CV-HIP
Hardware Accelerated Image Processing Library

Amir Abdrakmanov, Angelo Gonzales, Dave Sharma, Evyn Rissling, Mohamed Yassin
Schulich School of Engineering, University of Calgary

Abstract

The field of computer vision is rapidly changing and evolving. Within this rapidly evolving context improvements can be made to existing algorithms and libraries. One such improvement is hardware acceleration, particularly using the graphics card of a computer to speed up algorithm processing time.

The graphics card manufacturer NVIDIA has realized this and created a programming library called CV-CUDA that allows users to process images much faster. Unfortunately, CV-CUDA only runs on NVIDIA hardware. Our project, sponsored by the semiconductor company Advanced Micro Devices (AMD), solves the problem of hardware acceleration for AMD through the creation of an AMD compatible library which performs a similar function to that of CV-CUDA.

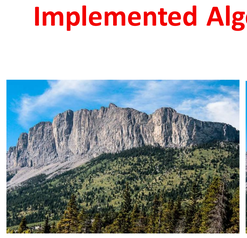
This project has four main deliverables:
- A set of 17 must-have image processing algorithms with and without graphics card acceleration
- The ability to use this library in three different programming languages
- The ability to perform unit testing of each algorithm
- A set of one or more sample applications that use the library to do meaningful work, like preprocessing images for machine learning.

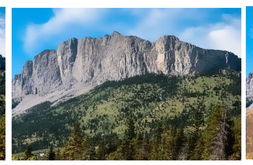
Our project makes use of several other pre-existing AMD libraries and the AMD Heterogeneous Interface for Portability (HIP) programming language to achieve these deliverables.

Results

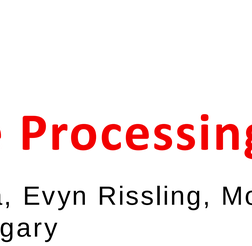
Over the course of this project:
- An API framework was created with C++, C and Python interfaces.
- All 17 must-have algorithms were implemented with the above interfaces
- An extra 4 algorithms were added on top of that, 3 of them have all the interfaces
- Correctness tests for each algorithm were implemented in each language
- Performance tests were added to compare against CV-CUDA's performance on similar hardware
- One simple C++ sample app for viewing the outputs of each operator
- Two complex C++ sample apps for doing useful work with the SIFT algorithm
- A simple C sample app to verify C compatibility
- A moderate Python app to preprocess images for useful work

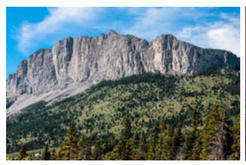
Implemented Algorithms


Original Image

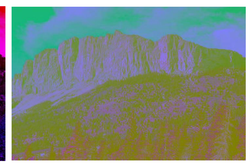

Bilateral Filter

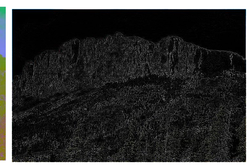

Bounding Box

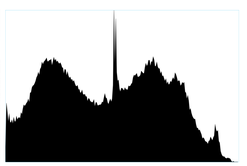

Advanced Color Conversion

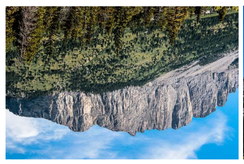

Box Blur


Channel Reorder

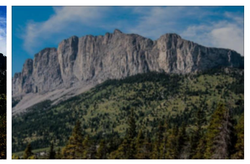

Color Conversion

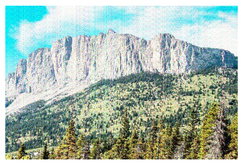

2D Convolution

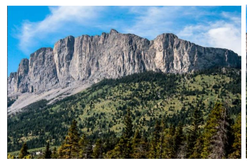

Histogram

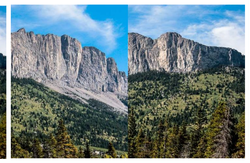

Flip

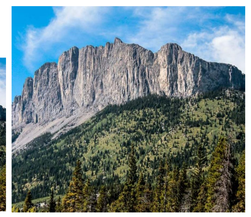

Gamma Correction

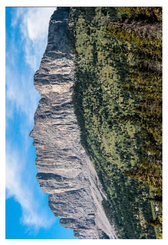

Gaussian Filter


Gaussian Noise


Median Blur


Remap


Resize


Rotate


Threshold


Warp Affine


Warp Perspective


Custom Crop


Scale Invariant Feature Transform