Software Projects

List and description of software projects I created in context of my studies and in my private time

Subject:	Software Projects
Author:	Maurice Tollmien – maurice.tollmien@gmail.com
Last modification:	February 7, 2019

Contents

1 Introduction

In context of job applications and interviews, it is not always clear, what skills and capabilities the candidate (in this case - me) has and where the main technical competence lies.
During the last years, it became apparent, that a simple list of relevant projects with short descriptions often circumvents and avoids such uncertainties.

With those projects, often linked with the correspondent repositories, I hope to create an impression of my strengths and capabilities.

Most of my more meaningful projects are uploaded or developed using the following GitHub repository:

`https://github.com/MauriceGit/`

All projects I designed, planned and implemented myself are explicitly published as Open Source for unrestricted further use.

The list of exercises and projects is not complete and rather serves to create a valid impression of the technical side of myself. The majority of the listed projects were created in context of my studies (Computer Science - Bachelor and Master). Several others I build in my private time.

2 Conference Talks

I gave a Technical Talk on the Google-Developer-Group Conference (GDG DevFest, October 2016) in Hamburg, Germany on my current research and Master Thesis subject: ***Illumination of density volumes with Neural Networks***.

3 Computer graphics and Virtual Reality

3.1 Basic Computer graphics Projects

2/3D programs in C, using the OpenGL interface and pipeline. Specific projects vary from different Lighting Computations in 3D, Shadow Volumes and Transparency to higher order Spline Interpolation.
Later projects include physically correct approximated calculations for sphere movement on spline surfaces, collision detection and particle simulations to give some examples.
Most later implementations make extended use of the OpenGL Shader language GLSL for Vertex, Fragment and Compute Shaders.

3.2 Procedural Modelling

Creation of a 3D model (from scratch) of a random item with Autodesk Softimage. The resulting object file could then be used by different projects.

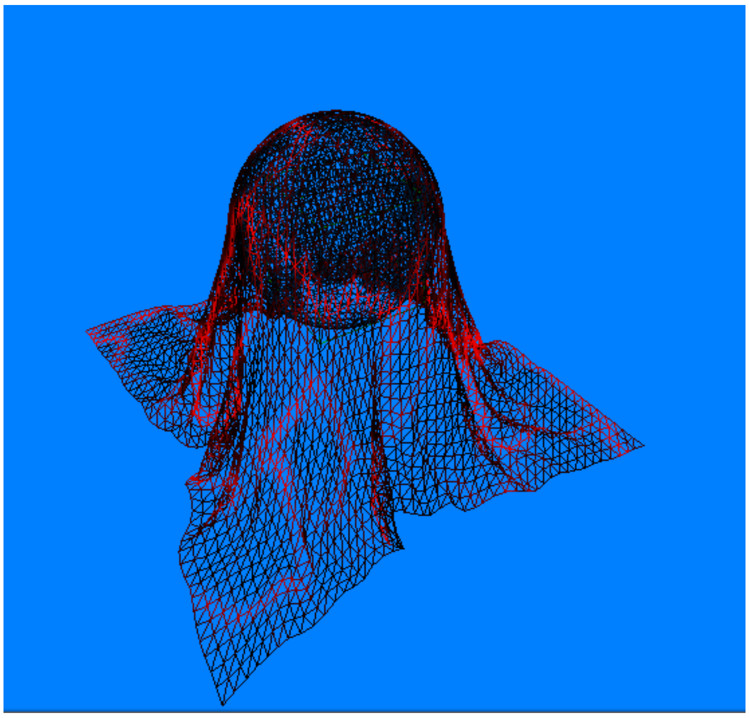
This project (including screenshots) is uploaded on my GitHub account:

`https://github.com/MauriceGit/Procedural_Modeling_Spyder`

3.3 Cloth Simulation

This project was created entirely in my private time. It implements basic cloth simulation using a particle based approach with virtual springs for structure. The cloth interacts with different objects like spheres, quads, planes, as well as virtual wind.
This project (including screenshots and further information) is uploaded on my GitHub account:

https://github.com/MauriceGit/Cloth_Simulation



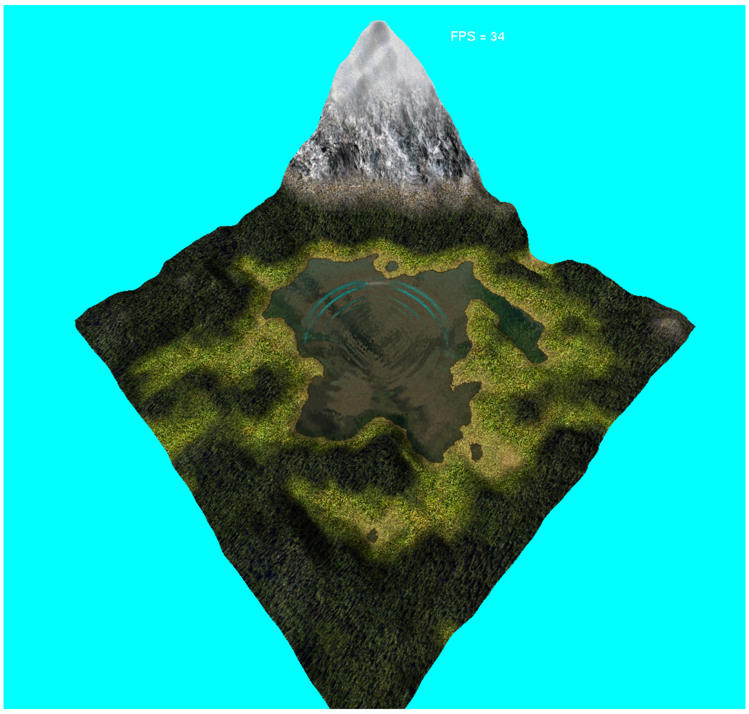
3.4 Water Simulation with Reflections/Refractions

Implementation of a pressure based water simulation, internally represented as a height field. The surroundings are procedurally generated using a self implemented Perlin Noise.

The reflections and refractions are entirely computed in screen space, using the OpenGL fragment Shader.

This project (including screenshots and further information) is uploaded on my GitHub account:

`https://github.com/MauriceGit/Water_Simulation`

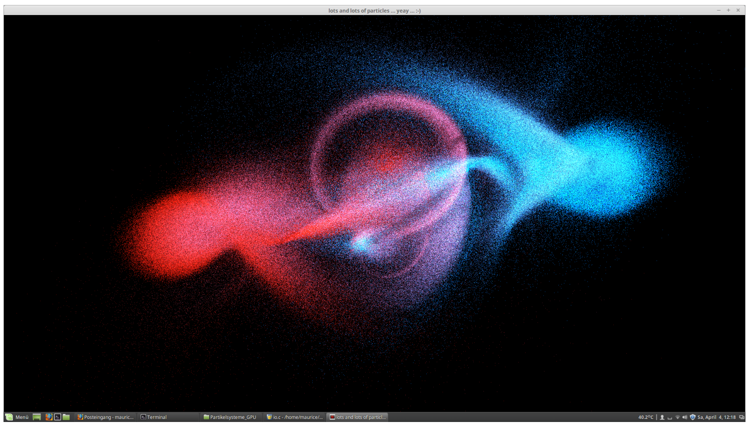


3.5 Particle Simulation on GPU

Implementation of a particle simulation (Euler Integration) by using the OpenGL compute Shader. All computations, regarding the particles, are done by the GPU entirely. Leading to impressive speed and particle count.

Further information and source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Partikel_accelleration_on_GPU`

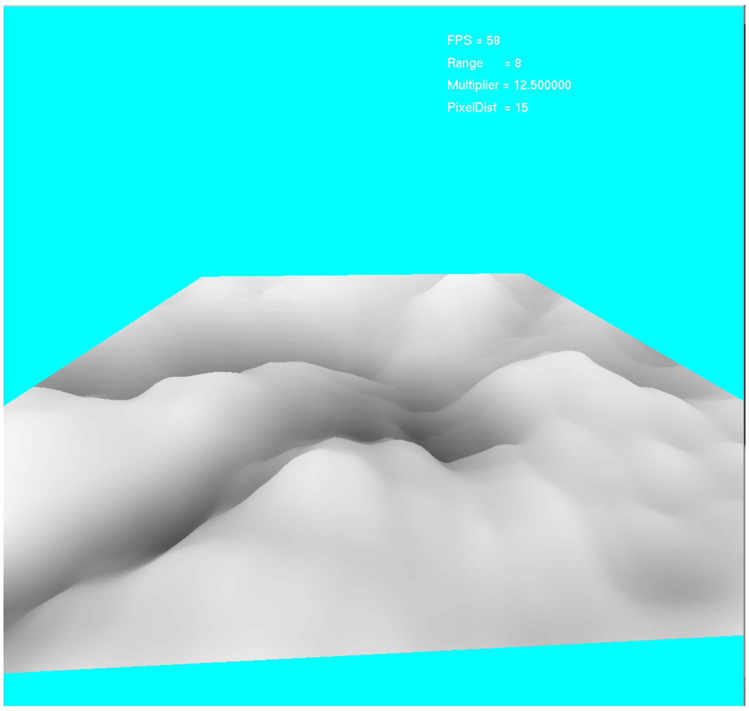


3.6 Screen Space Ambient Occlusion (SSAO)

A private project, implementing a version of Screen Space Ambient Occlusion (SSAO). Adjustments to the Ambient Occlusion are possible while running the sample. The implementation of SSAO is done using the OpenGL fragment Shader.

Further information and source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Screen_Space_Ambient_Occlusion`



```
FPS = 58
Range      = 8
Multiplier = 12.500000
PixelDist  = 15
```

3.7 Low-Level Quaternion Library

I designed a low level C library for common operations on Quaternions. The library is made completely Open Source and stand-alone.
Next to the library, I included a small proof-of-concept program, that rotates an object using Quaternions.

The library can be accessed on my GitHub account:

`https://github.com/MauriceGit/Quaternion_Library`

3.8 Virtual Reality Project

The main goal of this project was to implement software to make a Head Mounted Display (HMD) compatible to Linux. This included reverse engineering of the USB bit stream and interpret it as Quaternions for further use in arbitrary software. The final state of this project was a C library which could be used in any C program using the OpenGL pipeline to create a real 3D experience using the HMD.

3.9 Environment Mapping

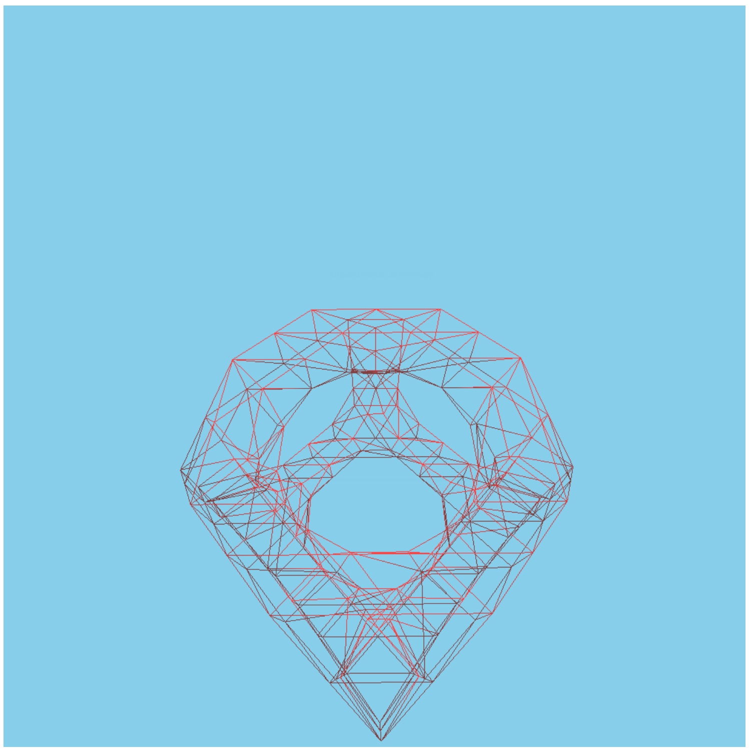
As extension to the implemented water simulation project (see 3.4 „Water Simulation with Reflections/Refractions“), I presented a seminar on how environment mapping could be implemented and realized in virtual environments. Mostly featuring space reflections and parallax corrected cube maps.

3.10 Marching-Cubes on GPU

This side-projects implements the marching-cubes algorithm directly on the GPU to achieve very fast runtimes. Any implicit function can be parsed, triangulated and displayed in 3D with realistic lighting applied.

Further information and source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Marching_Cubes_on_GPU`

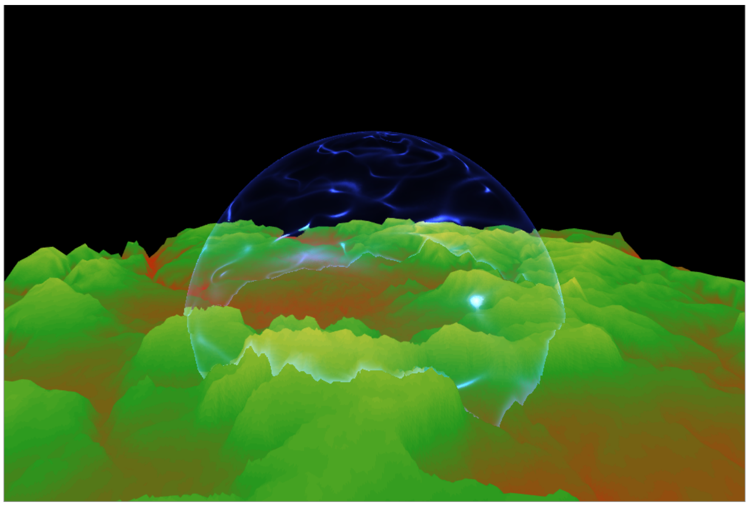


3.11 Heightmap Terrain with Energy Dome

This project was created with the intend to just have fun and create a visually appealing end-result of a real terrain map (of southern Germany) with a procedurally animated energy dome (something you would see as a defense shield in high quality games).

Further information and source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Energy-Dome_Terrain`

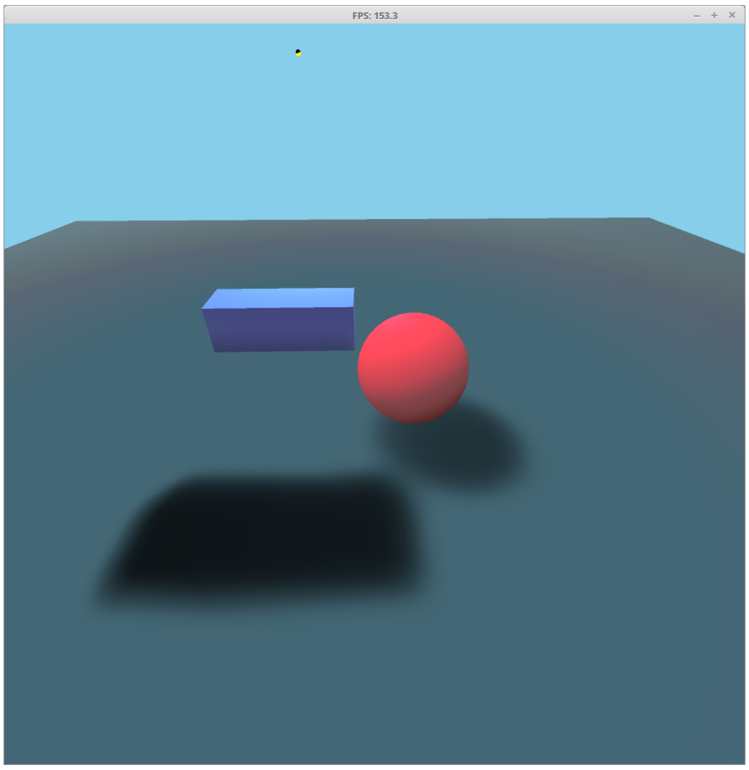


3.12 Variance Shadow Maps

An implementation of Variance Shadow Maps as a fast, hardware filtered (multisampled) variant for soft-shadows in real time.

Further information and source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Variance_Shadow_Maps`



3.13 Delaunay and Voronoi Partition of Images

As a private project, I implemented a software, which takes an image, extracts points according to the local image contrast and triangulates them with a

Delaunay triangulation (implemented completely from scratch). The triangulation is then converted to Voronoi regions. The output are images with the Delaunay/Voronoi regions colored in the mean color of the particular region.

I re-implemented Delaunay and Voronoi using a much more efficient algorithm (see Delaunay and Voronoi – A more efficient and robust approach). This implementation should now be seen as deprecated.

Further information and source code is uploaded on my GitHub account:

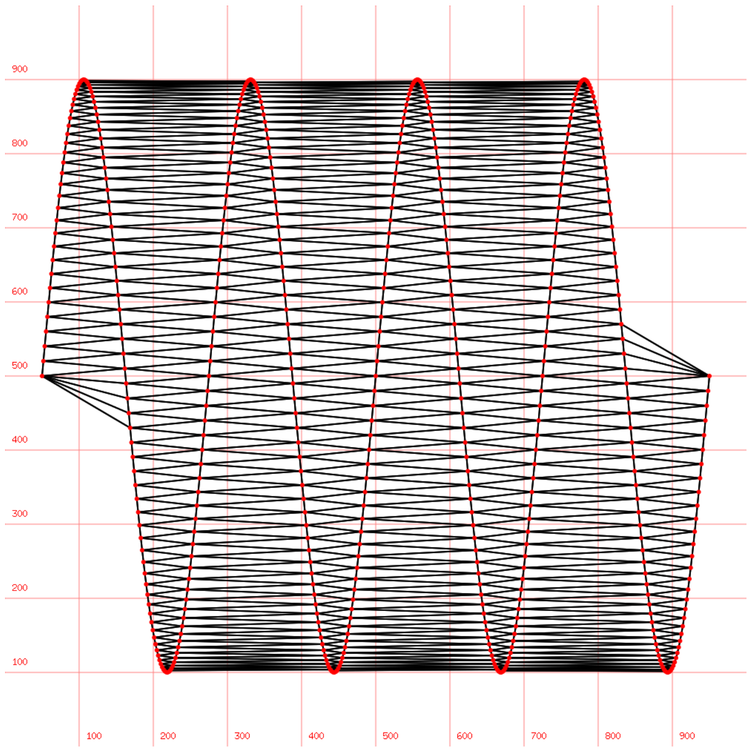
`https://github.com/MauriceGit/Delaunay_Triangulation`

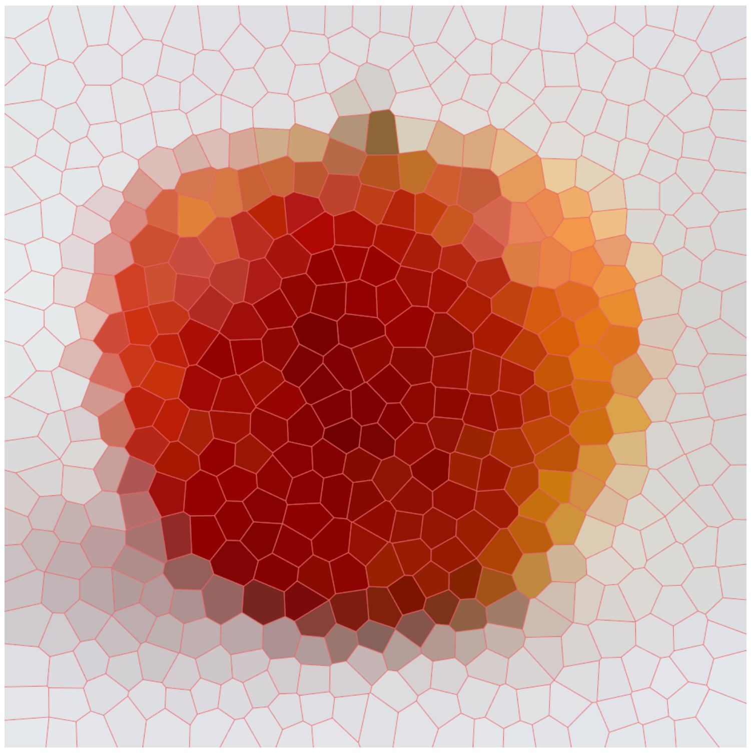
3.14 Delaunay and Voronoi – A more efficient and robust approach

This project aims to create a Delaunay triangulation and Voronoi tessellation that runs in $O(nlog(n))$ at all times and proves to be memory efficient as well as cache friendly. The resulting library is not yet production ready but proves to behave extremely stable independent of input point quality. I currently successfully use this implementation in a couple of other projects.

The library can be accessed on my GitHub account:

`https://github.com/MauriceGit/sweepcircle`





4 Algorithms/Libraries

4.1 Advanced Search in sorted Datastructures

This library contains algorithms for Binary-Search, Interpolation-Search and Quadratic-Binary-Search in generic sorted data structures, such as slices or arrays.

The library can be accessed on my GitHub account:

`https://github.com/MauriceGit/advsearch`

4.2 Skiplist

This library implements a highly efficient Skiplist in Go. In current benchmarks, my implementation is the fastest one available (regarding every single operation). It is highly tested and in a production ready state. Currently I use this library for other projects and in production.

The library can be accessed on my GitHub account:

`https://github.com/MauriceGit/skiplist`

4.3 2,3-Tree

This library implements a 2,3-tree. Other than the normal operations, the next smaller or larger node in the tree can be accessed in $O(1)$ at any time without prior caching or preparation. It was successfully used in production until it was replaced with the Skiplist (see above).

The library can be accessed on my GitHub account:

`https://github.com/MauriceGit/tree23`

5 Artificial Intelligence

5.1 Game AI (Bachelor Thesis)

In the course of my Bachelor thesis, I created a dissertation in a sub discipline of artificial intelligence (AI) with a complex board game (Arimaa[1]) as subject. Next to the theoretical approach, I implemented most of the relevant algorithms such as Alpha-Beta Pruning, Transposition Tables, Zobrist Hashing, Move Ordering and Parallelism.
The dissertation won an award for innovation (Wedeler Hochschulpreis).

The dissertation is published on the official Arimaa homepage:

`http://arimaa.com/arimaa/papers/MauriceTollmien/Thesis.pdf`

The source code of the AI is uploaded and available on my GitHub account:

`https://github.com/MauriceGit/Bachelor-Thesis-Arimaa`

[1]`http://arimaa.com/arimaa/`

5.2 Pure Logical Agent

Design and implementation of a purely logical agent for the game of Wumpus World[2], which dynamically acts according to the game situation. The resulting logical expression is then solved, using a SAT solver (Picosat[3]).
The implementation is functional and written in Haskell. The project is uploaded on my GitHub account:

`https://github.com/MauriceGit/AI-Logical-Agent`

5.3 Artificial Neural Network

This project was created for university and proceeded in my private time. It contains a from-scratch implementation of a backpropagation neural network for letter recognition. Several extensions like gradient descent and dynamic hidden layer count and size are implemented.
In the end it turns out to be roughly on the same level of effectiveness and efficiency as common backpropagation neural networks like PyBrain[4].

The implementation as well as a presentation are uploaded on my GitHub account:

`https://github.com/MauriceGit/Artificial_Neural_Network_Character_Classification`

6 IT Security

6.1 Seminar Paper on Compiler Backdoors

Preparing and presenting a seminar paper on backdoors in software implementation, especially in a compiler itself.
The seminar paper is written in English and published on the website of the university:

`http://www.fh-wedel.de/~gb/seminare/ws2012/tollmien_backdoors.pdf`

6.2 Cryptography Project

Extended workshop on cryptographic methods and algorithms. The workshop covered the practical approach as well as the mathematical background of common cryptographic encryption, hashing and signature algorithms.
Included subjects in this workshop were for example: DES, GOST, AES, message integrity/authenticity, random number generators, number theory, asynchronous techniques like Diffie-Hellman, RSA, elliptic curve cryptography and one way accumulators.

[2] `https://en.wikipedia.org/wiki/Hunt_the_Wumpus`

[3] `https://hackage.haskell.org/package/picosat`

[4] `http://pybrain.org/`

Every task and correspondent solution is uploaded on my GitHub account:

`https://github.com/MauriceGit/Cryptography_Workshop`

6.3 Secure File Encryption

As a private project I implemented an interactive secure file encryption in Python. It allows encryption/decryption, read mode and write mode (using Linux ‚less‘ and ‚nano‘ tools). The encryption consists of hashing the password and AES.

The complete implementation can be viewed under the following link:

`https://github.com/MauriceGit/Secure_File_Encryption`

6.4 Mental Poker

In this project a group of students (myself included) extended the official Bouncy Castle[5] Crypto library with the SRA Algorithm which can be used for mental poker. The main difference to the common RSA algorithm is its commutative behaviour regarding the encryption and decryption.

The SRA implementation is published under the following link:

`https://github.com/timpauls/bc-java`

7 Programming Contests

7.1 Shortest path - 2013

I took part in a programming contest with the goal, to find the shortest path through a set of points. I used a mesh triangulation (Delaunay) from Computer graphics and was able to reach the 3rd rank overall.

7.2 2048 AI - 2014

I participated in a programming contest with the main goal to implement a fast and efficient AI for the game 2048[6]. I reused most of my AI algorithms from my bachelor thesis (see 5.1 „Game AI (Bachelor Thesis)“) and was able to reach the 2nd rank overall.

The complete source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/2048_Game_AI`

[5] `https://www.bouncycastle.org/`

[6] `https://gabrielecirulli.github.io/2048/`

7.3 SameGame AI - 2015

I participated in the programming contest for implementing an efficient AI for the game SameGame[7]. My implementation used a Monte-Carlo tree search. I was able to reach the 3rd rank overall.

The complete source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/Programming_Contest_2015`

7.4 2D Bin Packing

For this year's programming contest, I implemented a paper[8] on optimal 2D bin packing.
I participated in the contest with a multithreaded version of the implemented paper, parallel with a few simpler but faster algorithms. I was able to reach the 2nd rank overall.

The complete source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/2D_Bin_Packing`

7.5 Organisation of the Programming Contest 2016

The programming contest for the next semester is organised by a fellow student and myself. The contest is a about a game similar to Agar.io[9]. All bots will play in real time against each other and will be challenged with different opponents, fields and situations.
All testing is done on a central server. Testers use our Middleware which handles all connections and data transfer to and from the server, making a participation very easy.
The programming contest has already started and is expected to run until the 9th November. At this time there will be an event and the final contest.
All information, including a live view of playing bots can be found under the following link. All information is in German as a courtesy for the participants.

`http://cagine.fh-wedel.de:8080/`

The complete source code can be found here:

`https://github.com/hpatjens/Programmierwettbewerb`

[7] `https://de.wikipedia.org/wiki/SameGame`

[8] „A two-level search algorithm for 2D rectangular packing problem“ by Mao Chen and Wenqi Huang

[9] `http://agar.io/`

8 Other Projects

8.1 XBox controller interface in C

I created a joystick interface for generic USB joysticks in C. Using this library makes it trivial to include and use a USB XBox controller in a C program. The interface methods are adapted to fit perfectly in an OpenGL environment for object or camera control.

The complete source code is uploaded on my GitHub account:

`https://github.com/MauriceGit/XBox_Controller_Linux_Interface`

8.2 Android App

In cooperation with the wildlife park Eekholt[10] we developed an Android App as a navigation help for visitors.

8.3 Lua Compiler

As part of a team of several students, we developed a Lua Interpreter from scratch in Java. My part was mostly the code generation (Generate assembly like code from an abstract syntax tree) but I was involved in decision making of the runtime environment and parser/scanner development as well.

8.4 Distributed Systems in Erlang

Partly for university, partly as a private project, I developed a few algorithms for distributed systems (like bully algorithm, time control, mail delivery, ...) in Erlang.

The implementation of the bully algorithm is uploaded on my GitHub account:

`https://github.com/MauriceGit/Distributed_Systems_Bully_Algorithm`

[10] `http://www.wildpark-eekholt.de/`