FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Generierung and Visualisierung von Kristallstrukturen

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Gerald Michl
Matrikelnummer 01425673

an der Fakultät für Informatik
der Technischen Universität Wien
Betreuung: Dipl.Ing Tobias Klein

Wien, 17. September 2017

Gerald Michl	Tobias Klein

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Generation and Visualisation of Crystal Structures

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Media Informatics and Visual Computing

by

Gerald Michl
Registration Number 01425673

to the Faculty of Informatics
at the TU Wien
Advisor: Dipl.Ing Tobias Klein

Vienna, 17th September, 2017

Gerald Michl	Tobias Klein

Erklärung zur Verfassung der Arbeit

Gerald Michl
Töllergraben 3
8605 Kapfenberg

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 17. September 2017

Gerald Michl

Danksagung

Zuallererst möchte ich meinem Betreuer Tobias Klein danken, der mich seit Beginn der Arbeit bestmöglich unterstützt hat und bei Problemen meinerseits immer mit seinem Fachwissen zur Verfügung stand. Seine hilfreichen Tipps sowohl zur praktischen, als auch zur schriftlichen Ausarbeitung haben immer dafür gesorgt, die Arbeit in die richtige Richtung zu lenken. Zudem gilt mein Dank auch Josef Weinbub und Thomas Windbacher für die großartige Projektidee und die nachfolgende Zusammenarbeit, sowie Eduard Gröller und Ivan Viola für ihr konstruktives Feedback während des praktischen Teils und ihre inspirierenden Ideen, die auch in die Arbeit eingeflossen sind.

Auch Jakob Knapp möchte ich meinen Dank aussprechen, da es ein Vergnügen war mit ihm den praktischen Teil dieser Arbeit gemeinsam zu implementieren. Trotz mancher schwer zu lösender Probleme, war es uns zu zweit immer möglich eine passende Lösung zu finden.

Des Weiteren danke ich allen die mich während meines bisherigen Studiums unterstützt, betreut und als Freunde oder Kollegen begleitet haben. Besonders hervorzuheben sind dabei meine Eltern, die mir vom ersten Tag an zur Seite standen und mir so das Studium erst ermöglicht haben.

Acknowledgements

First of all i want to thank my supervisor Tobias Klein, who supported me best possible throughout my work on the thesis and has always been on hand with his expertise when problems occurred. His helpful tips not only for the implementation but also for the written part of the thesis have always ensured to steer the work in the right direction. In addition, my thanks go to Josef Weinbub and Thomas Windbacher for the great project idea and the following collaboration, as well as Eduard Gröller and Ivan Viola for their constructive feedback during the practical part and their inspiring ideas, which have also been incorporated in the work.

I also want to express my gratitude to Jakob Knapp because it was a pleasure to implement the practical part of the thesis with him. Despite some difficult problems to resolve, we have always been capable of finding a proper solution.

Moreover, i would like to record my thanks to everyone who supported, supervised and accompanied me as friends or colleagues during my studies so far. Worthy of special mention are my parents, who stood by me since day one and enabled my studies in the first place.

Kurzfassung

Die Halbleiterindustrie befindet sich seit ihrer Entstehung vor fast 60 Jahren in einem unaufhaltsamen Aufwärtstrend, der mit immer fortschrittlicher Technologie einhergeht. Die Entwicklung immer kleinerer Chips ermöglicht es, kleine Alltagsgegenstände wie Schlüssel oder Uhren in die digitale Welt einzubinden. Ein wichtiger Baustein dieser Chips sind Transistoren, welche im Inneren aus mindestens einem Halbleiter aufgebaut sind.

Diese Arbeit beschäftigt sich genau mit diesen Halbleitern und präsentiert ein mit Unity3D entwickeltes 3D-Spiel, welches es möglich macht in das Innere eines Transistors zu blicken und diesen interaktiv zu erkunden. Der Spieler hat dabei die Kontrolle über ein Elektron, welches durch den Kristall fliegt. Der Flug durch den Kristall ist aber nicht frei von Hindernissen. Inmitten der Struktur finden sich immer wieder Oxidschichten, welche bei zu geringer Geschwindigkeit ihre Durchquerung verhindern und das Elektron abstoßen, nur bei entsprechend hoher Geschwindigkeit ist ein Durchdringen möglich. Der Fokus liegt dabei darauf, dass der User über verschiedene Parameter die Szene, zum Teil auch interaktiv, verändern und damit alle möglichen Halbleiterstrukturen generieren kann. Dies eignet sich nicht nur zum Betrachten und Lernen verschiedener Strukturformen und ihren Symmetrien, sondern verändert auch das Spielerlebnis und den Schwierigkeitsgrad.

Die Implementierung basiert auf cellVIEW, ein Tool zur Visualisierung umfangreicher Molekularstrukturen, welches uns das performante Rendern von Halbleiterstrukturen mit zum Teil weit mehr als 15 Millionen Atomen ermöglicht, was, wie die Ergebnisse am Ende der Arbeit zeigen, wirklich beeindruckend große Kristalle zur Folge hat.

Abstract

The semiconductor industry has been experiencing an inexorable upward trend since its emergence almost 60 years ago, with ever since advancing technology. The development of continually smaller chips makes it possible to integrate small everyday objects such as keys or watches into the digital world. An important component of these chips are transistors, which are built up internally of at least one semiconductor.

This thesis deals with these semiconductors and presents a 3D game developed with Unity3D, which makes it possible to take look inside a transistor and explore it interactively. The player therefore has control of an electron flying through the crystal. However, this flight through the crystal is not free from obstacles. In the middle of the structure there are consistently occurring oxide layers which, if the electrons speed is too low, prevent the passage and repel the electron. Only at a correspondingly high speed is it possible to tunnel trough them. However, the focus is on the fact that the user can adjust the scene, partly interactively, by means of various parameters and thus can generate all possible semiconductor structures. This is not only suitable for viewing and learning different structural shapes and their symmetries, but also changes the playing experience and the degree of difficulty.

The implementation is based on cellVIEW, a tool for the visualization of extensive molecular structures, which allows us to perform high-performance rendering of semiconductor structures of often far more than 15 million atoms, which implies, as the results at the end of the work show, really impressively large crystals.

Contents

CHAPTER 1

Introduction

With the upcoming generation of processors having more than 100 million transistors per square millimetre of chip the microelectronics industry has reached another significant milestone. This number gets even more impressive considering that ten years ago the density was only about 3 million transistors per square millimetre, which means it increased by about 3000 percent in that short amount of time [Cou17]. This rapid increase shows that the semiconductor industry is still the key for future electronics development especially considering the "Internet of Things" and mobile devices in general, where size is the limiting factor and chips have to be as small as possible. A look inside a chip reveals that the its main component are transistors. These transistors themselves consist of (one or more) semiconductors, which can be considered the core elements and enable the control of electronic signals and electronic power. All semiconductors have an inner crystalline structure, which varies depending on the type of semiconductor.

1.1 Motivation and Goals

The Institute for Microelectronics of the TU Wien are experts on this topic and want to make people who visit science2public events (e.g. “Tag der offenen Tür”, “Lange Nacht der Forschung”) aware of the importance of semiconductors, show them what semiconductors look like from the inside and how they are composed in order to give the people an idea of what their work is all about. In order to do so they came up with the idea of creating a game, which demonstrates the above mentioned points in an interactive 3D scene and can be played in virtual reality. This allows the people to have an immersive experience of a semiconductor and explore it by themselves.

The goal of this project is the creation of a 3D virtual reality game based on cellVIEW [MAPV15] using the Unity3D game engine [Uni17]. In this game, the player controls an electron which flies through the crystal structure of a semiconductor. This crystal structure can be altered by the user with several input parameters resulting in different

kinds of crystals being displayed. The navigation through the structure should be simple and quickly to get used to since the main purpose of this game is showing it at public events like science2public. While flying through the structure it should be possible to adjust the temperature of the scene which results in reduction or increase of the vibrations of the atoms. When colliding with the atoms the electron should be deflected or pushed away. The players goal is to get through the structure safely which means without touching any atoms and also as quickly as possible. However, there are obstacles in-between the crystal so called "oxide layers". In order to get through them the electron requires a certain speed, otherwise the oxide repels the electron. To attract as many people as possible it is also important that the scene looks impressive and visually appealing. Figure 1.1 shows an initial draft screenshot of an example crystal structure.

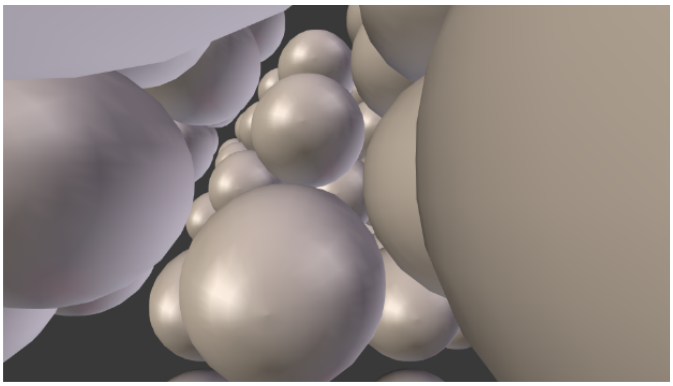

Figure 1.1: A screenshot of an example crystal structure opened in Blender.

While originally the project was proposed to be a single master thesis, it got split up into two separate bachelor theses each one of them focusing on different parts of the project. This sub-project focuses on the generation and visualization of the crystal structure as the other features, which include the interaction with the scene and the collisions of the electron with the atoms of the crystal structure, are developed by another student [Kna17].

1.2 Contribution

The main contribution of this thesis can be split into three major parts:

- Generation of large-scale crystal structures and storing them in performance-enabling data-structures
- High performance rendering of large-scale crystal structures in VR
- Generation and Visualisation of oxide layers and implementation of the "tunneling" through the oxide respectively the reflection of the electron off the oxide

The generation of the crystal involves the development of an algorithm that determines the positions of all the atoms of the structure and stores them with other for the rendering necessary information such as radius or colour of the atoms in a performance-enabling data-structure. Since the displaying of millions of atoms is a key factor to make the scene visually impressive and entertaining to explore, fast rendering is crucial. The reason why a large number of atoms is necessary to make the scene impressive and worth exploring is that looking at a structure in virtual reality, which seems to be larger than the player himself/herself arouses the interest to get into it and explore it, while looking at a small structure, where the player is able to see through it straight away, this feeling might not be as strong. The rendering of such a huge number of atoms can be achieved using cellVIEW, a tool that is used to render large molecular structures and capable of rendering millions of atoms while maintaining a decent framerate. The contribution concerning rendering is not only limited to feeding cellVIEW with the data but also to make sure that certain parameters can be interactively changed while running the game and. Furthermore the colouring of the atoms and the way the radius is determined has to be adjusted from cellVIEWs standard implementation, because we want to render each of the generated atom types in a different colour and with a different radius.

To create an accurate representation of oxide layers in semiconductor crystals this software uses a wall of atoms which do not take part in interaction as the other atoms do but act as single object which called oxide plane and has the above mentioned properties of reflecting and letting the electron through depending on its speed.

1.3 Structure of the Thesis

The structure of this thesis is as follows: Chapter 2 gives an introduction into the topic of crystal structures, oxide layers and cellVIEW. Different Crystal structure visualisation software and other to this topic related past work is presented in Chapter 3. The Methodology and Implementation as the main part of this thesis are explained in Chapter 4. The Results are presented in Chapter 5 including a critical reflection on them and finally Chapter 6 concludes the work and discusses future work.

CHAPTER 2

Background

In order to provide an overview for the subsequent chapters, the basics of crystal structures, oxide layers and the cellVIEW framework are described in the following. Therefore the first section explains the composition and the different types of crystal structures and gives examples of where they are used. The second section discusses oxide layers and their relation to crystal structures including the electron tunnelling through the oxide. The last section briefly addresses cellVIEW and outlines the reasons behind its use in this project.

2.1 Crystal Structures

Firstly, to get an idea of what a crystal structure is, which is important for all further explanations, we have to define a few terms. Then we proceed with explaining the different cubic crystal structure types and finally the primitive vectors of all mentioned types are listed including some example lattice constants.

2.1.1 Definitions

A crystal structure itself is a recurring pattern of atoms, which can be compared to a three dimensional wallpaper [HH91]. It consists of one or more unit cells. These cells are defined as the smallest number of atoms needed to describe the whole crystal structure. This is possible because the structure is periodical, i.e. a recurring pattern. This pattern is called the crystal lattice. To understand the difference between the crystal lattice and the crystal structure Figure 2.1 shows a two-dimensional crystal structure in (a) and the corresponding lattice to it in (b). The atom labelled O is the origin of the lattice and can be chosen arbitrarily but conventionally lies on an atom. The atoms A, B, C, D and E, imagining an infinite lattice, would also be suited as origin because their neighbour atoms are situated exactly as the origins neighbours.

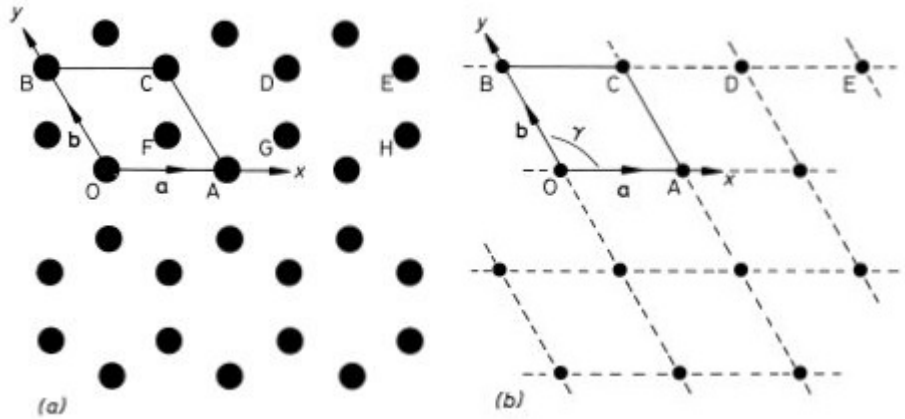

Figure 2.1: Comparison between a 2D crystal structure (a) and its corresponding lattice (b). The origin O and its neighbour atoms are labelled with capital letters, the lattice parameters and the axes in lower-case characters. [HH91]

However, atoms F, G and H cannot be chosen as origin for the same lattice because their neighbour atoms are positioned differently. This leads to the second lattice in this structure which has an origin F, which is shifted by a vector from origin O. The atoms in this lattice are G and H and could also be chosen as origin for this lattice. This means that all atoms having the same neighbourhood as the origin (and therefore also with all the other atoms) belong to the same lattice. In other words if all the relative positions of all neighbour atoms of an atom are exactly the same as the ones from the neighbour atoms of the origin then the atom belongs to the same lattice as the origin. The conclusion is that one crystal structure can have more than one possible lattice depending on the choice of origin where the different lattices are just shifted by a vector.

Figure 2.1 (a) is just an example lattice, since there are infinite different possibilities of how lattices can be arranged. As Figure 2.1 shows, there are the vectors $\vec{a}$, $\vec{b}$ and the angle γ, which define the layout of the lattice. γ is the angle in degrees between the so

called "primitive vectors" or "lattice vectors" $\vec{a}$ and $\vec{b}$. Usually just the lengths of these vectors, referred to as "lattice constants", are given in Ångstrom ($1\text{Å} = 10^{-10}m$). The abstract formula to calculate all lattice points is therefore $r = u \cdot \vec{a} + v \cdot \vec{b}$, where u and v are integer values [Kit05b] [HH91].

The definition of a three-dimensional crystal features not two but three lattice vectors $\vec{a}$, $\vec{b}$ and $\vec{c}$, which equals one vector per axis. Since there are three vectors from one origin, there are also three angles between them, which are named α, β and γ. The lattice points can be reached by plugging in all possible integer values for u, v and w in $r = u \cdot \vec{a} + v \cdot \vec{b} + w \cdot \vec{c}$ [HH91].

The unit cell is the only thing needed to generate the whole structure, which implies that a description of this unit cell is necessary. Conventionally this description contains just the positions of the atoms that lie within the unit cell. When taking a closer look at Figure 2.1 (a) where the bounding of the unit cell is defined by OACB, it can be seen that F also lies within these bounds and therefore has to be mentioned in the description of this unit cell. While the origin is of course also part of the unit cell, the atoms A, C and B on the other hand can be considered as outside the unit cell because they span a unit cell themselves, which is another reason why they are also suited as origin of the lattice. Atom F can be described by its position relative to the origin O, which is $(\frac{2}{3}, \frac{1}{3})$ but also as vector of the form $r = \frac{2}{3} \cdot a + \frac{1}{3} \cdot b$, since the origin is always at $(0, 0)$ respectively $r = 0$. The positions of atoms O and F combined are the so called "basis" and is usually written with the atom types to each position, e.g. $C(\frac{2}{3}, \frac{1}{3}), C(0, 0)$ for a so called "two-atomic" or "diatomic" carbon basis [HH91]. Figure 2.2 shows that the basis applied to every lattice point results in the complete crystal structure.

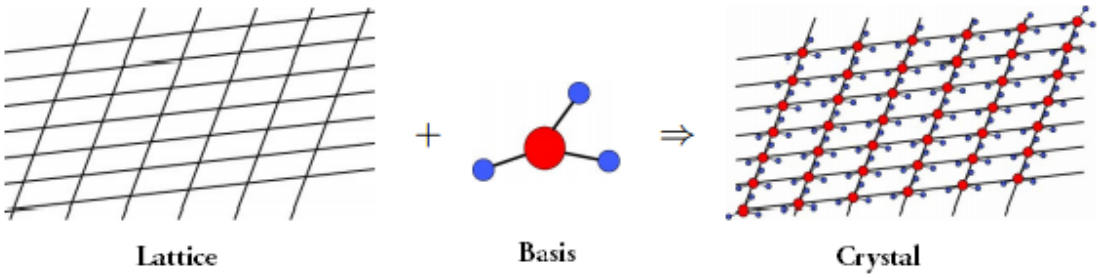

Figure 2.2: Starting with the lattice which is shown on the left and taking the basis consisting of one or more atoms, which is shown in the middle, a crystal structure is constructed by applying the basis at every lattice point, which is depicted on the right. [Kit05a]

2.1.2 Cubic Structure Types

With three different angles α, β and γ as well as three different vectors $\vec{a}$, $\vec{b}$ and $\vec{c}$ an overall of seven three-dimensional crystal types exist. However, since the cubic structure is one of the most featured in semiconductors and likewise one of the simplest of them

all, this thesis focuses on this one crystal type [VZ04]. What makes the cubic structure so special and simple is that all three vectors are the same in length ($|\vec{a}| = |\vec{b}| = |\vec{c}|$) and the three angles are also of the same size $\alpha = \beta = \gamma$ and equal to 90° [Kit05b].

The next few paragraphs present the different types of cubic crystal structures, where the type depends on their the number of atoms the unit cell contains and how theses atoms are positioned in it.

Simple Cubic Structure (sc) This is, as the name already suggests, the simplest structure with only one atom in the basis ("one-atomic" or "monatomic" basis). Figure 2.3 (a) shows the unit cell of this structure. The only atom that belongs to this cell is darkened in the image, all the other atoms already belong to a neighbouring cell.

Body Centred Cubic Structure (bcc) The body centred cubic structure is basically the same as the simple cubic structure but there is another atom in the centre of the cell as shown in Figure 2.3 (b). The atoms belonging to this cell are darkened again, the atoms in a lighter shade belong to the surrounding cells.

Face Centred Cubic Structure (fcc) Again, starting with the simple cubic structure, the face centred cubic structure is just an extension of it. Instead of adding an atom in the middle of the cell, this time the atoms are added in the middle of each face of the cube. This makes the unit cell containing four atoms, as shown in Figure 2.3 (c).

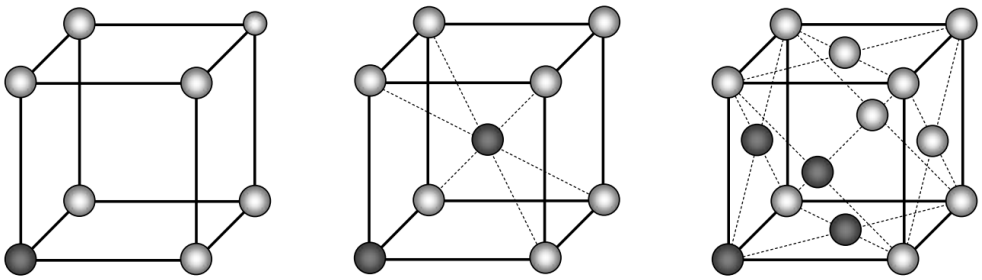

Figure 2.3: Simple (a), body centred (b) and face centred cubic structure (c) compared. The simple structure has just one atom at each cube vertex, the body centred has another atom in the cubes centre, the face centred is a simple cube with atoms in the middle of each of the cubes faces. All basis atoms are darkened. [Hun09]

Diamond Cubic Structure (dc) The diamond structure is a special case of the fcc structure and therefore not an actual type. However there are many important elements such as carbon or the semiconductors silicon and germanium, which have this diamond structure. This structure features a two-atomic basis because it contains two nested,

face centred cubic structures resulting in the unit cell containing eight atoms, which is depicted in Figure 2.4.

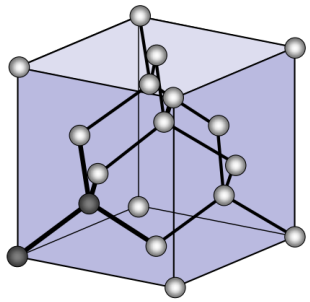

Figure 2.4: The diamond structure with its diatomic basis, which is recognizable by the darkened atoms. From each basis atom a face centred cubic structure emerges. [Hun09]

2.1.3 Primitive Vectors and Lattice Constants

These two terms, primitive vectors and lattice constants, are linked since the definition of the primitive vectors, no matter which type of the above listed, always contains the lattice constant. The reason for this is that different materials have different lattice constants, which results in tighter or looser compounds while having the same structure. Table 2.1 shows not only the primitive vectors and the number of atoms in the unit cell of each structure abbreviated n, but also one example material per structure type with its lattice constant a.

Structure type	n	Primitive Vectors $(\vec{a_1}, \vec{a_2}, \vec{a_3})$	Element (a in Å)
Simple	1	$a * \begin{bmatrix}1\\0\\0\end{bmatrix}, a * \begin{bmatrix}0\\1\\0\end{bmatrix}, a * \begin{bmatrix}0\\0\\1\end{bmatrix}$	None (only with monatomic basis)
Body Centred	2	$\frac{a}{2} * \begin{bmatrix}-1\\1\\1\end{bmatrix}, \frac{a}{2} * \begin{bmatrix}1\\-1\\1\end{bmatrix}, \frac{a}{2} * \begin{bmatrix}1\\1\\-1\end{bmatrix}$	Fe(2.87), Li(3.49), Na(4.23)
Face Centred	4	$\frac{a}{2} * \begin{bmatrix}1\\1\\0\end{bmatrix}, \frac{a}{2} * \begin{bmatrix}1\\0\\1\end{bmatrix}, \frac{a}{2} * \begin{bmatrix}0\\1\\1\end{bmatrix}$	Ag(4.09), Au(4.08)
Diamond	8	None (2 face centred lattices)	C(3.57), Si(5.43), Ge(5.66)

Table 2.1: The different crystal structure types with their number of atoms in the basis n, their primitive vectors $\vec{a_1}, \vec{a_2}, \vec{a_3}$ and a few example elements with their lattice constants a in Å. [Kit05a]

2.2 Oxide Layers

This section gives a short overview of oxide layers and their purpose. It does not cover any physical details since they are in no relation to this project. Furthermore a brief explanation of "electron tunnelling" and the negative effects of it is provided.

Semiconductors have an inner crystalline structure and are used as component in transistors. There are several different types of transistors, each of them consists of different components and has different usage. The two main transistor types are "bipolar junction transistors" (bjt) and "field-effect transistors" (fet), which again have a vast number of sub-types. The most popular of them all is the so called "MOSFET", which is an acronym for "metal–oxide–semiconductor field-effect transistor" and this is where the oxide layers come into use. Figure 2.5 shows the basic structure of a MOSFET and reveals the blue oxide layer as isolator between the two n-doped yellow areas (the semiconductor has more electrons than usual).

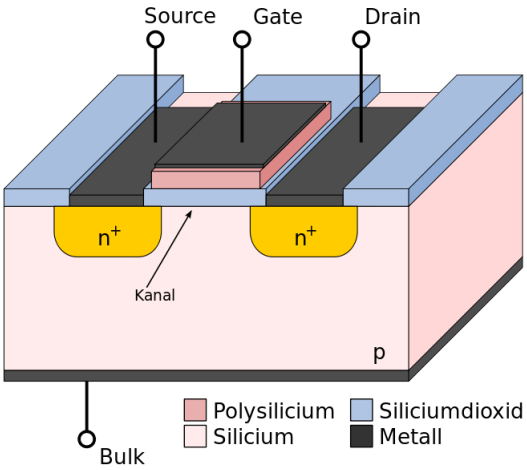

Figure 2.5: Architecture of a MOSFET with four connections: source, gate, drain and bulk. Positive voltage to the gate results in a flow of electricity from source to drain. The legend at the bottom shows which colour indicates which material. [Wik09]

If positive voltage is applied to the gate the electrons of the p-doped light pink area (the semiconductor has electron holes - less electrons than usual) get attracted and the transistor allows current to pass from source to drain, otherwise the oxide an the p-doped area are destined to block that connection [LS98].

However there is a phenomena called "electron tunnelling" that circumvents that barrier. In general the oxide can be seen as potential barrier between the electrons of the channel from the gate. With the oxide layer of a transistor being nowadays less than 1.5nm tick, the electrons have a chance to tunnel through that barrier. This current is called electrical leakage and can be the reason for improper circuit behaviour or above-average power consumption in standby [Cha13] [ESA05].

2.3 cellVIEW

cellVIEW is a tool by Le Muzic et al. [MAPV15], which allows to render large-scale biomolecular structures containing more than a billion atoms, based on the Unity3D engine. The system does not only render these structures but in fact renders everything in real time and allows for interaction with the scene. This is possible because of several optimization techniques.

Firstly, a level-of-detail scheme has been used, which not only improves the performance but also the appearance of the scene, since there is the appropriate complexity for each zoom level. Secondly, the data needed to render the scene is stored on the GPU memory, which is a lot faster than transferring the data form CPU to GPU in every frame. Therefore, different types of proteins containing the positions of one up to thousands of atoms, can be stored and then instanced on the desired location in the scene. Lastly, billboards, as described by [DN09], are used to render the whole scene, instead of rendering complex meshes. The shading of the atoms is also suited for high performance rendering. Since cellVIEW just renders the atoms with billboards, the shading is completely done with post-precessing effects such as ambient occlusion and edge detection respectively enhancement.

This high-performance rendering of millions of atoms is what our project has to be capable of as well, which is why we decided to take cellVIEW as a basis to start with and build our project on top of it. The Unity3D engine would allow us to easily add primitives with different shaders and other game objects such as audio sources or texts to our scene independently from cellVIEW.

CHAPTER 3

Related Work

There are several different tools to generate, model, visualise or animate crystal and molecular structures. This chapter introduces some of the most well known of these tools and reveals their strengths as well as their limitations. Furthermore an overview of previous work regarding crystal structures and virtual reality is given.

3.1 Crystal and Molecular Structure Tools

All of the software presented in this section are PC only and are either for free or offer demo versions, which have been used to test their functionality. A list of available tools can be found under the following link: `https://www.iucr.org/resources/other-directories/software`

3.1.1 VESTA

The first tool we present is called VESTA, which is an acronym for "Visualization for Electronics and Structural Analysis" [Mom17] [MI11]. It was developed by Koichi Momma and Fujio Izumi from the National Institute for Materials Science, is available for free and supports a variety of different file types for opening and saving structure and volumetric data, which makes it compatible to nearly every common crystal software. Despite from opening already existing data it also supports generating new structures with several input parameters, which include the structure type (e.g. cubic, hexagonal, ...), the space group, which determines the inner structure, the lattice parameter in Å and the three angles α, β and γ in degrees. To display the structure a new atom type (e.g. Carbon, Silicon, ...) has to be created, which is then used for the crystal. The choice of the atom type automatically adjusts the radius and the colour of the associated atoms but can be changed manually, as well as all other parameters, afterwards.

In order to add a so called "external morphology", planes can be added to the structure, which usually lay on the border of the structure and add a visual treat as Figure 3.1 shows.

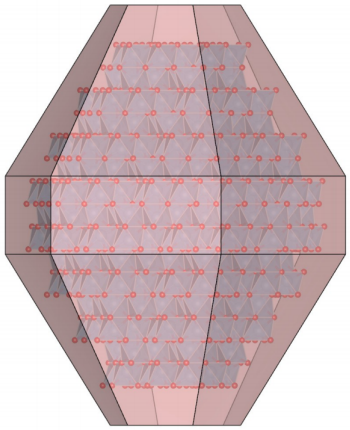

Figure 3.1: A crystal constructed in VESTA with external morphology added. [MI14]

Once created, different viewing options are available, including "Ball-and-Stick", "Space-filling" and "Wireframe". It is possible to rotate, zoom and animate the structure, as well as choose between parallel and perspective projection. Moreover, detailed information about the structure itself but also about each atom and plane can be displayed. Tools for measuring distances and angles are available from the toolbar on the left. The only point to mention on the negative side is that some of the menus are flooded with checkboxes, radiobuttons and spinners, which exacerbates the handling in general.

3.1.2 CrystalMaker

The CrystalMaker software suite consisting of CrystalDiffract, SingleCrystal, CrystalViewer and CrystalMaker X is one of the most complex crystal toolkits available and was founded by David Palmer and his wife Shirley at the University of Oxford Begbroke Science Park [PP17]. While CrystalMaker X and CrystalViewer are the heart of the suite and offer similar functionality to VESTA, CrystalDiffract and SingleCrystal are more advanced tools and focus on the diffraction properties of crystals. Both CrystalMaker X and CrystalViewer can be controlled with the Leap Motion Controller, which allows for rotating and scaling with hand gestures. The following paragraphs give a short overview of the two for this thesis worthwhile tools.

Crystal Maker X CrystalMaker X allows to create complex crystal structures from scratch with just a few clicks and comes with a lot of example structures, which demonstrate the capabilities of the software. Menus are being kept clear and simple, which makes the handling easy and quickly to learn. A separate render menu makes it possible

to adjust reflectance, lightning and perspective, add depth fading or switch to 3D Stereo mode. The only downside is that it is an commercial product and not available for free.

CrystalViewer This light tool is available for free and allows to open all files created with CrystalMaker X, which is intended for students to be able to open their teachers files and learn about the over 120 included structures.

3.1.3 Diamond

Diamond is a software created by the Germans Klaus Brandenburg and Holger Putz, both chemists at the University of Bonn. It offers similar functionality to CrystalMaker X and VESTA but hides it behind a very complex and unclear UI. The creation process is handled by an assistant menu, which makes it look more complex than it is. The upside is that once got used to the UI, the software offers powerful manipulation options, which include adding single atoms at any positions, adding atoms bonds between two arbitrary atoms and an own video recording function. Furthermore detailed properties and informations about the structure itself as well as each of the atoms and atom bonds can be displayed in a separate window [BP17].

3.2 Crystal Structures and VR

While software for creating, viewing and manipulating crystal structure is mainly available for PC, software with virtual reality integration is a mobile only segment. All of the presented apps are available for free and can be downloaded from Google Play Store. In order to get the apps running properly Google Cardboard or another smartphone VR headset is required.

3.2.1 Arthea

The Virtual Reality software startup Gwydion and its founder Duncan Abbot from the University of Michigan developed the mobile app Arthea [AVR17] [Abb17]. Its goal of the software is to help students understand and visualize the different kinds of crystal structures, as Prof. Joanna Millunchick, who has already used it in class, points out in the apps presentation video. From the menu different standard models can be selected, new structures can be imported and it is possible to switch between VR and AR. In Augmented Reality, which is a mixture between the real and the virtual world, the phones camera is used to detect coloured sheets, which function as indicators for rotating and scaling the structure, another sheet is the anchor point where the crystal resides.

3.2.2 PROtein VR and MoleculE VR

PROtein VR [PRO16] and MoleculE VR [Mol16], both developed by AppMinded Ltd, are mobile virtual reality apps intended for educational use. While PROtein VR is mainly a PDB-Viewer where the user is in control, similar to Arthea, MoleculE VR takes one on

a virtual reality tour trough a cell. The tour consists of five chapters through different molecules, explaining what they are, of what they consist and how the communication in a cell works in general. PROtein VR additionally requires a bluetooth controller to be able to move the viewed structure, MoleculE VR does not need any controls since its an automated tour.

CHAPTER 4

Generation and Visualisation of Crystal Structures

4.1 Methodology

This section presents the methods, which have been used to develop the crystal structure VR application. It involves the generation, structuring and rendering of the crystal as well as a discussion about different approaches to build the oxide planes including the reasons behind the choice of approach.

4.1.1 Generation of the Crystal Structure

In a first step the structure of the crystals with all its atoms has to be constructed. Since there are many different kinds of crystal structures, and it should be possible to create all of them, an appropriate kind of input has to be found from which the program can generate the desired output. This output is a list containing 3D points describing the centre of the atoms. The easiest method and likewise the one that needs the least input parameters to describe a crystal structure and get the 3D coordinates of the atoms is to specify the primitive vectors. For each type of crystal structure exist three of them and by combining them, all atom positions of a structure can be described. The formula to do so is $\vec{R} = n_1 a_1 + n_2 a_2 + n_3 a_3$, where a_1 , a_2 and a_3 are the primitive vectors and n_1 ,n_2 and n_3 are arbitrary integers [HH91]. The easiest way would be to just specify the number of atoms wanted and calculate them with the above mentioned formula. However, there is also a way that can make use of the repetitive structure of the crystal by using the primitive cell. This cell can be described with the primitive vectors by just using 0 and 1 for n_1 , n_2 and n_3 [Geo14]. The second input parameter that is needed is the side length of this primitive cell which is the lattice constant in Ångström. It is needed because it defines not only the length of the primitive cell but also the proportions of the

lattice to the atoms i.e. how far the atoms are away from each other. Having created a primitive cell, it is an easy task to replicate it and generate huge crystal structures consisting of thousands of atoms. Therefore, the user should be able to specify how many primitive cells the scene has to contain in each of the three directions (x, y, z) and the generator just takes that input and multiplies the primitive cell with the given number of times in each direction starting with x.

However, not all crystal structures can be created with this method. For example, the diamond cubic crystal structure has a diatomic basis. This means that it is not enough to just start at (0, 0, 0) (which is the conventional basis) and use the above formula to create the structure. To fully generate a crystal structure needs a second "start atom" which is in case of the diamond structure at (0.25, 0.25, 0.25) relative to the basis [Kit05a]. To allow structures with a basis consisting of two or more atoms the user should be able to enter the number of atoms in the basis and its coordinates.

4.1.2 Building the Oxide Layers

The initial concept for the oxide layers was to put them in the middle of the crystal. This would allow the user to explore half the crystal before having to tunnel through the oxide. However, since we planned to connect several crystals with each other anyway, the idea came up to position the layers in-between theses crystals to have a clear boundary, which also increases the number of layers that need to be generated and rendered.

For the visualisation there were three different possibilities of how the oxide layers could look like. The first idea was to visualize the oxide layers as planes since this comes closest to what we see as a layer. However, one plane per layer would not be enough because Unity planes can just be viewed from the top with the standard Unity shader. From the bottom they are invisible because of the back face culling that is enabled by default. To prevent that the backface culling could be deactivated just for the planes. However, this would result in incorrect lightning at the backside. A custom shader would be one possible solution to this problem, the effort necessary is not worth it though, since the easier and far more convenient solution is to simply add a second plane at the same position but facing opposite in direction. This means that the number of meshes is twice as high though and would possibly influence the performance not only in the rendering but also in the generation process because of our decision to add a plane in-between every crystal block, which could result in about a thousand planes (details can be found in section 4.2.2). The second approach features one cuboid as oxide layer. We thought of it because the toughness of the oxide, which should be an adjustable parameter, could be reflected in the thickness of the cuboid in flying direction.

The third and last approach is to use atoms, as in the crystal structure, to build the oxide layer. Literature research revealed that this approach reflects the representation of oxide layers best as depicted in Figure 4.1, which is why we decided to use this approach and discard the other two. The additional number of atoms is not a concern since it is a relatively small compared to the atoms in the structure itself.

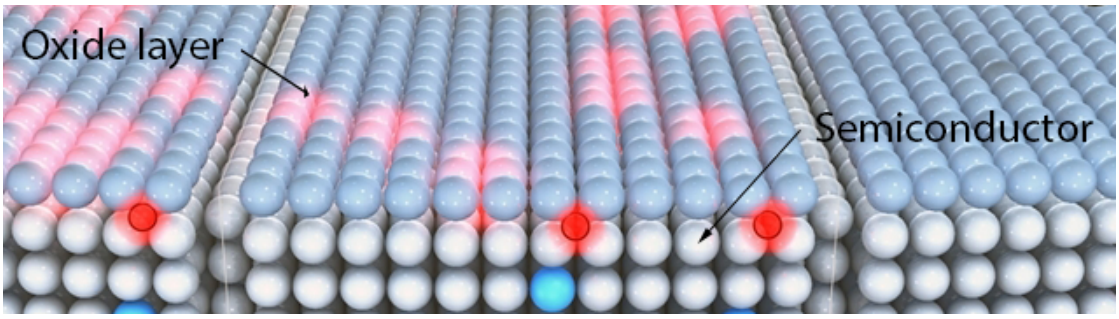

Figure 4.1: An oxide layer (grey blue atoms) residing on top of a semiconductor. This schematic depiction was the model for building the oxide layers (graphic adapted from [ANe09])

4.1.3 Displaying the Atoms

As already mentioned in Section 2.3, the project is based on cellVIEW, which provides a fast rendering mechanism for a large number of atoms. To be able to render the crystal structure generated in the first step, a special shader is needed which can render atoms just from a position in the 3D space. This shader is also supplied by cellVIEW but has to be altered slightly to fit our needs in this project because cellVIEW loads respectively calculates its atom colours and radii automatically, which we want to leave to the user to adjust, since it is not possible to set the element type for the atoms and therefore use the common van der Waals radius. This lets the user adjust the atoms radius arbitrarily, which allows for different scaling options (e.g. make scenes bigger by doubling the radii and the lattice parameter) and also adjust the proportion between the oxide atoms and the atoms of the crystal structure.

4.2 Implementation

This section gives detailed insights into the implementation. The first subsection describes the generation of the structure and the oxide planes as well as the collision of the electron with the oxide layers. Therefore not only textual explanation but also a code example is given to get a better view into the project. The organisation of the crystal structure is described in the second subsection. The third subsection focuses on how the collision of the electron with the oxide has been implemented. A listing of the adjustments in the render and colour shader of cellVIEW are addressed in the fourth subsection. Finally, the method used to detect changing parameter values is explained in the last subsection.

4.2.1 Generation Process

The generation process of the crystal structure is the first step in the initialisation of the game, which is executed on every start-up. Therefore a unit cell is created using the method described in Section 4.1.1 and shown in detail in Algorithm 4.1.

Algorithm 4.1: The generation algorithm for a unit cell

Input : The three primitive vectors $\vec{a}$, $\vec{b}$ and $\vec{c}$, an array *basisAtoms* containing the atoms of the basis with their colour, radius and position, and the lattice constant a

Output : A list *atomlist* containing all the atoms of the unit cell

```
1  foreach basisAtom ∈ basisAtoms do
2      for u ← 0 to 1 do
3          for v ← 0 to 1 do
4              for w ← 0 to 1 do
5                  pos = (basisAtom.position + u * a + v * b + w * c) * a;
6                  if pos ∉ atomlist then
7                      atomlist.add(new Atom(pos, basisAtom.radius,
                         basisAtom.colour));
8                  end
9              end
10         end
11     end
12 end
```

The next step is to extend the structure according to the users input. This input contains three length parameters, one for each direction. In each direction the unit cell gets duplicated as often as the length parameter requires. For example if the length parameter for the x-direction is three the unit cell gets duplicated three times in that direction. After the atoms for the crystal structure have been generated and added to the list of atoms, the oxide layers are created. The user input for them are the radius and the colour of the oxide atoms and the oxide toughness (see 4.2.3 for details). This procedure calculates the number of atoms that fit into this area, using the x and y size of the structure and the radius of the oxide atoms, when packed closely together and adds all of them to the list of atoms, with their z-position half a lattice parameter behind the crystal. It has to be mentioned that our oxide layer is just one atom thick instead of the 1.5nm, which would be around five atoms, because the collision respectively the tunnelling of the electron through the layer would be confusing and easy to get lost in for the player.

4.2.2 Organising the Crystal Structure

As already mentioned in the last section, the first step is to create a unit cell, which is then extended. However, to get even bigger and more impressive structures, further

extensions of the structure are possible. So far only one crystal with a oxide layer behind it has been created. In order to have multiple crystals and oxide planes this crystal can be duplicated as well. Therefore, again, the user can define three parameters each for one direction. But instead of duplicating all the atoms and add their positions to the list, cellVIEW lets us define instances and instance positions. Each instance type contains a list of atoms, which can the be instanced at arbitrary positions. We define those positions in a way that the crystal in x and y direction is seamless and in z direction that the new crystal starts directly after the end of the oxide layer of the one before (see Chapter 5 for images of the results).

Once calculated the atom positions and instance positions are stored in lists. While the instance positions are stored as vector list, the atom positions are stored as list of *AbstractAtoms*, a class specifically created for this purpose. It does not only store the position but also the colour, the radius and whether or not it is part of an oxide layer. This allows for more efficient access to the atoms data and is especially helpful when transferring the data to the shader. cellVIEWs way of handling data needed in shaders, is to first add it to the *CPUBuffer*, which is a singleton that manages lists containing the data needed for the scene to render. In order to do so the *CPUBuffer* offers a method called *CopyDataToGPU*, which transfers all data from the lists to their corresponding *ComputeBuffer* in the *GPUBuffer*. The *GPUBuffer* is also a singleton and contains only *ComputeBuffers*, which can be used as input for a memory buffer used in shaders. For our purpose we adjusted the *CopyDataToGPU* method, to only transfer the data needed to render the scene to the *GPUBuffer*.

4.2.3 Oxide Layer Collision

The purpose of the oxide layers in this game is that the user, i.e. the electron, can collide with them or tunnel through them. Which of those two events happen depends on the speed of the user on the one hand and on the toughness of the oxide layer on the other hand. This toughness can also be specified by the user as input parameter. The purpose of making it adjustable is to make the game either easier or tougher depending on the target audience.

Whenever the game detects a collision of the electron with any of the oxide layers it first checks if the players speed is higher than the oxide thickness and if so the electron tunnels through the oxide. In that process of tunnelling through the speed is decreased by a tenth of the oxides thickness parameter to simulate the resistance of the oxide. Because of this loss of speed it is possible when the speed gets to low, while being within the oxide, the electron gets repelled despite having enough speed initially. The repelling is just a inversion of the direction in which the electron moves when colliding to simulate a reflection in the opposite direction. Furthermore on every collision with the oxide a sound is played that gives the player an auditory feedback on which one of the two events has happened.

4.2.4 RenderProteins and ColorComposite Shader

These two shaders are not particularly related to each other but since they are both provided by cellVIEW and altered by us, this subsection will briefly address the changes they have undergone in the development of this project. Furthermore since the radius and the colour of the atoms are set by shaders, they can be dynamically adjusted by the user.

RenderProteins Shader The changes to the RenderProteins Shader involved two major parts. Firstly, cellVIEW has an adjustable scale parameter in all its calculations including positions and the radius of the atoms, which is not compatible with any of the Unity *GameObjects* because positions and scales are always off by that scale factor. Since it is of no use for us anyway, we removed the scale parameter completely form all calculations in the shader. Secondly and lastly instead of using triangle billboards as cellVIEW does, we implemented the common quad billboards [Fer09], which significantly improved the proportions of the atoms when being very close to them. The reason for this improvement is a bug in cellVIEW, which was fixed by that adaptation. Figure 4.2 shows the same atom viewed form the identical position with the old cellVIEW shader in (a) and with the quad billboards in (b).

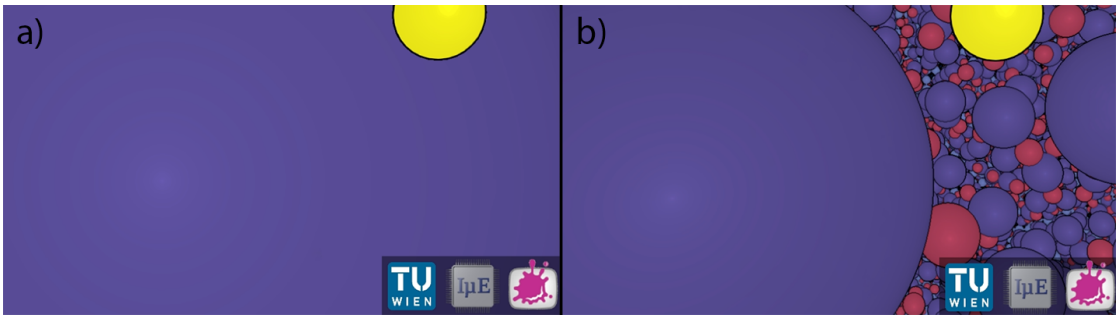

Figure 4.2: An atom viewed from the same direction but rendered with cellVIEWs standard shader in (a) and with our quad billboards shader in (b). While in (a) the atom covers all of the screen, in (b) the atoms proportions are far more accurate.

ColorComposite Shader cellVIEWs standard colour calculation in the ColorComposite shader was of no use for us since it calculated the atoms colours automatically using the technique proposed by Waldin et al. [WLMW+16]. What we implemented was a shader that just sets the colour for each atom which defined by the user for every atom type. This colour is then passed through to the RenderProteins shader where it is used to render the atoms.

4.2.5 Detecting Parameter Changes

Unitys editor offers the possibility to adjust public variables declared in scripts as well as variables which have the *SerializeField* attribute. These variables are therefore displayed in the *Inspector*, as depicted in Figure 4.3, which features the *CrystalStructureManager* respectively its attached *CrystalStructureGenerator* script.

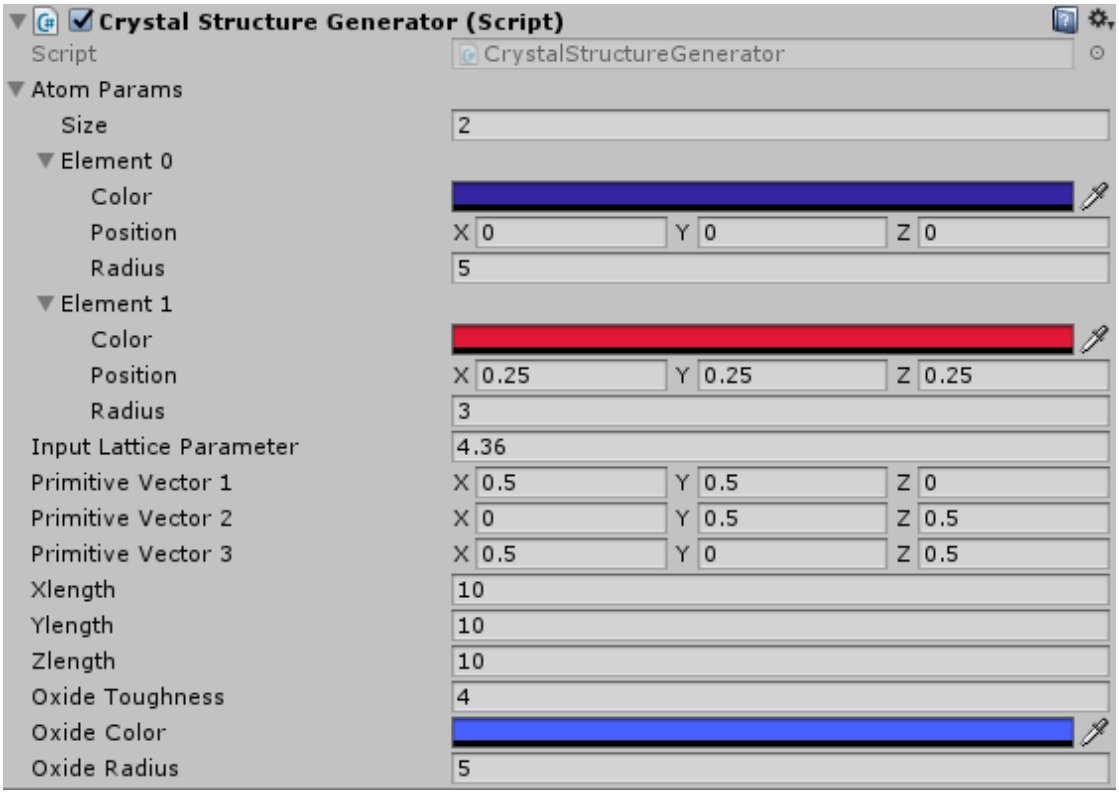

Figure 4.3: A screenshot from the Unity editor, which shows the adjustable variables of the *CrystalStructureGenerator* script. The *AtomParams* size reflects the number of atoms in the crystal structures base. Changes to all colour and radii variables as well as the oxides toughness take effect at run time, while changes to the other variables require a restart.

Changes made in the *Inspector* are directly applied to the variables, even at runtime. However, since the *CrystalStructureGenerators* variables such as the colour and the radius have to be applied to all atoms, changes need to be detected and passed through to the corresponding script, which then updates the atoms accordingly. This is enabled by the *OnValidate* method of the *MonoBehaviour* class that the *CrystalStructureGenerator* inherits. The *OnValidate* method gets called every time a value is changed in the *Inspector*, which fits our needs but does not take care of determining which value has changed. Therefore our implementation of the *OnValidate* method matches every value with its previous value. If a change to the colour or the radius of an atom is detected our method

directly calls the atoms update method. Any change in value that needs the positions of the crystals atoms to be recalculated is ignored because the computation of the atoms positions is too expensive to take place at runtime. It requires a restart of the game and would therefore not take effect immediately.

CHAPTER 5

Results & Discussion

This chapter presents the results of this thesis and determines if the results match the goals defines in the first place. Furthermore the limitations of the developed game and some performance aspects are discussed.

5.1 Presentation of the Results

Having finished all set tasks, our software is capable of generating large-scale crystals of arbitrary inner structure. All parameters are adjustable in the Unity Editor by the user, which enables changes to take place quickly and easily. Moreover the rendering of the structure has been implemented using cellVIEW as base and has been adjusted to our needs throughout the development. Finally, oxide planes have been added to the existing structure, which are not only repelling the electron when its speed is not sufficient but also letting it tunnel through when it has enough speed to outdo the oxides toughness.

Figures 5.1, 5.2 and 5.3 show example screenshots of the Crystal Structure VR software. The crystal is a diamond structure with the violet atoms radius set to 5, as is the oxide atoms radius, and the red atoms radius set to 3. The chosen lattice parameter is 4.36 and the lengths are all set to 10 in x, y and z-direction, which results in about 15 million atoms.

As the screenshots show and the successful implementation of chapter 4 entails are the results exactly what we set out to do. Furthermore, the outcome of this thesis combined with the thesis by Knapp [Kna17] matches the goal defined in Chapter 1.1 entirely and therefore also fulfils the expectation the Institute of Microelectronics had for this project.

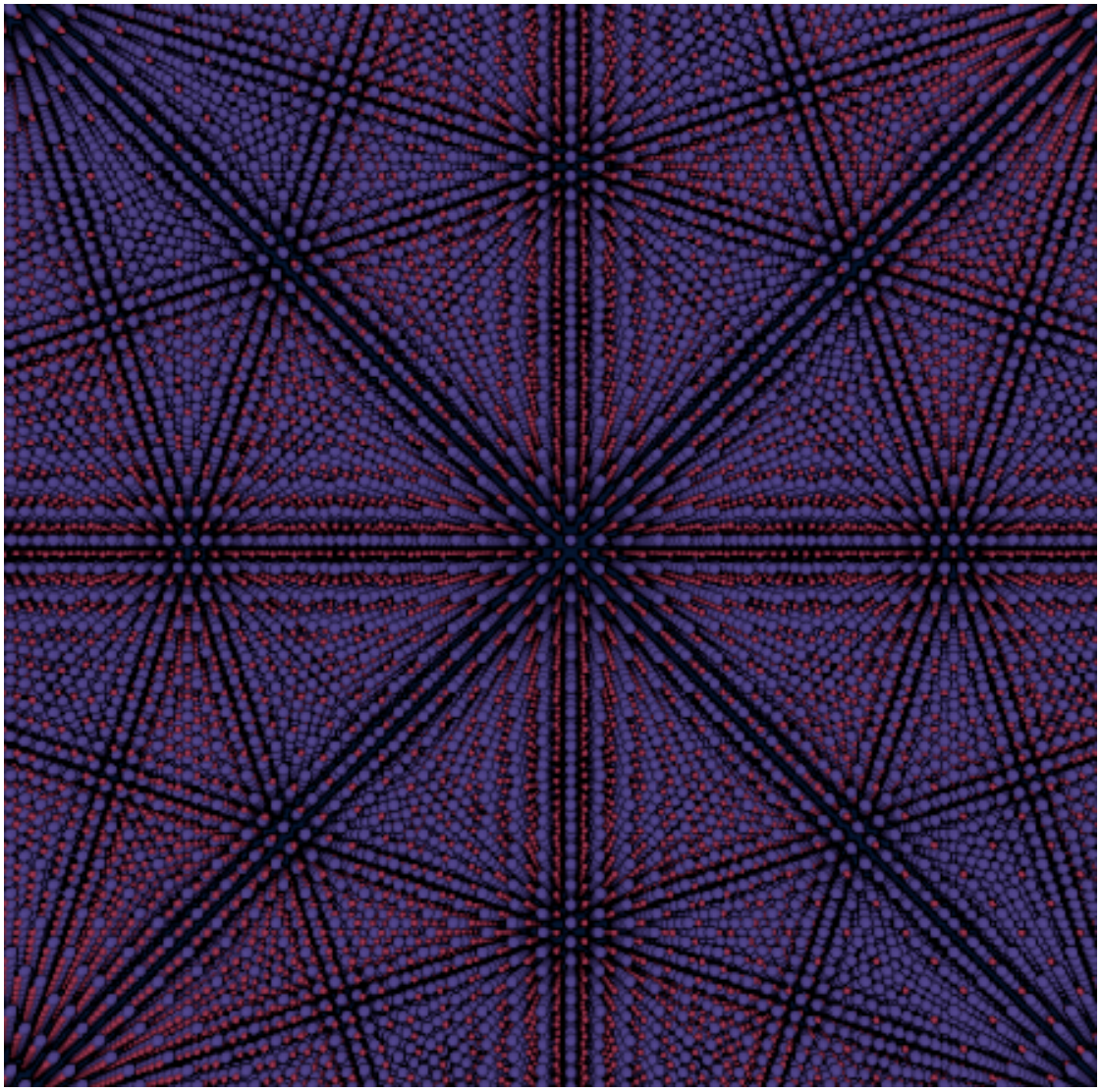

Figure 5.1: An example crystal viewed from distance. The electron has been disabled to be able to see the whole structure.

5.2 Limitations and Performance

The main limitation of this project is the number of atoms that can be generated and rendered. With the structure consisting of about 15 million atoms an average of 80 fps were achieved on our Test-PC, which features a Nvidia GTX 1080 GPU, a Intel Core i7-4790K CPU, 16 GB RAM and a SSD. However, when increasing the atoms to about 26 Million we discovered massive frame drop to about 50fps on average. 50 Million atoms are only running at about 30fps, which is clearly recognizable in-game and significantly reduces the immersion. Furthermore, when trying to generate structure

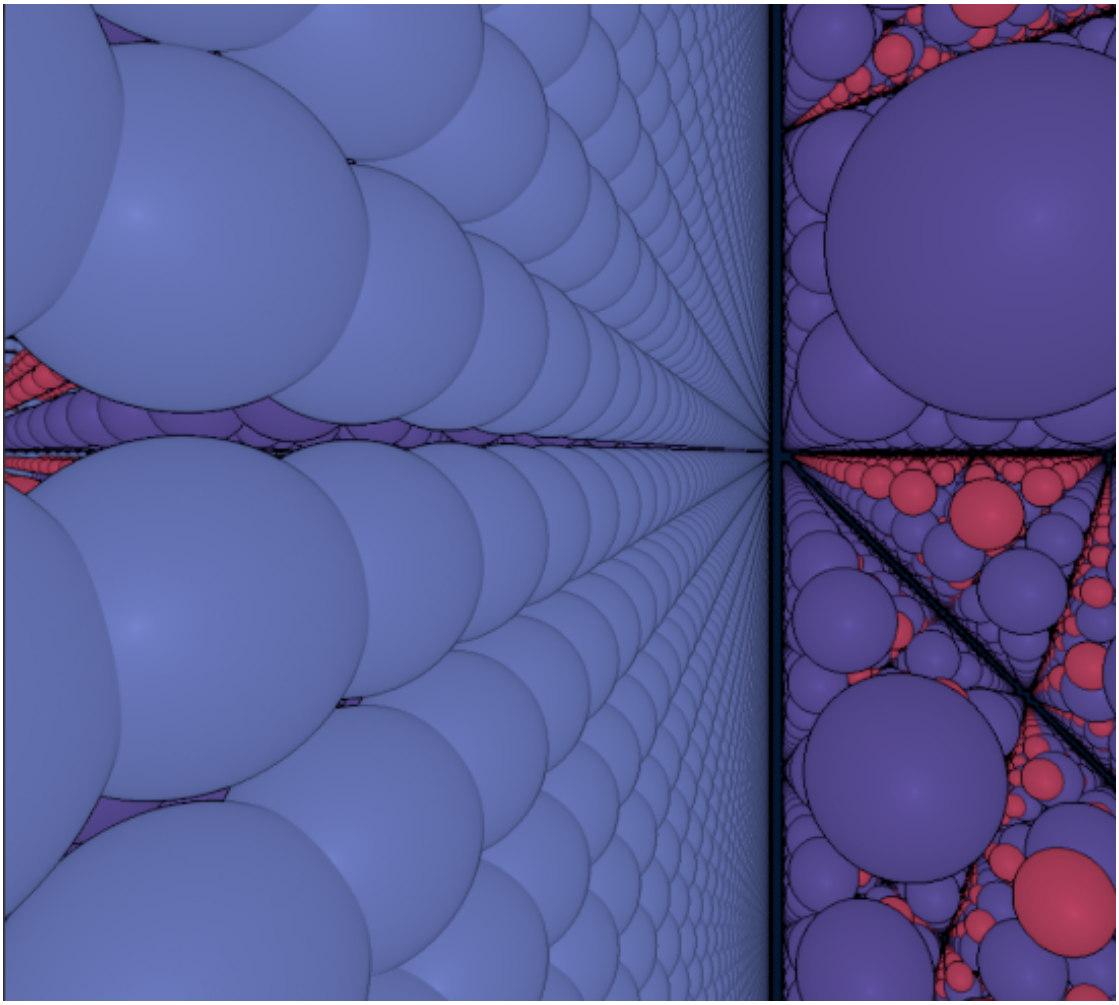

Figure 5.2: The side view of an oxide layer, which shows the extent of the whole structure. The atoms of the oxide layer are packed closely together in contrast to the atoms of the crystal

instances consisting of about 3000 unit cells (30000 atoms) the generation process aborts and unity crashes, which is due to an out of memory error that leads to an "access violation writing location 0xFFFFFFFFFFFFFFD8" exception.

What we found out is essential to prevent errors and increase performance, is to find the right balance between the size of the instances, i.e. the number of atoms or unit cells they contain, and the number of instances. The results show that with those two numbers close to each other, this means if the number of instances matches the number of unit cells and does not exceed the above mentioned limits, the performance of the game should enable a pleasant experience.

Another important limitation is that not all of the parameters the user can control are

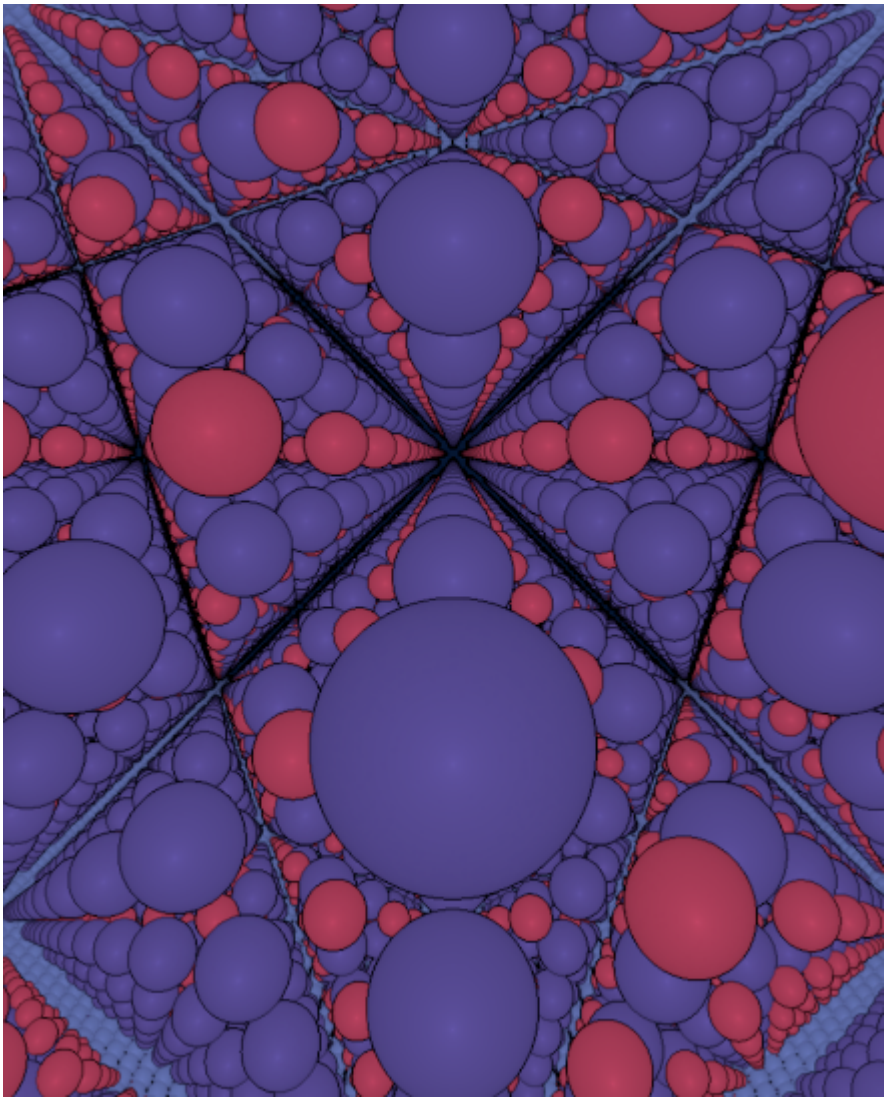

Figure 5.3: A screenshot from within the crystal. The symmetries in the structure are clearly visible.

interactively adjustable. This means that when changing any value of a parameter that has influence on the positions of the crystal or oxide atoms the game has to be restarted for the changes to take effect. The problem is that the change of one position requires us to regenerate all the other positions as well, as they depend on each other and this is far to performance expensive with the game still running. However, as mentioned in Chapter 4.2.4, the parameters that do not require a change in the atoms positions such as the radius and the colour of the atoms, as well as all other parameters as for example the oxide toughness that are position independent.

CHAPTER 6

Conclusion & Future Work

In order to bring this thesis to a conclusion we summarise the work that has been presented. As prospect of the potential of the project, a short listing of the ideas that could be added to the project in the future is provided.

6.1 Summary

Our project has fulfilled all goals set in the workplan that was created in the beginning. The goal was the creation of an interactive 3D scene in which the player controls an electron and flies through a semiconductor crystal in VR. While exploring the crystal several obstacles in the form of oxide layers occur, which can be passed by flying through them with a high enough speed. Otherwise, if the speed is too low the electron gets reflected. However, the whole scene should not be fixed but be adjustable in size and type of crystal structure, as well as colour, radius of the atoms and difficulty.

This work has achieved this goals by creating a game based on cellVIEW and in Unity3D, which is capable of generating and high-performance rendering crystal structures of (nearly) arbitrary size and type, which enables an immersive experience in VR. Furthermore, oxide layers in-between the crystal are generated automatically and add the possibility to tunnel trough them or get pushed away, depending on the players speed and the oxides toughness, which is adjustable. The colour of all atoms as well as the radius of the atoms are customizable, even with the game running. Finally, the limits of the software have been tested, listed in section 5.2 and have no great influence on the gaming experience.

6.2 Future Work

Possible extensions of this project would be for example that all parameters are fully individually adjustable, which means no restart of the game is necessary when changing

any value that has influence on the position of an atom. This would allow the user to interactively generate different structures and look at their differences as well as their similarities. Another idea would be to implement different instance types, which is already supported and used in cellVIEW, enabling the user to fly through different types of crystals in one scene and exploring them as it is now possible with one single type of structure.

Both of the so far mentioned extensions would increase the user experience by adding more interactivity but would also further extend the educational possibilities. In order to also add more gaming elements in this project, a high-score function could be added. This would count the collisions and measure the time the player needs getting through the crystal. Especially on science2public events, this would give people a chance to challenge friends or other attendants.

A way to make people more aware of the context of the scene and the game would be to create an automatic fly-in sequence at the start, which shows a processor from the outside and explains what the user is going to see, while flying into the processor and finally arriving in a transistor and being faced with the crystal structure.

All in all, despite having completed all our goals, there are still many interesting tasks to work on in future projects, which is also something that was intended in the first place and should encourage people who are interested in this topic to take this game to another level.

Bibliography

[Abb17] Duncan Abbot. Arthea – gwydion.co. https://gwydion.co/projects, 2017. (Accessed on 08/15/2017).

[ANe09] A New Glance on Microscopic Images. https://phys.org/news/2009-09-glance-microscopic-images.html, 2009. (Accessed on 09/15/2017).

[AVR17] A VR-Powered Crystal Structures App (Video) | ECE @ Michigan. http://eecs.umich.edu/eecs/about/articles/2017/a-vr-powered-crystal-structures-app.html, 2017. (Accessed on 08/15/2017).

[BP17] Klaus Brandenburg and Holger Putz. Diamond - Crystal and Molecular Structure Visualization. http://www.crystalimpact.com/diamond/, 2017. (Accessed on 08/15/2017).

[Cha13] Amit Chaudhry. NanoScale Effects: Gate oxide leakage currents. In *Fundamentals of Nanoscaled Field Effect Transistors*, pages 25–36. Springer, 2013.

[Cou17] Rachel Courtland. Intel Now Packs 100 Million Transistors in Each Square Millimeter - IEEE Spectrum. http://spectrum.ieee.org/nanoclast/semiconductors/processors/intel-now-packs-100-million-transistors-in-each-square-millimeter, 2017. (Accessed on 08/15/2017).

[DN09] Philippe Decaudin and Fabrice Neyret. Volumetric billboards. In *Computer Graphics Forum*, volume 28, pages 2079–2089. Wiley Online Library, 2009.

[ESA05] M Abdelgawad EL-Sayed and M Atef. Study of tunneling current through ultra-thin gate oxide MOSFET and its effect on CMOS circuits. *Journal of Engineering Sciences, Assiut University*, 33(3):929–941, 2005.

[Fer09] António Ramires Fernandes. Billboarding Tutorial, 2009. (Accessed on 08/15/2017).

[Geo14] Georg-August-Universität Göttingen. LP – Kristallkoordinaten. https://lp.uni-goettingen.de/get/text/6914, 2014. (Accessed on 08/15/2017).

[HH91] John R Hook and Henry Edgar Hall. Solid State Physics (The Manchester Physics Series), 1991.

[Hun09] Siegfried Hunklinger. *Festkörperphysik*. Oldenbourg Verlag, 2009.

[Kit05a] Achim Kittel. Festkörperphysik. Energie- und Halbleiterforschung Fakultät 5, Institut für Physik at Universität Oldenburg, 2005. (Accessed on 08/15/2017).

[Kit05b] Charles Kittel. *Introduction to solid state physics*. Wiley, 2005.

[Kna17] Jakob Knapp. Explorable Semiconductors - Interacting with Crystal Structures in Virtual Reality, 2017.

[LS98] Michael Levinshtein and Grigory Simin. *Transistors: from crystals to integrated circuits*. World Scientific, 1998.

[MAPV15] Mathieu Le Muzic, Ludovic Autin, Julius Parulek, and Ivan Viola. cellVIEW: a Tool for Illustrative and Multi-Scale Rendering of Large Biomolecular Datasets. In Katja Bühler, Lars Linsen, and Nigel W. John, editors, *Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 61–70. EG Digital Library, The Eurographics Association, 2015.

[MI11] Koichi Momma and Fujio Izumi. VESTA 3 for three-dimensional visualization of crystal, volumetric and morphology data. *Journal of Applied Crystallography*, 44(6):1272–1276, 2011.

[MI14] Koichi Momma and Fujio Izumi. VESTA: a Three-Dimensional Visualization System for Electronic and Structural Analysis. http://www.geocities.jp/kmo_mma/crystal/download/VESTA_Manual.pdf, 2014. (Accessed on 08/15/2017).

[Mol16] MoleculE VR - AppMinded. http://www.appmindedapps.com/molecule-vr.html, 2016. (Accessed on 08/15/2017).

[Mom17] Koichi Momma. VESTA. http://jp-minerals.org/vesta/en/, 2017. (Accessed on 08/15/2017).

[PP17] David Palmer and Shirley Palmer. CrystalMaker Software: Crystal & Molecular Structures Modelling and Diffraction. http://www.crystalmaker.com/index.html, 2017. (Accessed on 08/15/2017).

[PRO16] PROteinVR - AppMinded. http://www.appmindedapps.com/proteinvr.html, 2016. (Accessed on 08/15/2017).

[Uni17] Unity - game engine. https://unity3d.com/de, 2017. (Accessed on 08/16/2017).

[VZ04] Bart Van Zeghbroeck. Principles of semiconductor devices. *Colarado University*, 2004.

[Wik09] Wikipedia, the free encyclopedia. N-Kanal-MOSFET (Schema), 2009. (Accessed on 08/15/2017).

[WLMW+16] Nicholas Waldin, Mathieu Le Muzic, Manuela Waldner, Eduard Gröller, David Goodsell, Autin Ludovic, and Ivan Viola. Chameleon. In *Eurographics Workshop on Visual Computing for Biomedicine*, volume 2016. NIH Public Access, 2016.