Musemage
The Revolution of Image Processing

Kaiyong Zhao – Hong Kong Baptist University, Paraken Technology Co. Ltd.
Yubo Zhang – University of California Davis
Outline

• Introduction of Musemage
• Why GPU based
• Musemage Features
• Musemage Framework
• Background & Future Works
What is Musemage?

• The first comprehensive image processing software fundamentally based on GPU.

• Ultra-fast image processing experience

• Rich and easy-to-use functions

• Friendly user interfaces
Outline

- Introduction of Musemage
- Why GPU based
- Musemage Features
- Musemage Framework
- Background & Future Works
Why Musemage uses GPU?

CPU + GPU co-processing
Why Musemage uses GPU?

- GPU has massive compute power
  - SP peak performance of Fermi GPU more than 1T flops/s
  - GPU compute power is growing faster
  - GPU has much higher memory bandwidth (10X CPU)

- The architecture of GPU is designed for 3D graphics processing
  - Image processing in a sense is similar to 3D graphics
  - High precision (fp32) compute power is free on GPU
  - Programmability of GPU dramatically improved in past 3 years, complicated shaders enable advanced filters
What are the difficulties?

• Memory management is more complicated
  – Need to manage 3 levels of memory (GPU mem, host mem, HDD buffer)
  – Hide the data transmission delay
  – Efficient undo/redo

• Many image processing algorithms are rarely implemented on GPU

• GPU memory is limited
  – Image processing needs large memory
  – Memory size of main stream GPU is limited (256MB-1GB)

• Issues raised by GPU’s graphics characters
  – Driver timeout
What GPU brings for Musemage?

• Because all adjustment tools and filters of Musemage are implemented on GPU, it can provide:
  – High processing speed
    • It takes less than one second when filtering 20M pixels image on a commodity GPU
    • Some filters are 15-100x faster than existing image editing softwares
  – Real-time full screen feedback when user dragging the sliders to adjust parameters.
  – Zooming and moving canvas by mouse wheel and right button at any time, even while adjusting filter parameters.
Outline

• Introduction of Musemage
• Why GPU based
• **Musemage Features**
• Musemage Framework
• Background & Future Works
What Musemage Offers?

- Rich painting tools
- Complete color adjustment tools
- Various filters
  - Color, blur/sharpen, noising, distortions, stylize...
- Beautify tools
- Lens corrections and lens effects
Powerful and Easy-to-Use Beautify Tools
Powerful and Easy-to-Use Beautify Tools
Real Time User Experience

- Real time feedback when dragging slider
- Zooming and moving picture are also available while editing
- What You See Is What You Get
Lens Corrections and Lens Effects
Lens Corrections and Lens Effects
Fast GPU Extraction
Simple User Interface

• All filters and tools can be easily found in the side panel
  – Filters listed as thumbnail pictures to provide effect estimation
• All pictures of current folder are listed as thumbnails on Picture Wall
  – Fast access a collection of pictures
  – More picture management functions in the feature
• Menu and hot-key accesses are also provided
Outline

• Introduction of Musemage
• Why GPU based
• Musemage Features
• Musemage Framework
• Background & Future Works
Musemage Framework

- Musemage
- Qt
- OpenGL
- OpenCL
- CUDA
- GPU
- CPU

Musemage Plugins
Musemage Data Management

Video RAM

Host RAM

Hard Disk
Musemage Work Flow

Image Files → Musemage Image → Operators Filters → User Parameters

Fragment Shaders
CUDA/OCL Kernels
Example - Radial/Surface Blur

• Use GLSL fragment shaders for color sampling
  – Input color texture and selection mask texture
  – For each pixel, sample surrounding pixels and take weighted average
  – Multi-level acceleration
Radial Blur
Performance Test

Platform:
- Intel Core i7 870, 2.93GHz
- NVIDIA GeForce GTX480

21Mp 5dmark2 picture
Example - Extraction Tool

- Provide binary segmentation and alpha matting
- Binary segmentation with edge optimization
  - For nearly opaque object selection
  - For semi-transparent object extraction (as initial trimap)
- Alpha matting (linear model)
  - Segmentation based
  - Direct matting (require user painted trimap)
GPU Matting

• Reformulated alpha matting algorithms
  – Color sampling
  – Linear alpha blending model
• Multiple GLSL fragment shaders
  – Interactive user input
  – Trimap generation
  – Alpha estimation
  – Iterative optimization
Binary Segmentation
Binary Segmentation
Iterative Edge Optimization
Segmentation-based Alpha Matting
Outline

• Introduction of Musemage
• Why GPU based
• Musemage Features
• Musemage Framework
• Background & Future Works
Background of Musemage

• The project started on Oct. 2009
• Standard GLSL 1.2 program
• Cross-platform — Qt, OpenGL (Windows, Linux, Mac OS)
• Multi-language interface — English, Chinese, and more...
• All codes are written by Musemage develop team, except RAW data decoding (DCRAW is used)
• Scalable software architecture
• Sep. 1\textsuperscript{st} 2010 to release Musemage v1.0
Future Works

• Improve selection and layer management
• Add channel and mask support
• More automatic functions (alpha matting, HDR...)
• Support very large images (>64MP images)
• Higher precision image formats and processing
• Improve image management and exploration
• New RAW engine
• Mac OS Version
Roadmap

• V1.5
  – GPU alpha matting and other advanced features
  – Quality enhancement and performance optimization
  – More languages

• V2.0
  – Big images (not limited by maximum texture size)
  – Enhanced layer operations and selection tools
  – CUDA/OpenCL support
  – GPU-based RAW engine and 16bit channel format
Questions?