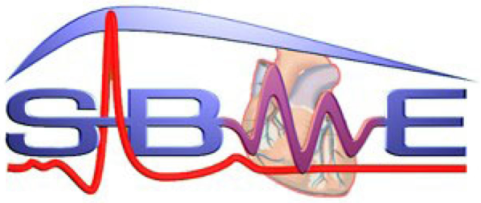
Graduation Project Proposal

Project Title

Project Subtitle, if exists

Author
author@email.com



Systems & Biomedical Engineering Department
Faculty of Engineering
Cairo University

October 5, 2020

Contents

List of Figures

List of Tables

List of Acronyms

CUDA Compute Unified Device Architecture

GPU Graphics Processing Unit

HPC High Performance Computing

IP Internet Protocols

OpenCL Open Computing Language

OpenGL Open Graphics Library

1 Background

Add two to three paragraphs to introduce the problem and its relevant background.

2 Problem Statement & Objectives

Define the problem in detail in two to three paragraphs. It is also required to use graphical illustrations when possible, for example Figure 1 and Figure 2.

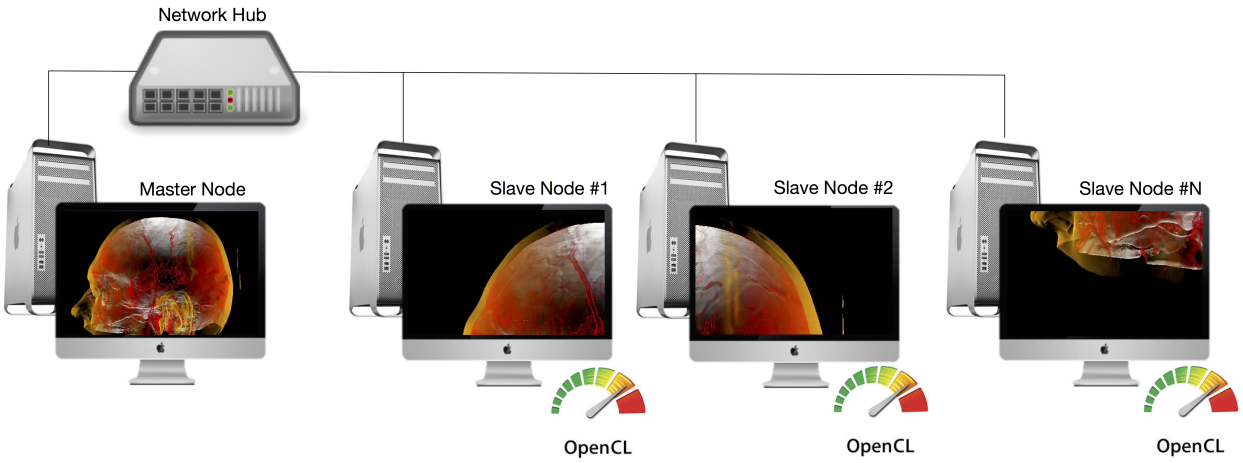


Figure 1: Sample Figure. **Distributed volume rendering**. The rendering load is distributed across a network of slave computing nodes controlled by a master node.

3 Intended Learning Objectives

Define the ILOs of the project and what are the topics, tools, and methods the students will learn by the end of the graduation project. Enumeration is better. An example is given below.

Upon a successful completion of this project, the contributing students are expected to learn and get acquainted with the following

1. Computer graphics concepts, visualization techniques and rendering algorithms.[1]
2. High performance computing concepts.[2]
3. Network programming and distributed computing.[3]
4. Principles of heterogeneous computing.[4]
5. HPC toolkits such as OpenCL[5] and CUDA.[6]

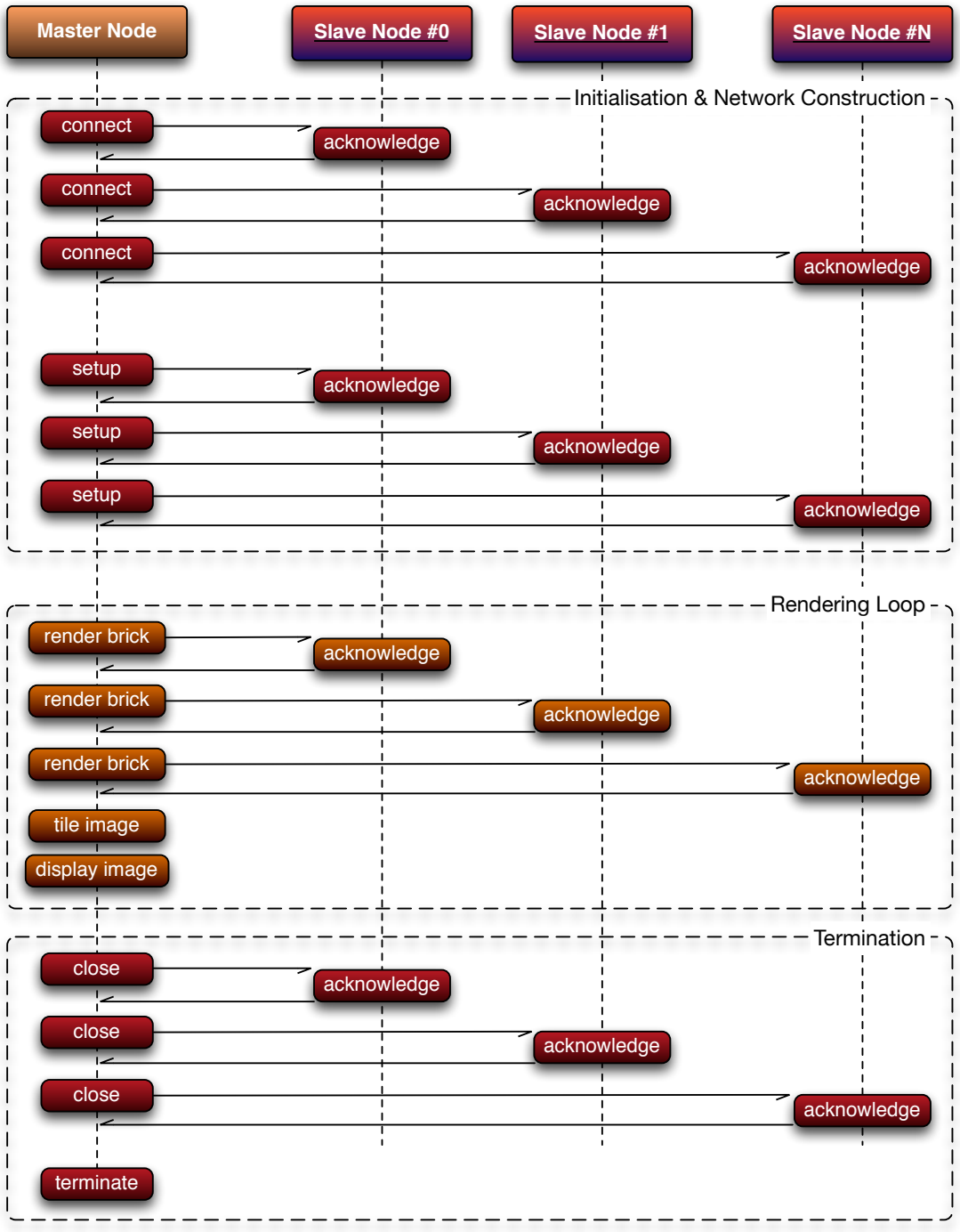


Figure 2: A sequence diagram showing the communication and interaction between the different computing nodes in the system.

6. Optimization techniques.

7. The OpenGL[7] rendering pipeline and the interoperability between rendering and com-

pute contexts on the GPU.

8. Software engineering[8] and principles of code design and refactoring.[9]
9. Code tractability and maintainability via software version control applications.[10]
10. Software build systems.[11]

4 Expected Deliverables

List all expected deliverables of the project including software frameworks, hardware designs, documentations or papers. An example is given below.

4.1 Software

The developed framework is expected to be composed of

1. a server application that configures and controls the client nodes for rendering large volume dataset on a given set of computing nodes with specified IPs interconnected via a local network hub.
2. a configurable user interface designed in Qt to select among the different rendering kernels.
3. a statistical performance analysis library for evaluating the rendering performance of the different platforms.

After the defense, the framework will be released and open-sourced to an open repository on GitHub.[12]

4.2 Documentation

1. Automatically-generated code documentation using Doxygen.[13]
2. Project documentation written in LaTeX.

5 Supervision & Management

Define the logistics of the project including the frequency of the follow uo meetings, code review, or other interactions depending on the project. An example is given below.

5.1 Followup Meetings

There will be a followup meeting with the supervisors on a biweekly basis. Every meeting will be summarized in a short report of maximum two pages.

5.2 Software Management Policy

- The code will be based on the CMake[14] build system.
- The code will be initially developed under Linux (Ubuntu 14.04 or Ubuntu 15.04). The code will be ported later to Mac OSX and Microsoft Windows if the time permits.
- The code will be written in C/C++ & Python.
- The code will be written in OpenCL/OpenGL.
- The code will be hosted on Bitbucket.com[15]
- The individual contributions will not be merged into the master branch of the repository until the code is reviewed and accepted by the supervisor.
- After the successful completion of the project, the code will be tested and uploaded to github.com.[12]

References

[1] Volume Rendering. https://en.wikipedia.org/wiki/Volume_rendering.

[2] What is HPC. http://insidehpc.com/hpc-basic-training/what-is-hpc/.

[3] Client-server computing model. https://en.wikipedia.org/wiki/Client%E2%80%93server_model.

[4] Heterogeneous Computing. https://en.wikipedia.org/wiki/Heterogeneous_computing.

[5] OpenCL. https://en.wikipedia.org/wiki/OpenCL.

[6] CUDA. https://en.wikipedia.org/wiki/CUDA.

[7] OpenGL. https://en.wikipedia.org/wiki/OpenGL.

[8] Software Engineering. https://en.wikipedia.org/wiki/Software_engineering.

[9] Code refactoring. https://en.wikipedia.org/wiki/Code_refactoring.

[10] Version control. https://en.wikipedia.org/wiki/Revision_control.

[11] Software build systems. https://en.wikipedia.org/wiki/Software_build.

[12] GitHub, a powerful collaboration, code review, and code management for open source and private projects. https://github.com/.

[13] Doxygen, a tool for documentation generation from source code. http://www.stack.nl/~dimitri/doxygen/.

[14] CMake, the cross-platform and open-source build system. http://www.cmake.org/.

[15] Atlasian Bitbucket, free for small teams + unlimited private repositories. https://bitbucket.org/.