Likely Kernel Library- An image recognition runtime for heterogenous architectures

Jordan Cheney Jake Weiner

Abstract

The software development of a complex system is often facilitated by a Domain Specific Language (DSL) whose syntax is designed to concisely and efficiently solve problems encountered in that domain. Many communities rely on DSLs; including TeX for typesetting, ggplot2 for plotting, SQL for database queries, GraphViz for graph layout, and Mathematica for symbolic math. The goal of likely is to provide a domain specific language for efficient image processing. Likely holds to several guiding principles which inform design decisions:

1. Immediate visual feedback while developing algorithms
2. Heterogenous hardware architecture support
3. Typeless and efficient kernel syntax for statistical learning
4. Embeddable into other projects and languages

In particular, our specific goals will be to provide basic image processing functions as a proof of concept for likely. These functions will hold to the aforementioned principles of likely, and result in a demonstrable product at the end of the class.

1 Motivation

While there are other image processing DSL's currently available [1][4][5] we feel that a DSL for image processing has yet to be created that allows for immediate visual feedback for algorithm design. This is a feature that we believe is uniquely suited to this field of computer science and will for allow much easier algorithm design in the future. We believe that many people (including the authors themselves) would benefit from seeing feedback of complex image manipulations in real time, and that this would facilitate exciting leaps in the field of computer vision.

2 Background

There currently exist several domain specific languages for image processing, Many of these languages are widely used and come from large and reputable companies like Adobe and Apple [1], and these languages generally offer high level algorithm creation and generate low level code with an emphasis on efficiency. They allow for GPU access for increases in performances and offer multithreading capability.

In addition, DSLs have also been created for specific tasks, like medical image manipulation. For example, the DSL Diderot uses a multithreaded low level code design to create a high level and easy to learn language for medical professionals to perform image manipulations on MRI and CT scans [5].

3 Our Idea

Likely will utilize two well documented libraries, LLVM and OpenGL to implement the heterogeneous architecture design and efficient computation seen in previous image processing DSLs. Likely will move beyond previous work with its implementation of a live coding environment [3]. Using LLVM's intermediate representation language and just-in-time compiler [2], the results of algorithms written in likely's high level language will be immediately visible to the user.

4 Goals

1. Implement very basic math functions; add, subtract, multiply, divide. Achieve good understanding of existing code base
2. Implement more advanced math functions; log, exp, sqrt, etc. Implement casting of matrices to different types.
3. Implement thresholding and saturated arithmetic. Prepare a live demo of the coding environment.

References

[1] Adobe. *Pixel Bender Developer's Guide.*

[2] Edward Barrett. 3c - a JIT compiler with LLVM. Technical report, Bournemouth University, 2009.

[3] Joshua C. Klontz. Likely kernel library.

[4] Brian Guenter Diego Nehab. Neon: A domain-specific programming language for image processing. Technical report, Microsoft Research, 2010.

[5] Charisee Chiw Gordon Kindlmann John Reppy Lamont Samuels Nick Seltzer. Diderot: A parallel dsl for image analysis and visualization. Technical report, University of Chicago, 2012.