UNIVERSITÄT OSNABRÜCK

Faculty of
Cognitive Science

Bachelor Thesis

Romeo: Scripting Environment with interactive Visualizations

Author: Simon Danisch
sdanisch@email.de

Supervisor: Prof. Dr.-Ing. Elke Pulvermüller

Co-Reader: Apl. Prof. Dr. Kai-Christoph Hamborg

Filing Date: 01.02.2014

I Abstract

This bachelor thesis is about writing a simple scripting environment for scientific computing, with focus on visualizations and interaction. Focus on visualization means, that every variable can be inspected and visualized at runtime, ranging from a textual representation to complex 3D scenes. Interaction is achieved by offering simple GUI elements for all parts of the program and the visualizations. All libraries are implemented in Julia and modern OpenGL, to offer high performance, opening the world to scientists who have to work with large datasets. Julia is a novel high-level programming language for scientific computing, promising to match C speed, making it the optimal match for this project. -This section needs more work, and should probably be written in the end

II Table of Contents

III List of Figures

IV List of Tables

V Listing-Verzeichnis

VI List of Abbreviations

1 Introduction

This Bachelor Thesis is about writing a fast and interactive visualization environment for scientific computing. As GUI elements and editable text fields are supplied, one can also write and execute scripts, and immediately visualize all bound variables of the script and edit them via simple GUI elements like sliders. The focus is on creating a modular library, that is written in a fast high-level language, making the library easy to extend. The introduction is structured in the following way. First, an introduction to the general field of research and its challenges is given. From these challenges, the problems relevant to this thesis will be extracted. Finally this chapter will conclude with a solution to the problem, how to measure the success and give an outlook on the structure of the entire Bachelor Thesis.

1.1 Field of Research and Problem

Figure 1: different visualizations of f(x,y,z)=sin(x/15)+sin(y/15)+sin(z/15), visualized with Romeo. From left to right: Isosurface, isovalue=0.76, Isosurface, isovalue=0.37, maximum value projection

Table 1: FE Implementation comparison

implementations	Language	Speed in Seconds
JFinEALE	Julia	9.6
Comsol 4.4 with PARDISO	Java	16
Comsol 4.4 with MUMPS	Java	22
Comsol 4.4 with SPOOLES	Java	37
FinEALE	Matlab	810

The general research field is making computer science more accessible and convenient to use. This is especially relevant for users, that don't have a broad computer science background, or programmers implementing complicated state of the art algorithms. These users can be found especially in scientific computing, which makes this field the focus of this bachelor thesis. More precisely, it focuses on research, which involves writing short scripts, while visualizing the results. An example would be a material researcher,

who is investigating different shapes and materials and their reaction to pressure. The researcher would need to read in the 3D object he wants to analyze, have an easy way to tweak the material parameters and it would be preferable to get instant feedback on how the pressure waves propagate through the object. Several demands by the researcher makes it challenging, to offer software for this area. 1. Money and Time is constrained This means the research has to conclude quickly and most likely, it is not an option to employ a person or even a company to solve sub problems. From this we can deduce three preferences: If code needs to be written, it should be in an easy to understand high-level language. Computation times and feedback should be without delay to speed up development. The used libraries should be accessible to the researcher, because when something doesn't fit his demands, he most likely needs to resolve it himself due to the money constraint. 2. Visualizations Visualizations are very important for two reasons: First, it is difficult to infer any meaningful results from a 10 megabyte file full of 3D points. In this domain, problems become only managable by visualizing them. 1 Secondly, research is getting published, together with visualizations explaining the results. As they present the research to the public, they should be as understandable as possible and preferably look good. To publish good looking, understandable visualizations, tweaking and editing the visualization in a creative, interactive process is of great importance. 3. Speed Speed can be both seen as a usability or time/money constraint. While the money/time constraint is obvious, even slightly slower software can reduces usability by a large amount. Speed is also important for immidiate feedback. As scientists are used to slow computing times of their complicated algorithms, it still holds that it is far more satisfying and productive to immidiately see the results of your work. If the simulation of pressure takes 30s compared to 1s, this is not only frustrating, but has an immidiate influence on how many material combinations the researcher can try out.

Another thing that can happen is, that the researcher notices that the software he is using is missing functionality, or doesn't solve his problem as expected. This can interfere with his research, if it is difficult or impossible to extend the simulation software. As the researchers often do not have a computer science background, all the coding should be done in an high-level language, which allows the researcher to focus on his research without wasting his time with difficult language constructs.

From this example we can deduce different demands which researchers might have for the used software.

1.) Tweaking of parameters via GUI elements 2.) Fast feedback, low latencies 3.) Extensibility 4.) Visualizations 5.) High-Level language

The biggest challenge is to combine these demands. It is not much use, if the visualization software is fast and extensible, if the simulation part is slow and closed source. All the elements must be equally fast, extensible and must work together, so that the researcher does not run into areas, where he gets slowed down in his research. To clarify this point, take for example an algorithm to count the occurrence of a character in a string. Here are different implementations:

The last one actually produces an overflow error, when there are too many occurrences of the character that one searches. The author himself has given up on finding the error, simply because the resulting code was too hard to debug. This nicely illustrates, how fast and optimized code can pose a development barrier, rendering parts of a library untouchable. As this is unacceptable for areas like scientific computing, where the state of the art is advanced on a daily basis, much research is put into generating high performant code from understandable high-level code. On of the most famous research projects in this area is the LLVM project.

1.2 Problem Solutions and Measurements of Success

* Write it in one, open source, high level, high performance language (Julia) * Write it in OpenGL * Open Source * Easy ways of creating GUIs * Modularity * Offer a broad variety of visulizations

1.3 Outlook

1.4 Used Technologies

1.4.1 Julia

1.4.2 OpenGL

1.5 Similar Work

1.5.1 Other Languages

1.5.2 Ipython Notebook

1.5.3 Matlab

1.5.4 Mathematica

1.5.5 Other Graphic acceleration APIs

2 Background

3 Design

3.1 Architecture Overview

3.2 OpenGL Wrapper

3.3 Window System

3.4 Event System

3.5 Visualization System

4 Results and Analysis

4.1 Performance Analysis

4.2 Extendability Analysis

4.3 Usability Analysis

5 Conclusion

5.1 Discussion

5.1.1 Performance

5.1.2 Extensability

5.1.3 Usability

5.2 Future Work

Appendix

A GUI

A nice Appendix.

Screenshot

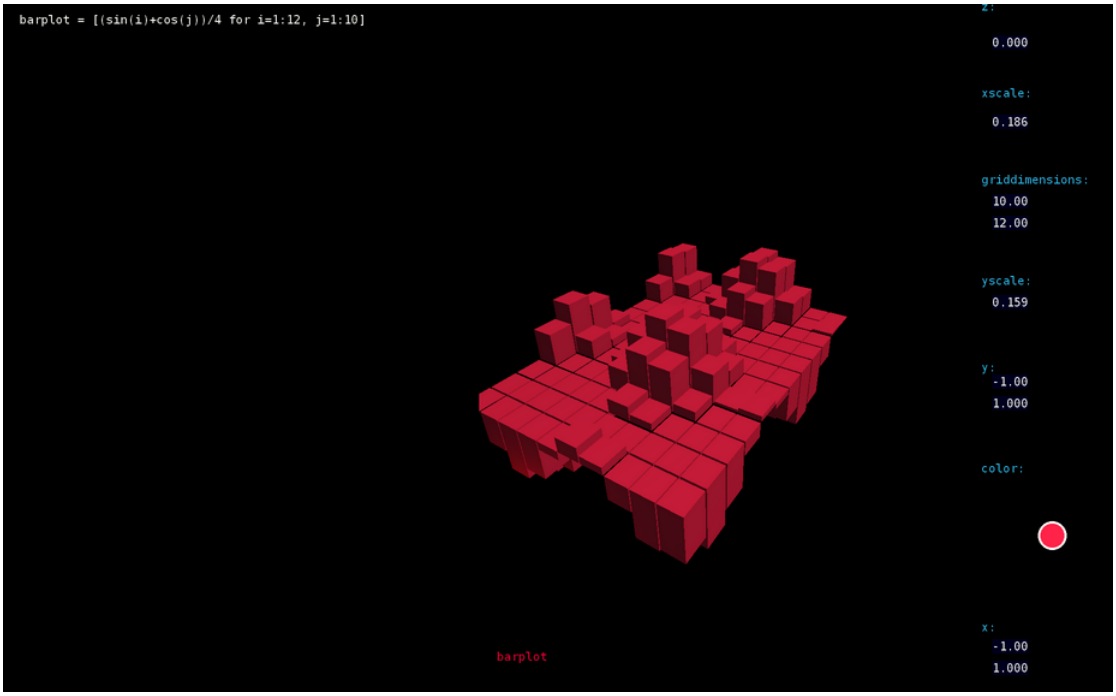

Figure 2: Screenshot of the prototype. Left: evaluated script, middle: visualization of the variable barplot, right: GUI for editing the parameters of the visualization

Official Statement

I hereby guarantee, that I wrote this thesis and didn't use any other sources and utilities than mentioned.

Date: .....

(Signature)