Real-Time Lens Blur Effects and Focus Control

Sungkil Lee

Sungkyunkwan University
Big Picture in 3D Computer Graphics

Modeling
Rendering
Animation
Rendering?

- Simulation of light physics
- Goals: photorealism, better perception
Today’s Topic

Lens Blur Rendering
Lens Blur?

“Adding a sense of focus and depth to a 3D scene”
Lens Blur?

Depth of Field
• Sharp focus
• Blurry foreground
• Blurry background
Depth of Field
Benefits

- Photorealism
- Monocular depth cue
- Attract attention/guide observer
Creative Possibilities

- Artistic effects
Creative Possibilities: Lens Blur Effects

- Artistic effects
Technical Contributions

General lens
- Arbitrary shape, physical refraction

High performance/quality
- Real-time and artifact-free
Technical Contributions

Focus control
- Intuitive user interface, arbitrary focal surface
Technical Contributions

Focus control

- Intuitive user interface, arbitrary focal surface
Previous Work: Standard Methods:
Distributed Ray Tracing [Cook, 1984]
Previous Work: Standard Methods:
Accumulation Buffer [Haeberli, 1990]
Previous Work:
Fast Approximation?

Filtering!
of a single-view image using a depth map
Previous Work:

Fast Approximation
Previous Work:

Fast Approximation

Single-view approximations cannot deliver correct visibility

Typical foreground problem

Reference
Use multiview images!
Yet, without actual rendering
Previous Work:

Multiview Synthesis [Lee et al., 2009]
Previous Work:

Multiview Synthesis [Lee et al., 2009]

Limitations

- High memory consumption
- Layers incomplete might miss geometry

Our new method

- consumes less memory
- handles hidden surfaces correctly
- more efficient generalized ray traversal
Rendering Algorithm
Rendering Algorithm

1) Lens-ray generation
2) Scene representation
3) Ray tracing
1) Lens-Ray Generation
1) Lens-Ray Generation: *Thin-Lens* Model

**Analytical equation** [Potmesil et al., 1981]
1) Lens-Ray Generation: Our **Geometric-Lens** Model

- Double refraction (ray traced)
- Characterized by **lens shape/refraction index**
1) Lens-Ray Generation: Our **Geometric-Lens** Model

**Spherical aberration**
1) Lens-Ray Generation: Our Geometric-Lens Model
1) Lens-Ray Generation: Our **Geometric-Lens** Model

**Curvature of field:** curved image (or focal) plane
1) Lens-Ray Generation: Our Geometric-Lens Model
Chromatic aberration: wavelength-dependent refraction
2) Scene Representation
2) Scene Representation

- **Previous work** [Lee, 2009]:
  - efficient ray tracing, but only for **thin lens**
  - exploits **uniform image-distance** layer spacing
  - artifacts for **missing hidden surfaces**
2) Scene Representation

Our accurate solution:

- **Depth peeling** [Everitt, 2001]: per-pixel depth layers
- **Do not miss** any hidden surfaces
2) Scene Representation

- **Problem**: Many layers necessary
- **Reducing** layers is crucial!
- Our solution: *culling*
Umbra Culling

Image-based = pixel is geometry
- Find umbra

"Umbra"

Next depth-peeling threshold
Umbra Culling

2 good properties
- fast marching: next threshold > current threshold
2 good properties

- **number** of layers is bounded (**pixel size > lens size**)
Umbra Culling

Number of layers:
- 30 layers always enough (independent of scene)

for FOV=30, near=1m, far=∞, lens radius=9mm, resolution=800x600
Extended Umbra Culling

- Exploit rasterization/geometry **ambiguity**
Extended Umbra Culling

- We can extend umbra without error!

- ~2 times bigger **geometry** rasterized with the same pixel pattern
Extended Umbra Culling

Extend pixel geometry to ~2 pixels
- Same result due to sampling ambiguity
Extended Umbra Culling

Number of layers:
- 10 layers always enough (independent of scene)

for FOV=30, near=1m, far=∞, lens radius=9mm, resolution=800x600

In practice,
- 3-4 layers sufficient
2) Scene Representation

- Example results with *Extended Umbra Culling*
2) Scene Representation

- Example results with *Extended Umbra Culling*

![Our method (3 layers)](image1)

![Reference method](image2)
3) Ray Tracing
3) Ray Tracing

- Image-based ray tracing: trace the footprint of ray

footprint of lens ray on the image plane
3) Ray Tracing

- Footprint view

Far plane

Max. depth

Min. depth

Near plane

footprint of lens ray
3) Ray Tracing

- Bounding the ray to a segment: much shorter footprint
3) Ray Tracing

- Intersection test with the footprint

No intersection

Near plane
Configuration

Hardware:
- NVIDIA GeForce GTX285
- Microsoft DirectX 10

Software:
- Resolution: 800x600
- Number of rays: 100 samples
- Depth-peeling layers: 4 layers
Thin Lens Model: Comparison

- PRE (Depth peeling)
- RT (Ray tracing)
- MS (Lee et al., SIG. ASIA 2009)
- REF (Haeberli, 1991): accumulation buffer

Town Scene: 98K tri.

- OUR: 10 ms (100 fps)
- MS: 15 ms (67 fps)
- REF: 125 ms (8 fps)
Thin Lens Model: Comparison

- PRE (Depth peeling)
- RT (Ray tracing)
- MS (Lee et al., SIG. ASIA 2009)
- REF (Haeberli, 1991): accumulation buffer

Dragon Scene: 935K tri.
Geometric Lens

Constant overhead
e.g., Spherical Lenses: (roughly 10 ms) to test the lens intersection twice
In Practical Scenario

- Thin-lens model
- 100 K triangles
- 1024x768
- Moderate blur: 3 layers and 32 lens samples

Mostly above 100 fps
Performance

Fully dynamic:

    scenes

    lens properties

    focal distance
Focus Control
Spherical / Thin Lens Models

Adapt focal distance: Simple and common usage
Realistic Lens Models

Tilt-shift photography: tilting focal plane by tilting a lens
- Non-intuitive
- hard to control
Realistic Lens Models

Our intuitive **tilt-shift control**

focal plane = 3 control points

>3: **least-square fit**
Focus can guide observer

How to achieve and control such behavior?
Beyond Physics

Focal Surface
- Defined via control points
- Moving least-square fit
- Keyframed

We want to focus only on the ogre's eye, while blurring its face
Beyond Physics
Conclusion
Recap

**High performance/quality**
- Culling and efficient ray tracing

**Flexible solution**
- Diverse lens blur effects
- Optical aberrations

**Focus control**
- Physical and expressive results