.

5.4.2 IntVectorPropertyWidget

En widget för IntVectorProperty. "Textbox", satt till endast läsning (read only), som innehåller de värden som finns i tillhörande IntVectorProperty.

6 Utvecklingsmöjligheter

De egenskaper som visualiseras under 2018 års projekt kan fortfarande vidareutvecklas för att få nya funktioner eller för att presentera datan på ett annat sätt om det anses mer lättförståeligt eller givande.

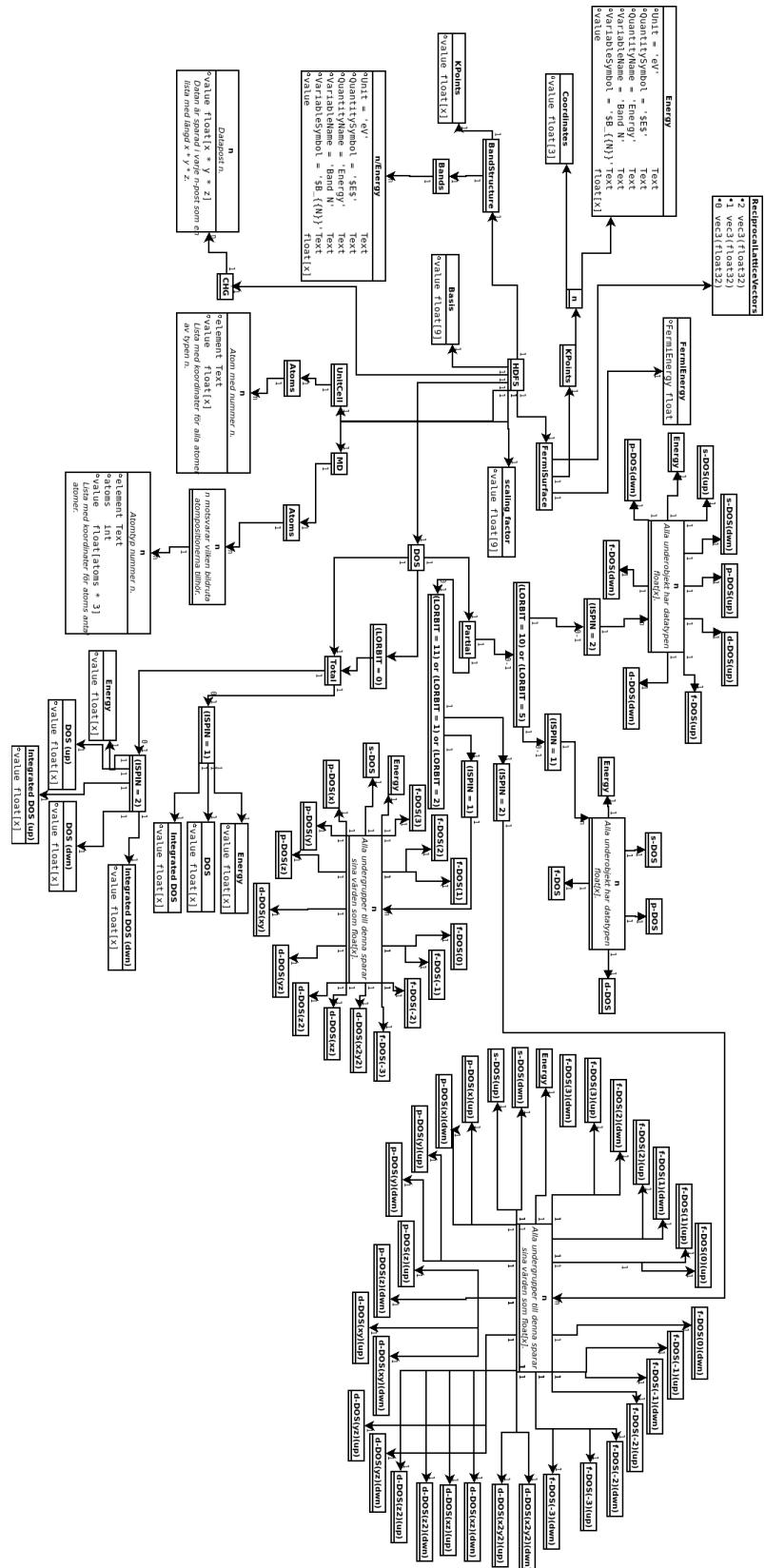
Molekyldynamik och bandstruktur är två egenskaper som inte har behandlats under detta års projektarbete men som det finns möjlighet att uppdatera från 2017 års projektarbete. I StructureMesh-processorn kan en property läggas till som möjliggör molekyldynamik. För bandstrukturvisualisering finns en parser och samt en visualiseringssmodul som kräver uppdatering.

Detta år har Fermi-ytor varit en av egenskaperna som skulle visualiseras, det hann dock inte bli helt klart. Mycket arbete har ändå lagts ner som det finns möjlighet att bygga vidare på och få att fungera.

Referenser

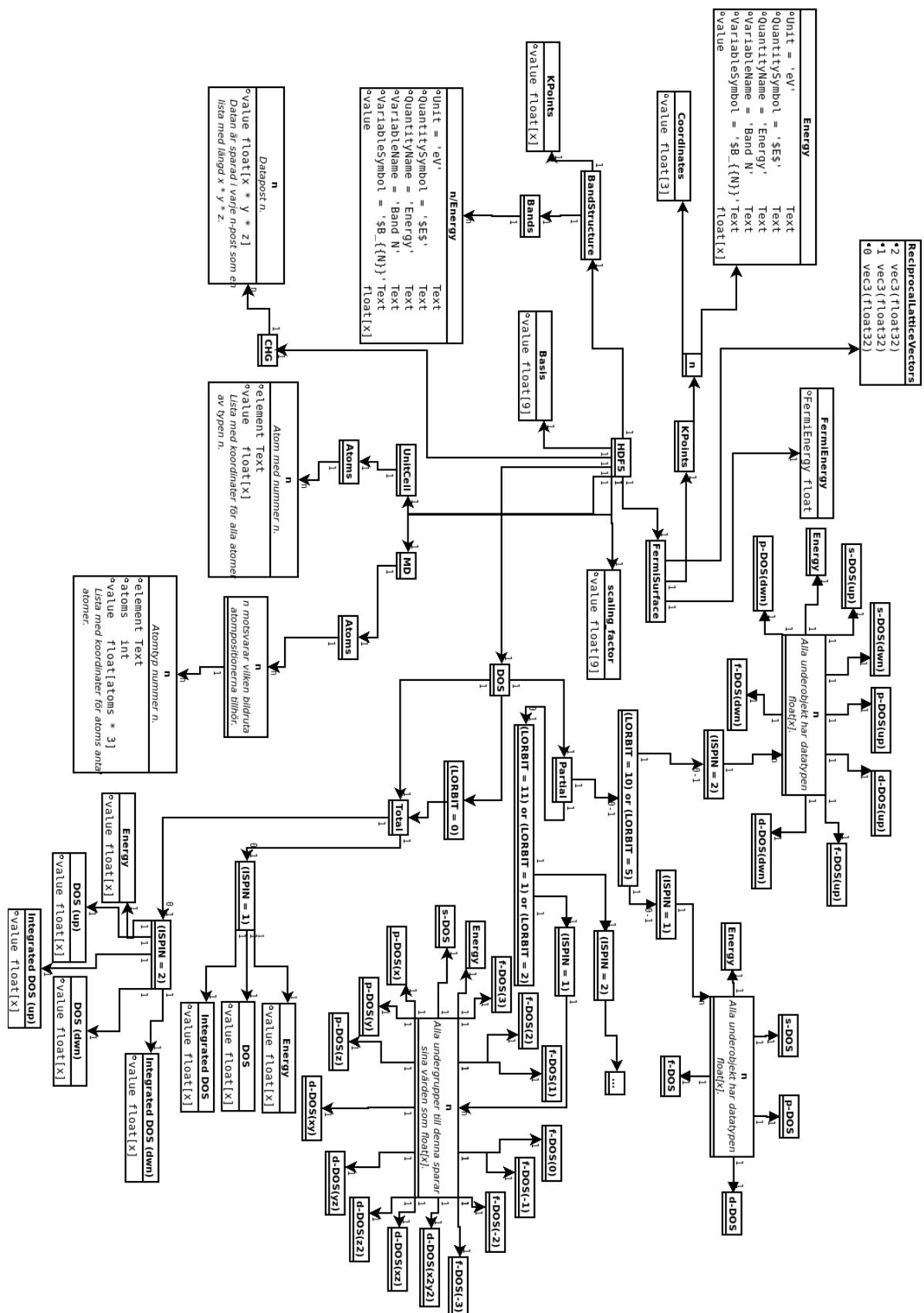
- [1] *About VASP*. URL: <https://www.vasp.at/index.php/about-vasp/59-about-vasp> (hämtad 2018-02-22).
- [2] Neil Ashcroft och David Mermin. *Solid State Physics*. 1976, s. 141.
- [3] *BSD2*. URL: <https://opensource.org/licenses/BSD-2-Clause> (hämtad 2018-02-23).
- [4] *C++*. URL: <http://www.cplusplus.com/info/description/> (hämtad 2018-02-23).
- [5] *Git*. URL: <https://git-scm.com> (hämtad 2018-02-23).
- [6] The HDF Group. *Hierarchical Data Format, version 5*. 1997-2018. URL: <https://support.hdfgroup.org/HDF5/> (hämtad 2018-02-21).
- [7] The HDF Group. *High Level Introduction to HDF5*. 23 sept. 2016. URL: <https://support.hdfgroup.org/HDF5/Tutor/HDF5Intro.pdf> (hämtad 2018-02-21).
- [8] *Nationalencyklopedin. API*. URL: [https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-\(data\)](https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-(data)) (hämtad 2018-02-23).
- [9] *Nationalencyklopedin. Fermi-tyta*. URL: <https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/fermi-tyta> (hämtad 2018-02-23).
- [10] *Nationalencyklopedin. GUI*. URL: [https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-\(data\)](https://www.ne.se/uppslagsverk/encyklopedi/1%C3%A5ng/api-(data)) (hämtad 2018-02-23).
- [11] *Ordguru. Rastergrafik*. URL: <https://www.ordguru.se/synonymer/rastergrafik> (hämtad 2018-04-26).
- [12] *Python*. URL: <https://www.python.org> (hämtad 2018-02-23).

A HDF5-datastruktur

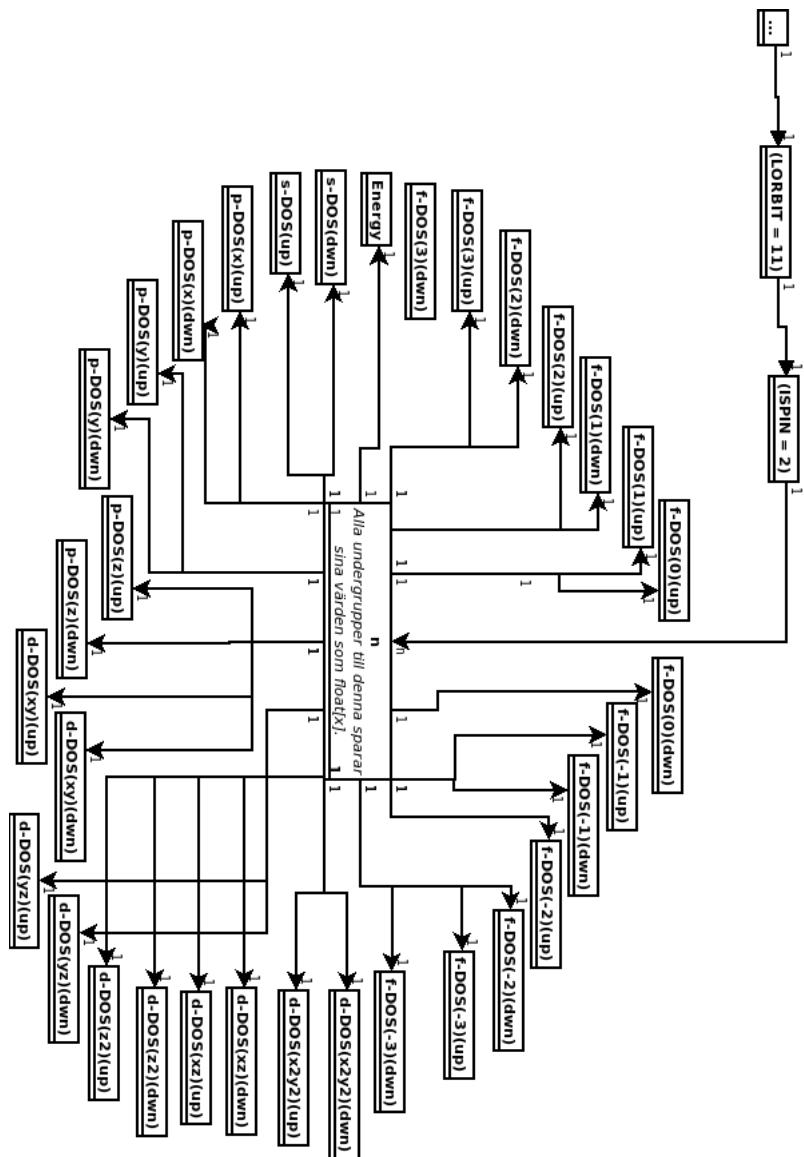


Figur 16: Dataformatet som används när VASP konverteras till HDF5.

B HDF5-datastruktur i två delar



Figur 17: Dataformatet som används när VASP konverteras till HDF5 del 1.



Figur 18: Dataformatet som används när VASP konverteras till HDF5 del 2.

L Naturvetenskaplig undersökning**Naturvetenskaplig undersökning**

Analys av visualiseringar gjorda med visualiseringssverktyget
ENVISIoN

Redaktör: Marian Brännvall

Version 0.2

Status

Granskad	PL	18-05-25
Godkänd		

PROJEKTIDENTITET

2018/VT, Grupp 2
Linköpings Tekniska Högskola, IFM

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Anders Rehult	Projektledare (PL)	076-3161206	andre449@student.liu.se
Marian Brännvall	Dokumentansvarig (DOK)	070-7280044	marbr639@student.liu.se
Andreas Kempe	Sekreterare (SE)	073-9796689	andke133@student.liu.se
Viktor Bernholtz	Viktor Bernholtz (VB)	073-0386030	vikbe253@student.liu.se

Kund: IFM, Linköpings universitet, 581 83 Linköping

Kontaktperson hos kund: Rickard Armiento, 013-281249, rickard.armiento@liu.se

Kursansvarig: Per Sandström, 013-282902, persa@ifm.liu.se

Handledare: Johan Jönsson, 013-281176, johan.jonsson@liu.se

Innehåll

Dokumenthistorik	iv
1 Inledning	1
1.1 Syfte	1
1.2 Bakgrund	1
2 Utförande	1
3 Resultat och slutsatser	1
3.1 Kristallstrukturer	1
3.2 Elektrontäthet	2
3.3 Tillståndstäthet	4
3.4 Elektronlokaliseringsfunktion, ELF	5
4 Diskussion	6

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2018-05-24	Första utkast.	PG	PL
0.2	2018-05-25	Andra utkast.	PG	PL

1 Inledning

Dokumentet är en naturvetenskaplig undersökning för kandidatprojektet i visualisering av elektronstrukturer. Här undersöks fysiken kring de egenskaper som har visualiseras i projektet för att få en bättre förståelse för dessa fenomen.

1.1 Syfte

Syftet med den naturvetenskapliga undersökningen är att ge en beskrivning av fysiken tillhörande de framtagna visualiseringarna.

1.2 Bakgrund

Kandidatprojektet i visualisering av elektronstrukturer har innefattat visualisering av ett antal fysikaliska fenomen: kristallstruktur, elektronräntäthet, tillståndstäthet samt elektronlokaliseringsfunktionen (eng. electron localization function, ELF). Se kapitel 1.4, *Definitioner*, i projektets slutrapport för information om dessa. För att få en bättre förståelse för fysiken kring dessa görs denna naturvetenskapliga undersökning som utifrån de framtagna visualiseringarna beskriver dessa fenomen.

2 Utförande

Det framtagna visualiseringssverkyget, ENVISIoN, har använts för att visualisera de olika fenomenen. Visualiseringar har gjorts för olika ämnen och genom att analysera dessa har slutsatser kunnat dras, dessa presenteras nedan i kap. 3.

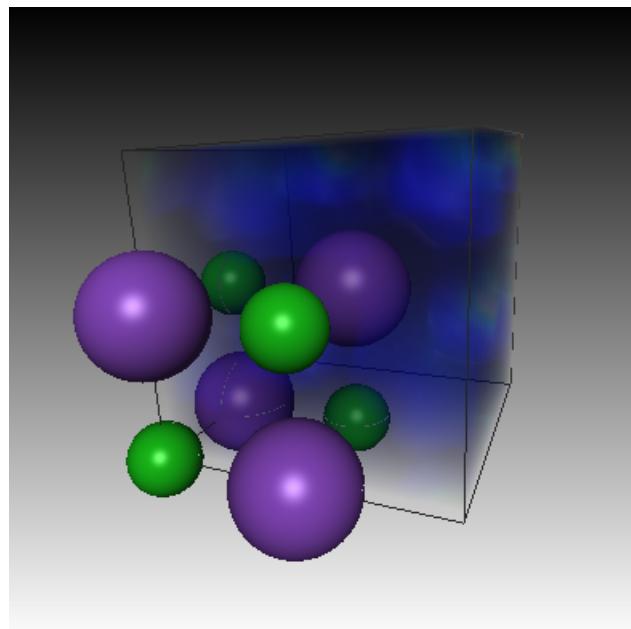
3 Resultat och slutsatser

Här beskrivs resultaten av visualiseringarna av de olika egenskaperna samt slutsatserna som drogs utifrån dem.

3.1 Kristallstrukturer

En av egenskaperna som kan visualiseras med det framtagna visualiseringssverkyget är kristallstrukturen för ett givet ämne. Kristallstrukturen beskriver atomernas positioner i enhetscellen där atomerna representeras av sfärer med olika radie för de olika atomtyperna.

I figur 1 ses kristallstrukturen för natriumklorid, NaCl. NaCl är av ytcentrerad-kubisk struktur vilket syns i skärmbilden där (de gröna) Cl-atomerna sitter i hörnet samt på ytorna till den kubiska enhetscellen.

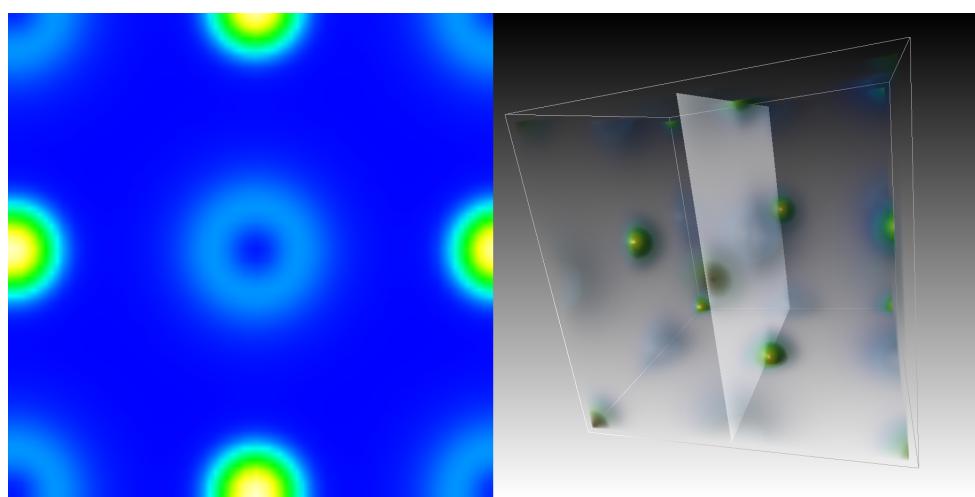


Figur 1: Skärmbild från visualisering av enhetscell tillsammans med elektrontäthet för natriumklorid.

3.2 Elektrontäthet

En av egenskaperna som kan visualiseras med det framtagna visualiseringssverktyget är elektrontäthet. Elektrontätheten beskriver sannolikheten att hitta en elektron på en given plats.

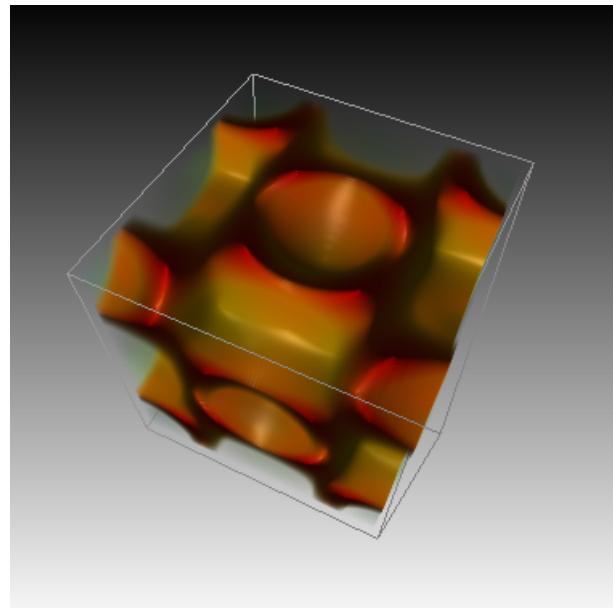
I figur 2 ses en skärmbild av en visualisering av elektrontätheten hos natriumklorid, NaCl. I detta fall är slice-funktionen påslagen vilken gör att ett plan fås som kan flyttas i den tre-dimensionella bilden för att få en två-dimensionell bild till vänster. Vad som kan ses här är en betydligt högre sannolikhet att hitta elektroner kring Cl-atomerna än kring Na-atomerna. Detta faller sig naturligt eftersom NaCl är en jonbindning där en elektron har överförts från Na-atomen till Cl-atomen.



Figur 2: Skärmbild från visualisering av elektrontäthet för natriumklorid, NaCl.

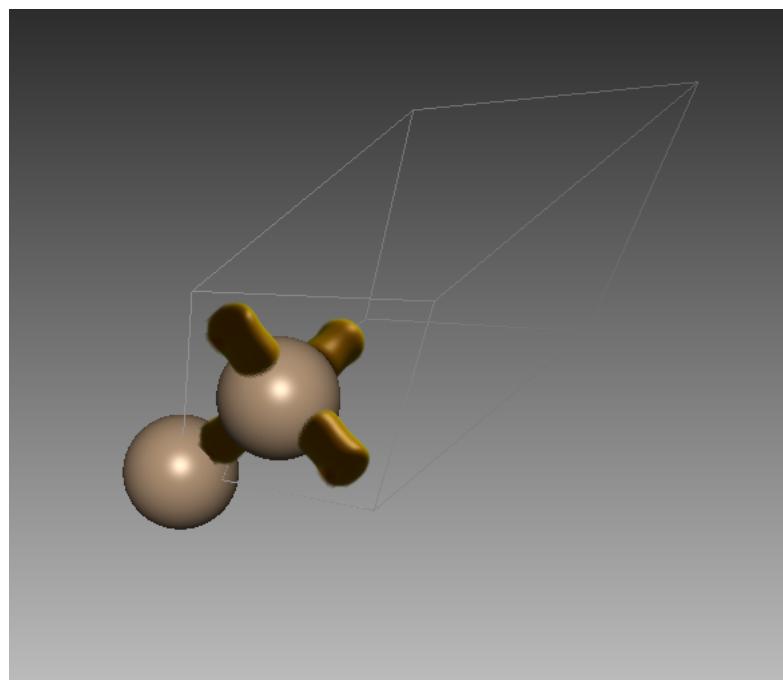
Figur 3 är en skärmbild från visualiseringen av elektrontäthet för aluminimum, Al. Här syns

att det är hög elektrontäthet kring och mellan Al-atomerna. Al är en metall och har alltså fria valenselektroner som delas av alla atomer. Vissa antydningar till bindningar kan urskiljas men är i jämförelse med de kovalenta bindningarna i figur 4 diffusa. Det relevanta är att elektrontätheten är hög mellan atomerna och inte koncentrerat till bindningarna.



Figur 3: Skärmbild från visualisering av elektrontäthet för aluminium, Al.

Figur 4 är en skärmbild från visualisering av elektrontäthet för kisel, Si. Si-atomerna har kovalenta bindningar mellan sig dvs. de delar elektronpar mellan sig. Bindningarna syns tydligt i visualiseringen eftersom elektrontätheten är hög just kring dessa bindningar.

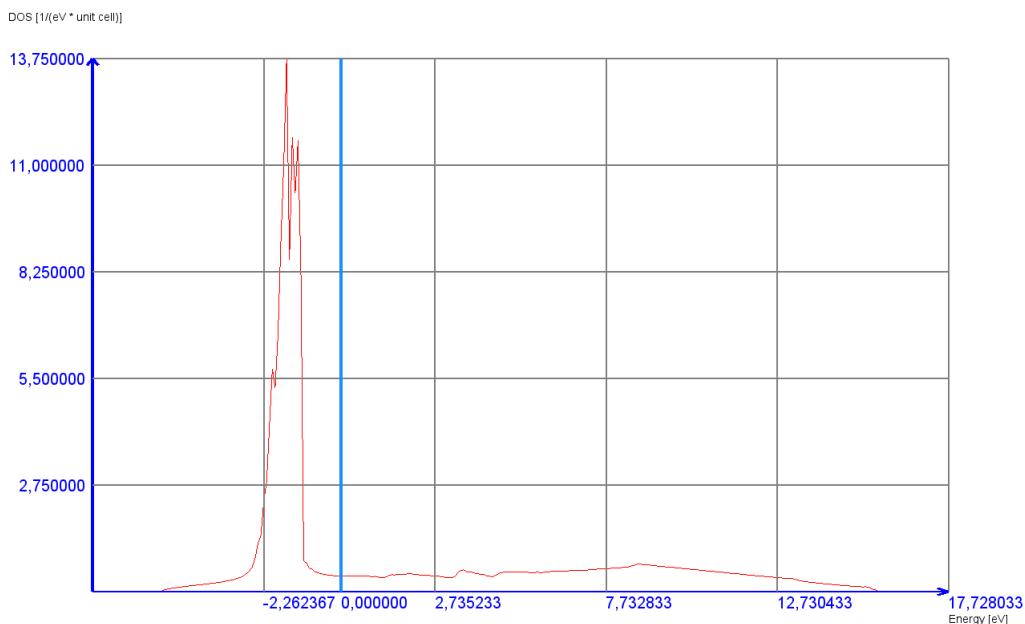


Figur 4: Skärmbild från visualisering av elektrontäthet för kisel,Si.

3.3 Tillståndstäthet

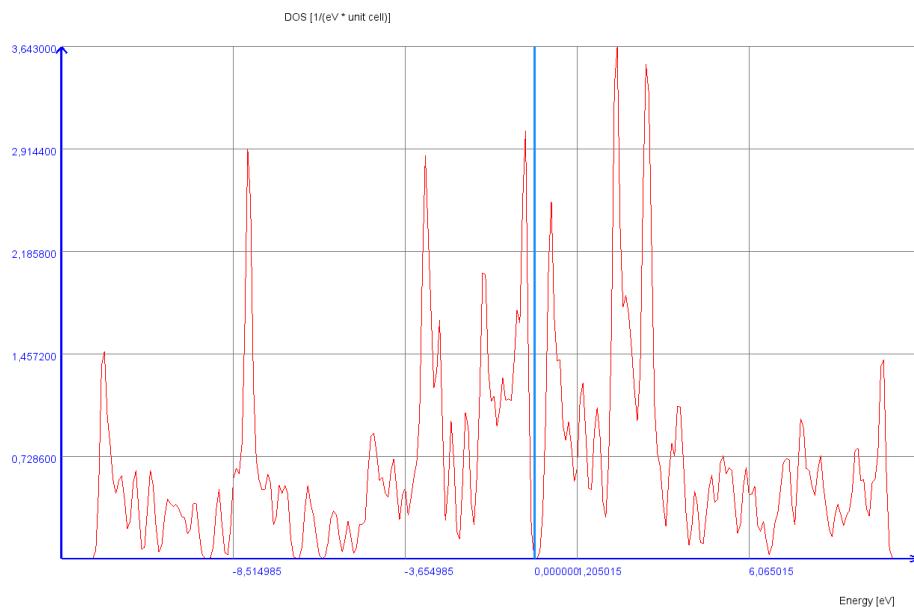
Två av egenskaperna som kan visualiseras med det framtagna verktyget är total och partiell tillståndstäthet. Tillståndstäthet beskriver antalet tillåtna tillstånd för olika energier.

I figur 5 ses en skärbild av en visualisering av den totala tillståndstätheten för koppar, Cu. Noll på x-axeln ligger vid Fermi-energin och är markerad med en blå linje. För en ledare är tillståndstätheten skild från noll vid Fermi-energin men hos en isolator och halvledare ligger Fermi-energin mitt i bandgapet mellan ledningsband och valensband vilket gör att tillståndstätheten är noll kring Fermi-energin. Cu är en metall och borde alltså ha en nollskild tillståndstäthet vid Fermi-energin och utifrån figur 5 ses att detta stämmer; även om tillståndstätheten är låg så går den aldrig ner till noll.



Figur 5: Skärbild från visualisering av total tillståndstäthet för koppar, Cu.

I figur 6 ses en skärbild av en visualisering av den totala tillståndstätheten för kisel, Si. För energi lika med Fermi-energi är tillståndstätheten i detta fall lika med noll, dvs. i bandgapet mellan ledningsband och valensband. Si är en halvledare vilket alltså bekräftas av denna figur. Bandgapet är ungefär 0,2 eV.

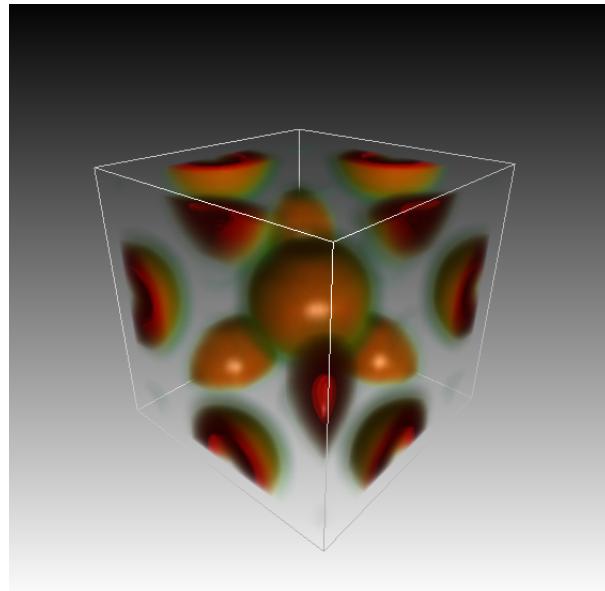


Figur 6: Skärmbild från visualisering av total tillståndstäthet för kisel,Si.

3.4 Elektronlokaliseringsfunktion, ELF

En av egenskaperna som kan visualiseras med det framtagna verktyget är ELF. ELF beskriver sannolikheten att hitta en elektron kring en referenselektron. Det kan också beskrivas som ett mått på hur svårt det är att få in ytterligare en elektron på en viss plats.

Figur 7 är en skärmbild av en visualisering av ELF för NaCl. Denna kan jämföras med elektron-tätheten för NaCl som sågs i figur 2 i kap. 3.2. Elektron-tätheten var hög kring Cl-atomerna men ELF är istället hög kring Na-atomerna.



Figur 7: Skärmbild från visualisering av ELF för natriumklorid, NaCl.

4 Diskussion

Att använda visualiseringar av olika ämnen kan bidra till ökad förståelse för olika materials egenskaper. Som har visats i denna naturvetenskapliga undersökning så kan visualiseringarna ge en förklaring till de olika ämnenas ledningsförmågor t.ex. kan dessa visualiseringar förklara varför ett ämne är en halvledare och ett annat en isolator.