

SIGGRAPHASIA2010
SEOUL

MobiRT: An Implementation of OpenGL ES-based CPU-GPU Hybrid Ray Tracer for Mobile Devices

Jae-Ho Nah, Yoon-Sig Kang,
Kwang-Jo Lee, Shin-Jun Lee,
Tack-Don Han, Sung-Bong Yang

Yonsei University, Korea

Contents

- Motivation and goals
- Problems and solutions
  - Performance
  - Secondary rays
  - Texture mapping
- Experimental results
- Conclusions and future work

Contents

- Motivation and goals
- Problems and solutions
  - Performance
  - Secondary rays
  - Texture mapping
- Experimental results
- Conclusions and future work

Motivation

- 3D user interfaces (UI)
  - A key application of visualization on mobile devices
- The difficulties of 3D UI design
  - Complex Shader programming
  - Low rendering performance of mobile GPUs

Motivation

- Ray tracing [Whitted 1980]
  - A technique for generating an image by tracing the paths of lights
  - Widely used for off-line rendering

- Ray tracing can be a solution for 3D UI
  - naturally supports global illumination effects
    →generates high-quality images & simplifies Shader programming
  - Performance is inversely proportional to the pixel size.
  - supports flexible primitive types.

Goals

- Implement an OpenGL ES-based CPU-GPU hybrid ray tracer
- Support full Whitted ray tracing
  (reflections, refractions, hard shadows)
- Support dynamic scenes

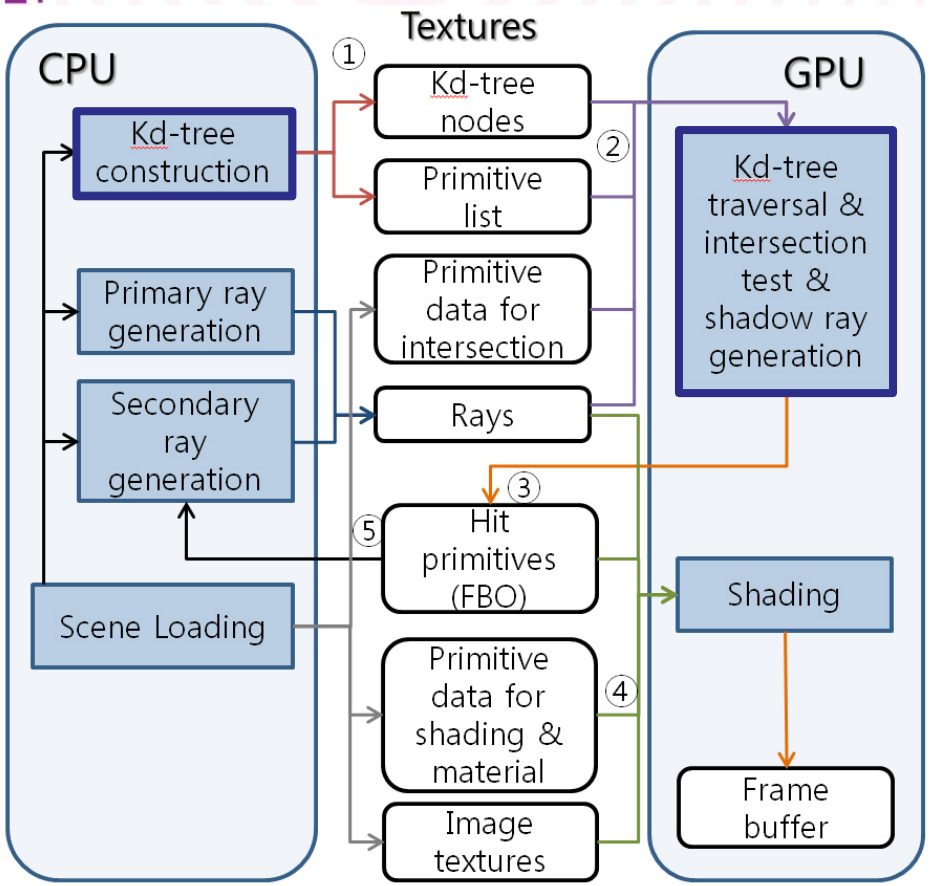
Contents

- Motivation and goals
- Problems and solutions
  - Performance
  - Secondary rays
  - Texture mapping
- Experimental results
- Conclusions and future work

Problems to Solve

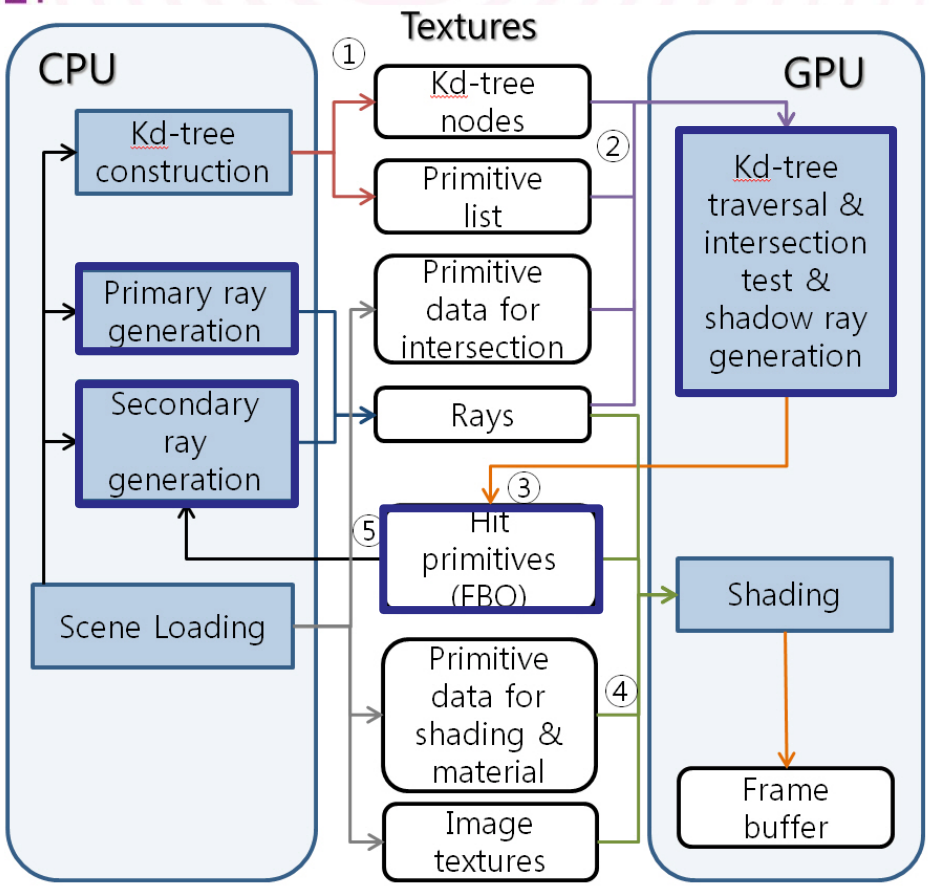
- Performance
  - Mobile GPUs have much poorer performance than desktop GPUs
- Secondary rays
  - OpenGL-ES 2.0 doesn't support multiple render targets (MRTs) → only in the extension specification
  - Management of the ray tree is limited on the GPU.
- Texture mapping
  - Ray tracing requires access of the entire scene data.
  - # of textures in the entire scene
    - > # of multi-texture units in the GPU

Solution for Performance



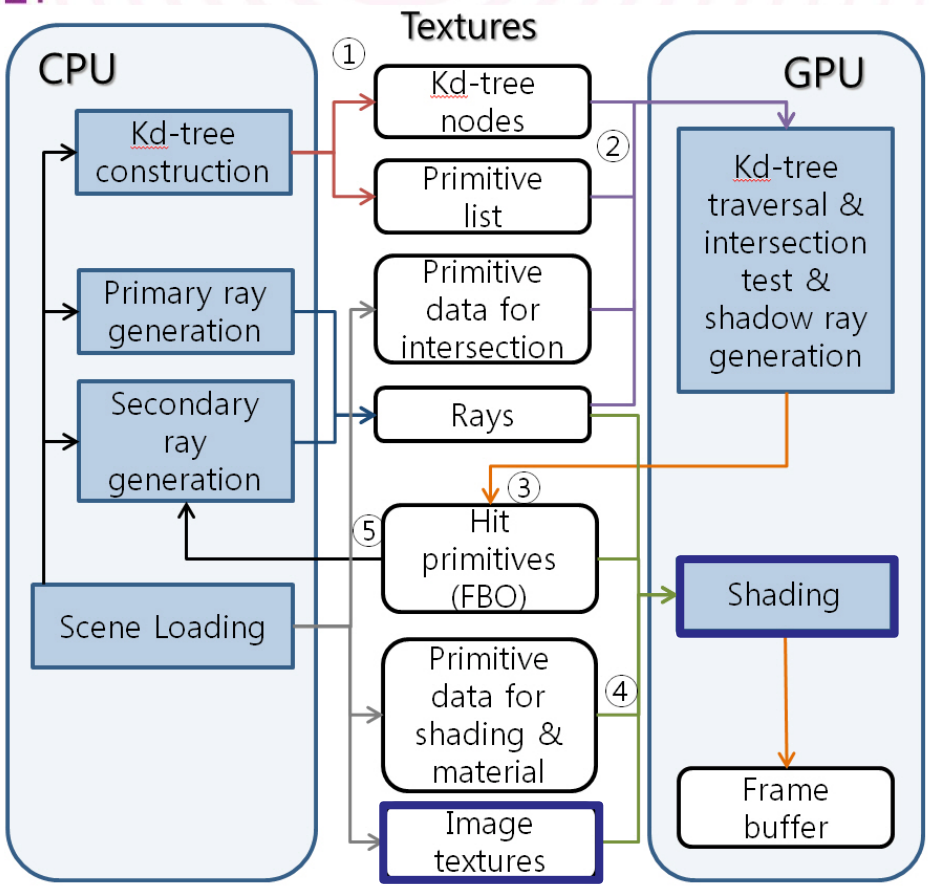
- Exploit the availability of CPU and GPU architectures
- Kd-tree build on CPU
  - Binned SAH approximation [Shevstov et al. 2007]
- Ray traversal on GPU
  - Short-stack algorithm [Horn et al. 2007]

Solution for Secondary Rays



- 32-bit compact output format
- Ray traversal kernel
  - 24bits : primitive index
  - 8bits : shadow results
- Shading kernel : 32bit RGBA
- CPU manages
  - Ray tree for secondary rays
  - Hit points, normals, texture coordinates for shading

Solution for Texture Mapping



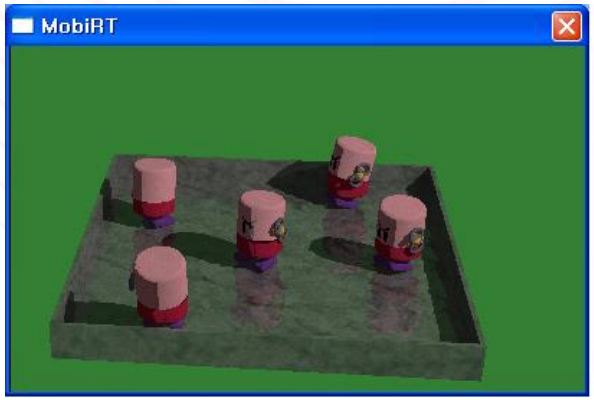
- Apply texture atlases [NVIDIA 2004]
- 16 textures (<=512x512 size) → 1 global texture (2,048x2,048 size)
- Support variable size textures

Contents

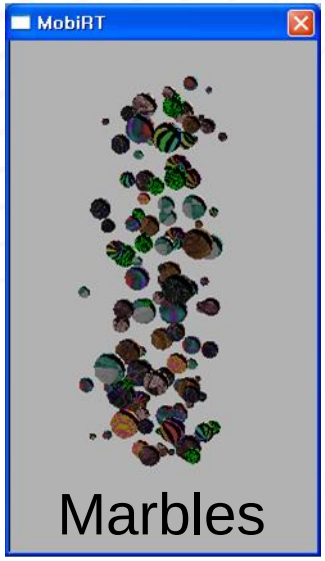
- Motivation and goals
- Problems and solutions
  - Performance
  - Secondary rays
  - Texture mapping
- Experimental results
- Conclusions and future work

Test Setup

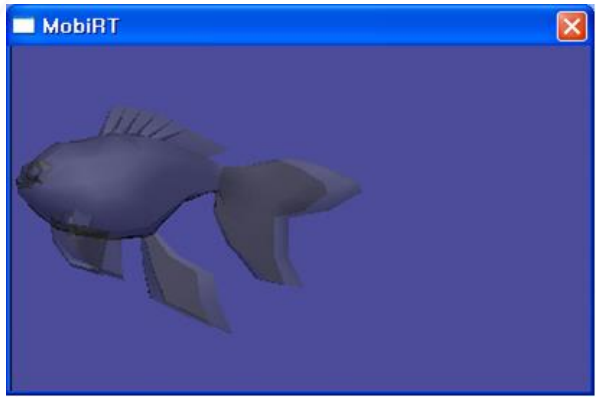
- AMD OpenGL-ES emulator 1.4
- 2.9GHz AMD Athlon-X2, 2GB RAM, NVIDIA Geforce 9800GT
- Benchmark scenes



Toaster (11K tris.)



Marbles (8K tris.)



Fish (1.4K tris.)

Video

- AMD OpenGL-ES emulator 1.4
- 2.9GHz AMD Athlon-X2, 2GB RAM, NVIDIA Geforce 9800GT
- Benchmark scenes

Results

Table 1: *Benchmark results on the OpenGL ES emulator (frames per second)*

Scene (resolution)	Toaster (400x240)	Marbles (240x400)	Fish (400x240)
ray casting	24	26	48
+shadow	20	N/A	N/A
+1-bounce reflection/refraction	15	N/A	26
+2-bounce reflection/refraction	12	N/A	20

- We expect that the MobiRT will show 1-5 FPS on real mobile devices.

Contents

- Motivation and goals
- Problems and solutions
  - Performance
  - Secondary rays
  - Texture mapping
- Experimental results
- Conclusions and future work

Conclusions and Future Work

- The implementation of an OpenGL ES-based CPU-GPU hybrid ray tracer
  - CPU : kd-tree build and management of the ray tree
  - GPU : kd-tree traversal, intersection tests, and shading
  - Supports full Whitted ray tracing of dynamic scenes.

- Future work
  - Implementation on real mobile devices
  - Using OpenCL for faster and more efficient ray tracing

Acknowledgements

- This work was supported by Samsung Electronics Co., Ltd.

YONSEI UNIVERSITY

Q&A