# The Occlusion Camera

Chunhui Mei

Voicu Popescu

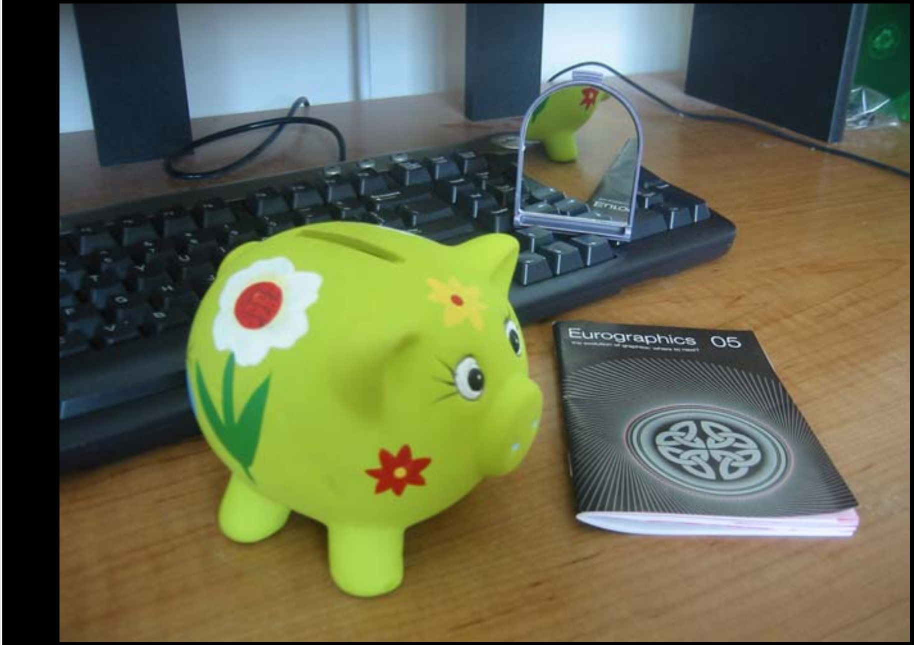
Elisha Sacks

PURDUE
UNIVERSITY

# Image-Based Rendering (IBR)

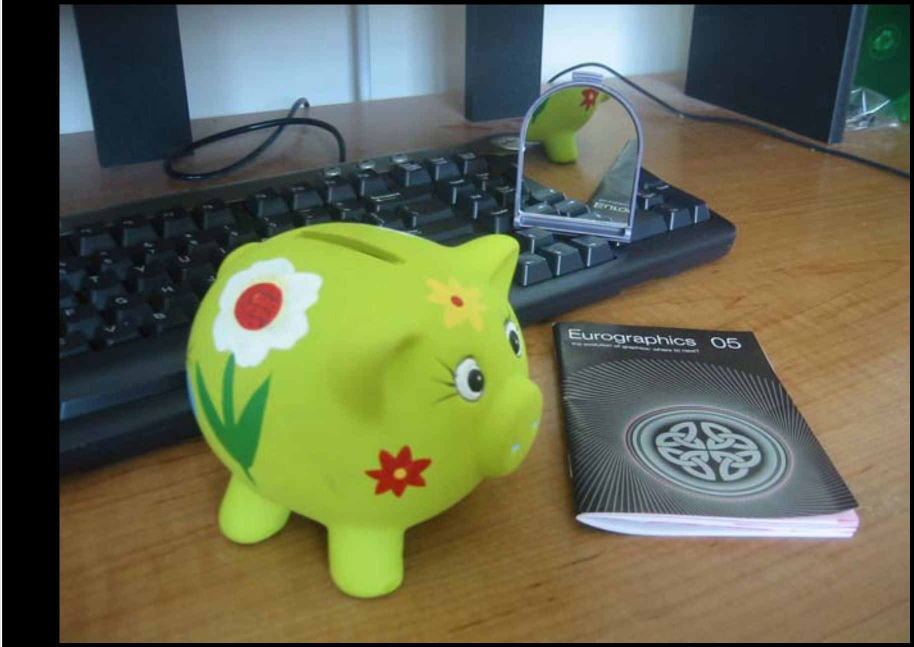
- Definition
  - Computing an image of a scene *[from a novel view]* by interpolating between *[depth and]* color samples
- Advantages
  - *Quality*: photorealism transferred from input to output
  - *Efficiency*: scene independent rendering cost

# IBR challenges

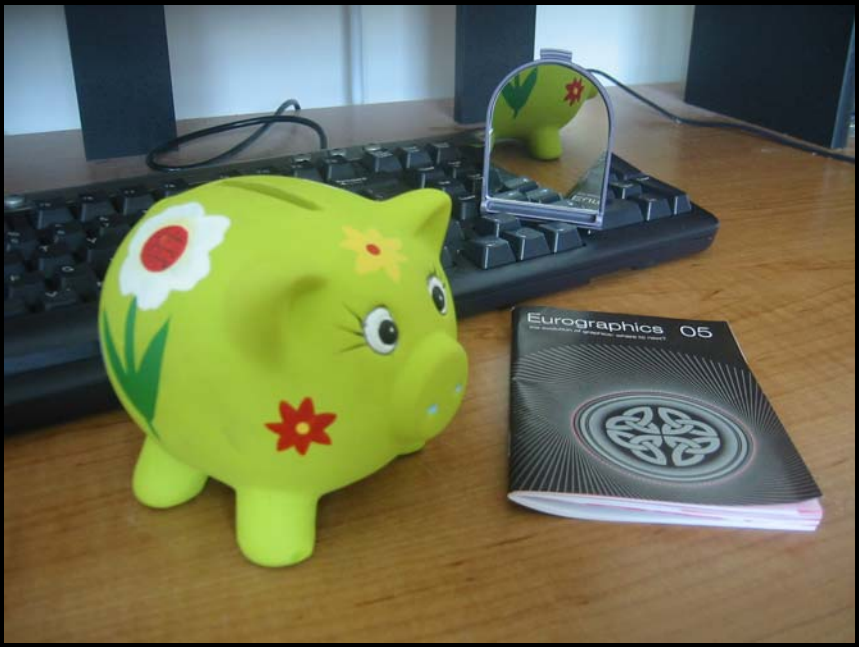


*reference image*

# IBR challenges

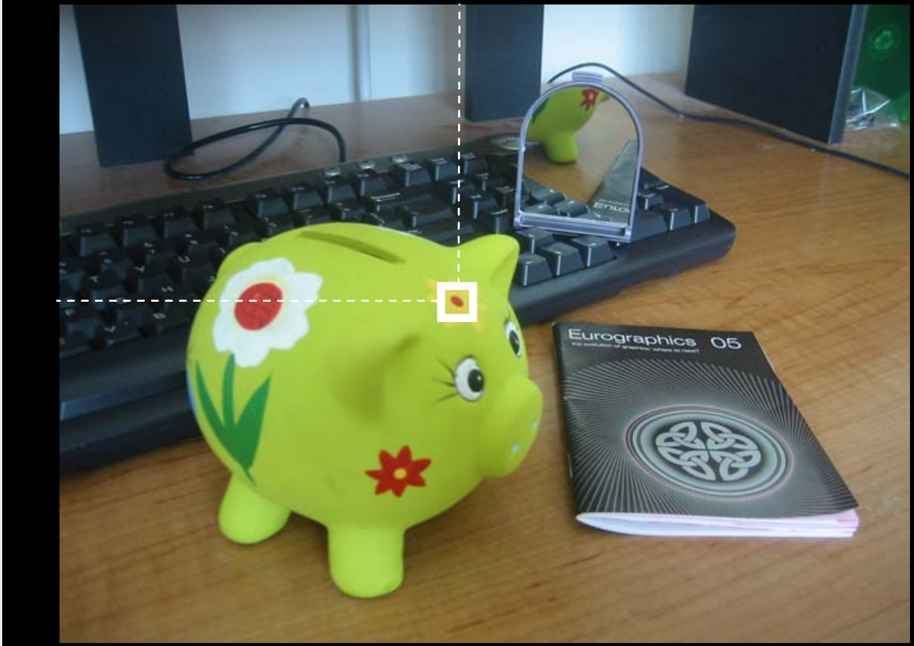


*reference image*

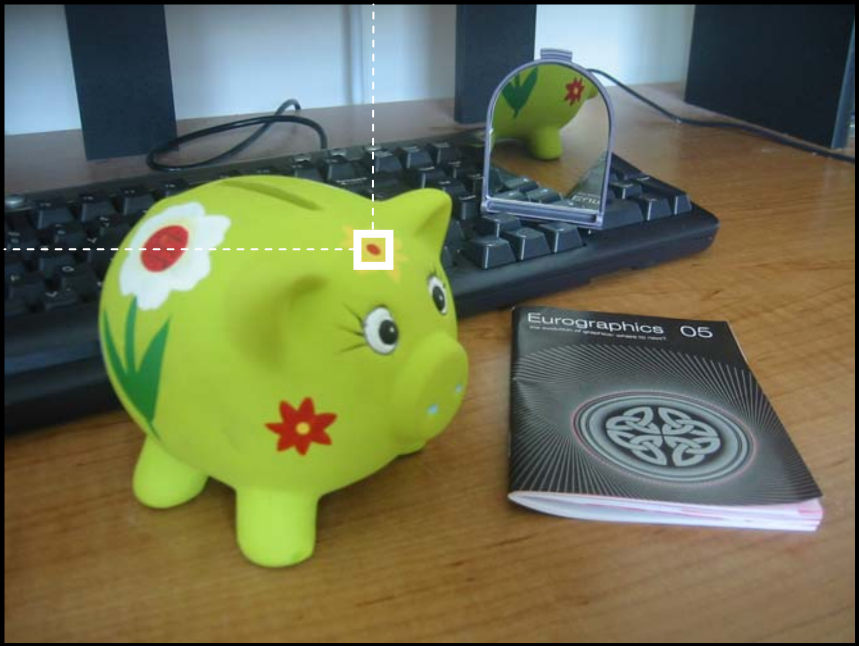


*desired image*

# IBR challenges



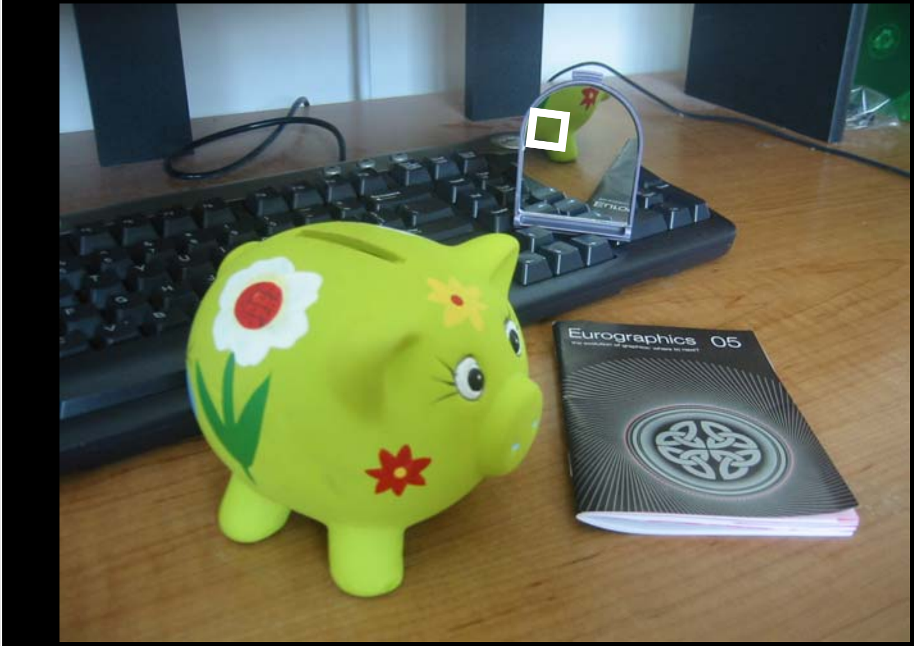
*reference image*



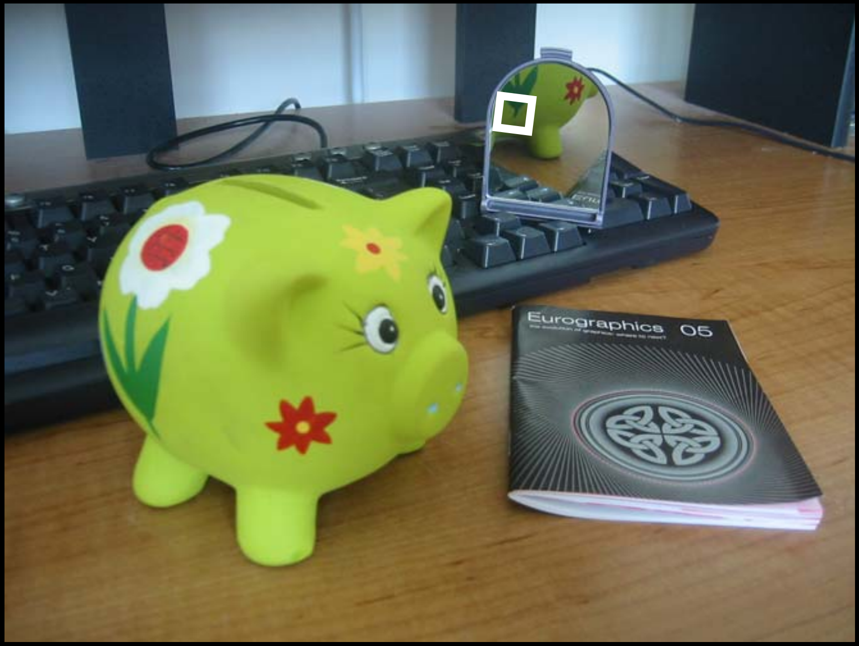
*desired image*

- Acquisition
  - depth extraction, registration

# IBR challenges



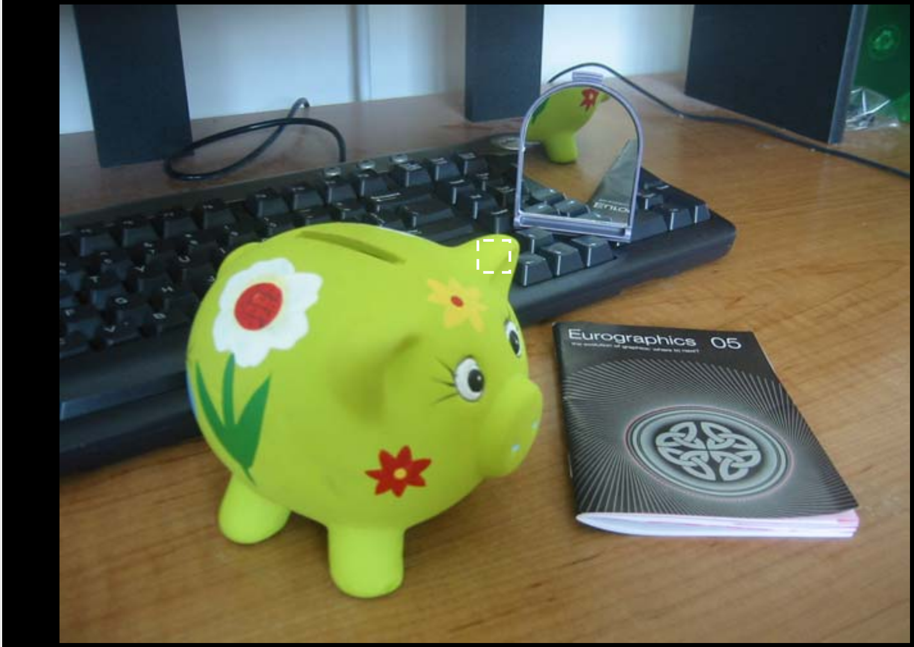
*reference image*



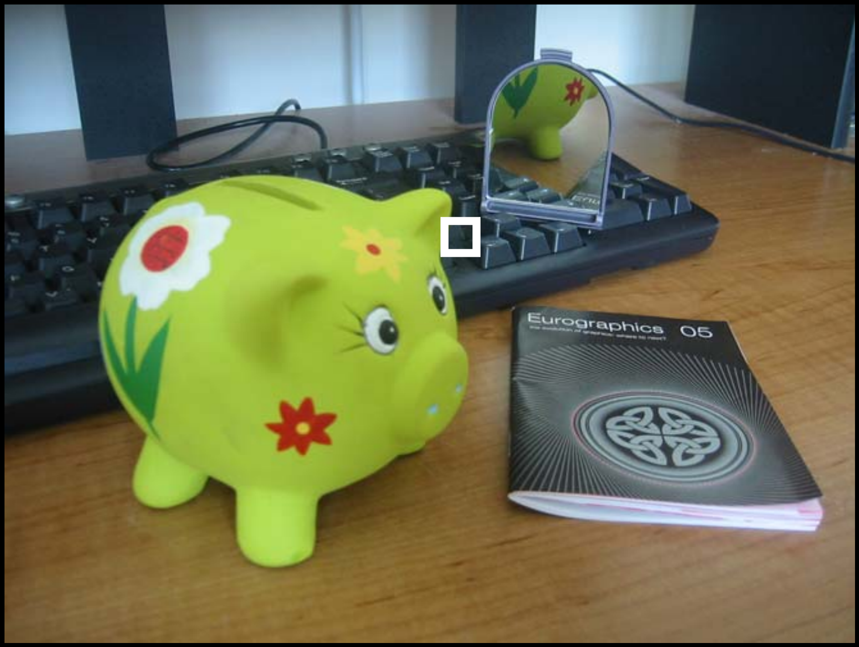
*desired image*

- View dependent appearance
  - – Reflections, refractions, blending at depth discontinuities

# IBR challenges



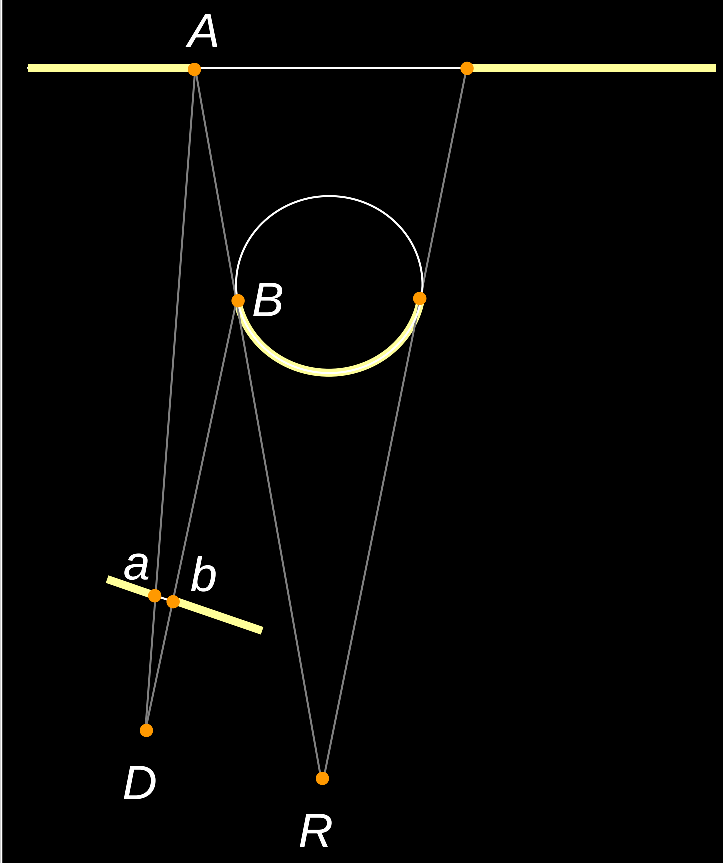
*reference image*



*desired image*

- Disocclusion errors

# Disocclusion errors

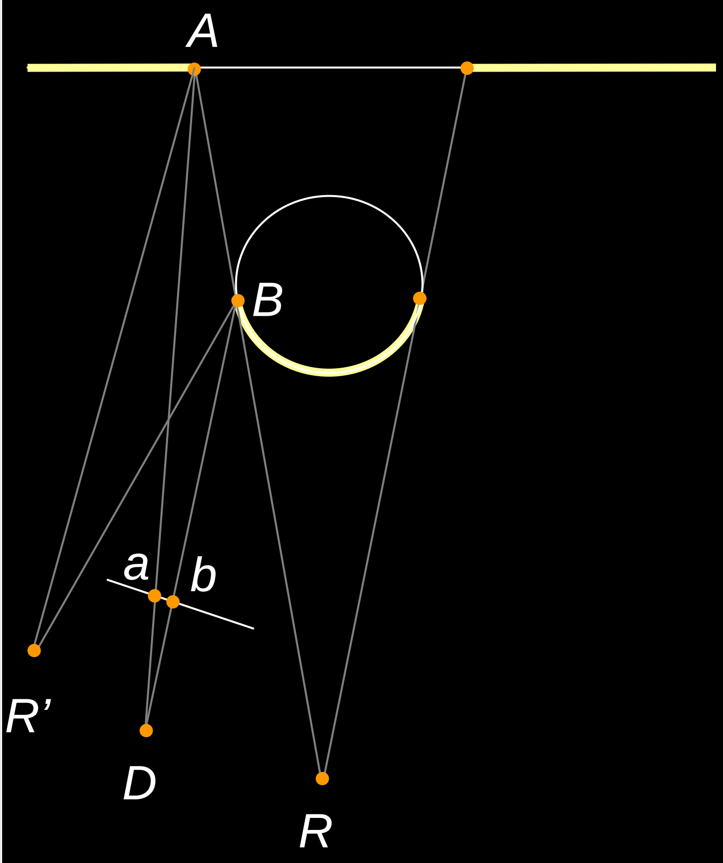


*R:* reference depth image [McMillan 95]

*D:* desired image

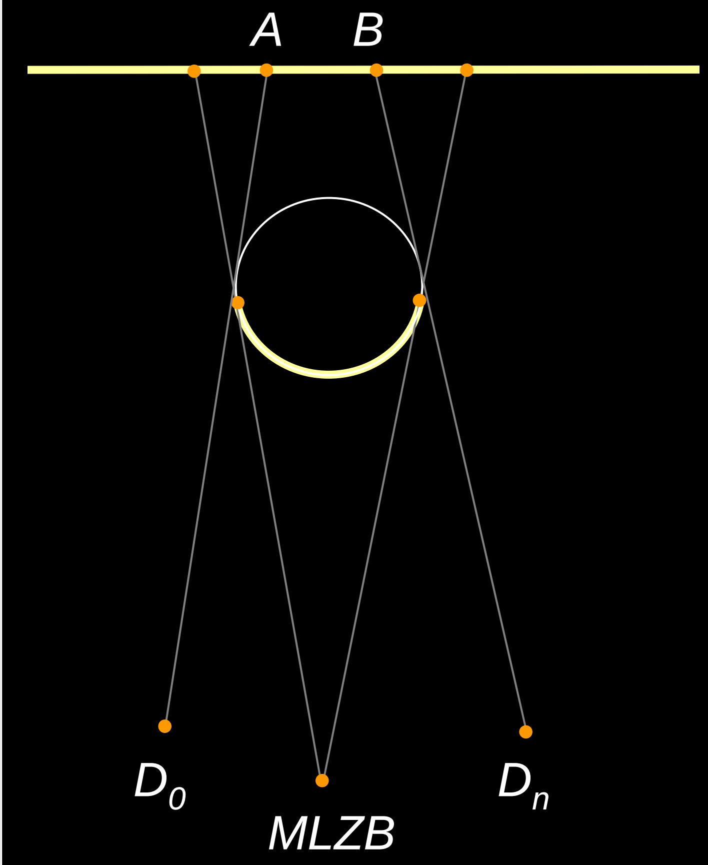
*(a, b)*: disocclusion error

# Multiple depth images [McMillan97, Mark97]



- Repeated samples
- Scene dependent cost
- Heuristic

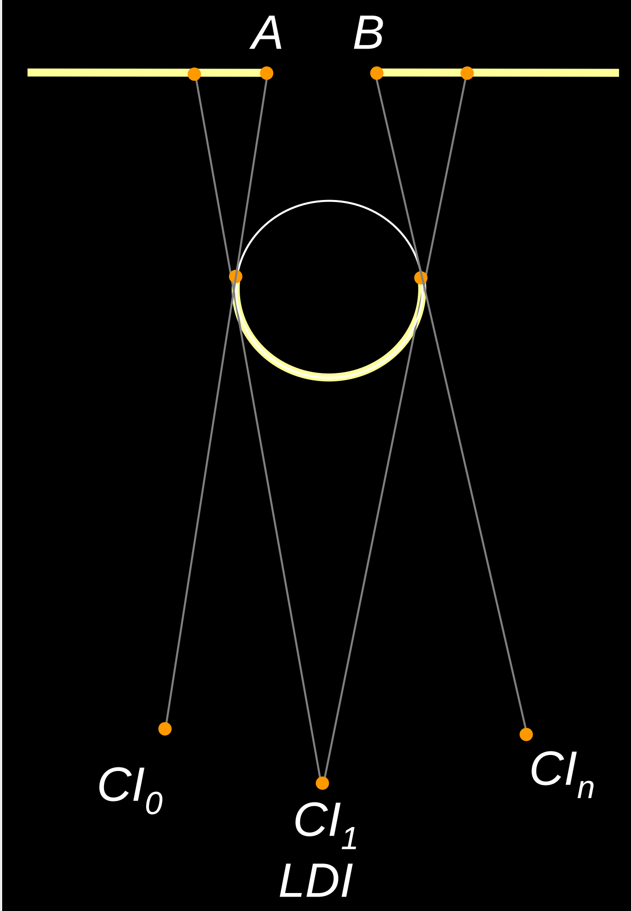
# Multi-layered z-buffers [Max95]



- + Unique samples
- - Scene dependent cost
- - Some samples never needed
- - Irregular data structure
- - Expensive preprocessing

# Layered-Depth Images

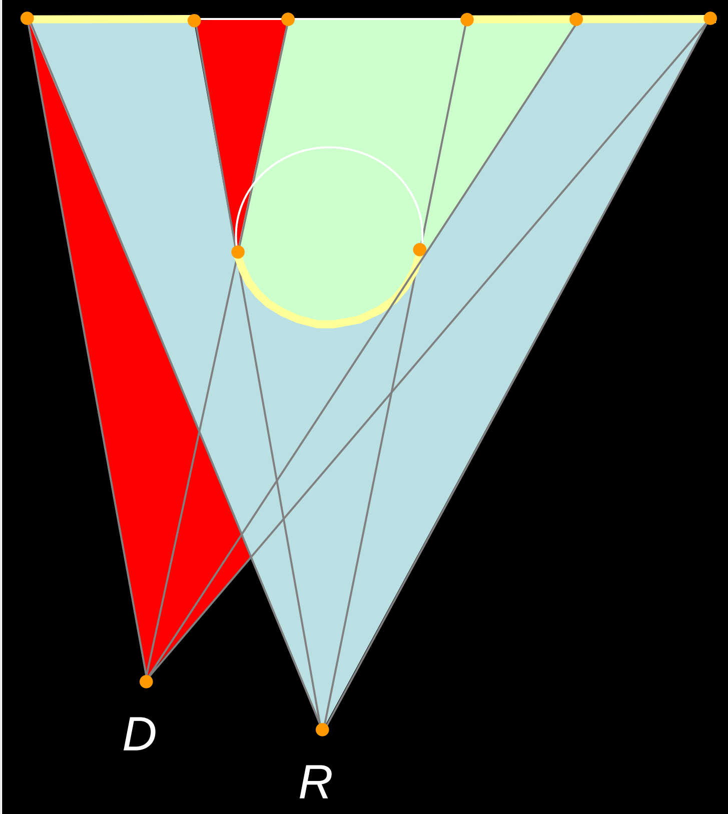
[Shade98, Popescu98, Chang99]



+ Unique samples
+ All samples eventually needed
- Scene dependent cost
- Heuristic
- Irregular data structure
- Expensive preprocessing

PURDUE UNIVERSITY

# The Vacuum Buffer [Popescu01]



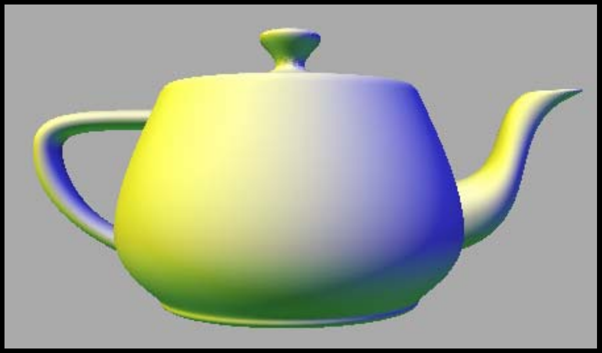
- + Conservative
- - Scene dependent cost
- - Expensive run-time algorithm

# Approach

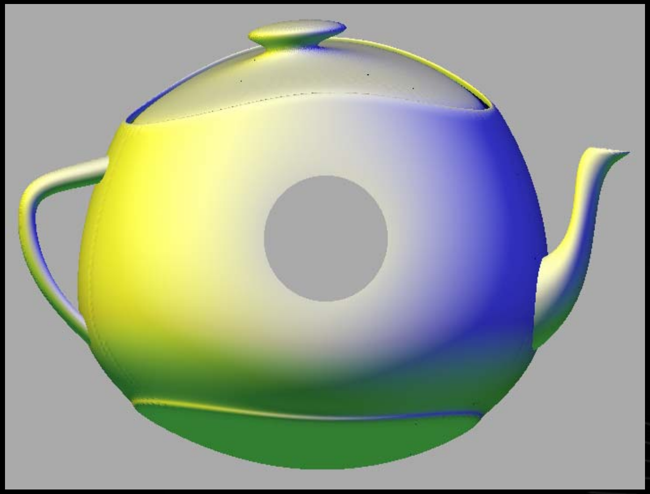
- Prevent rather than remove disocclusion errors
- Occlusion camera
  - Non-pinhole camera that sees around occluders
  - Fast projection: efficient feed-forward rendering

# Approach

- Occlusion camera reference image (OCRI)
  - Hidden samples likely visible in nearby views
  - Single layer (bounded, implicit connectivity, incremental processing)

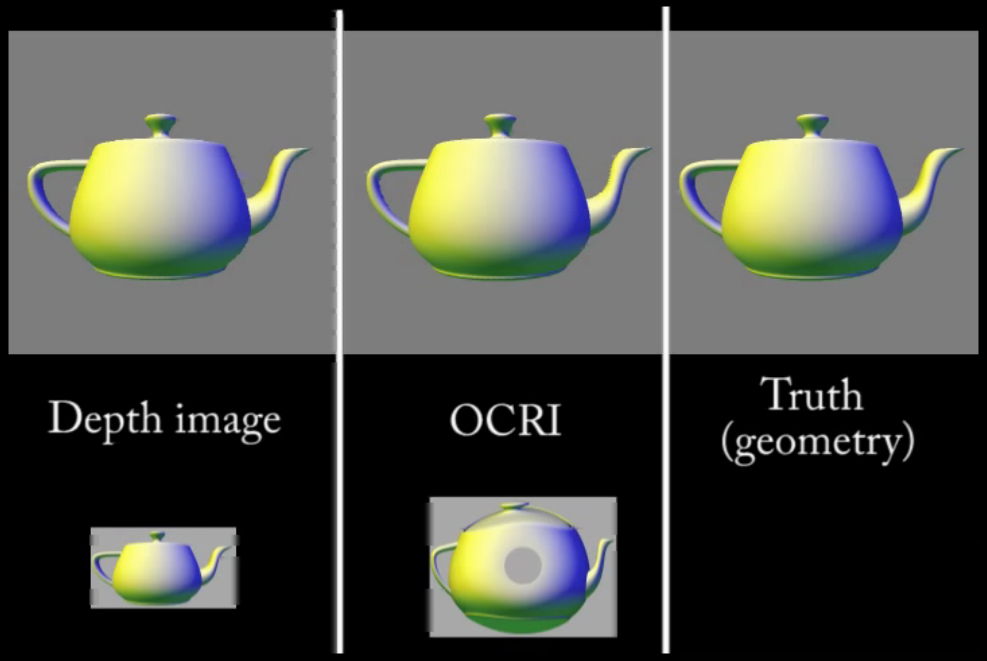


*depth image*



*OCRI*

# Teapot example



# Outline

- Introduction
- Occlusion camera model
- OCRI construction
- Rendering with OCRI
- Discussion

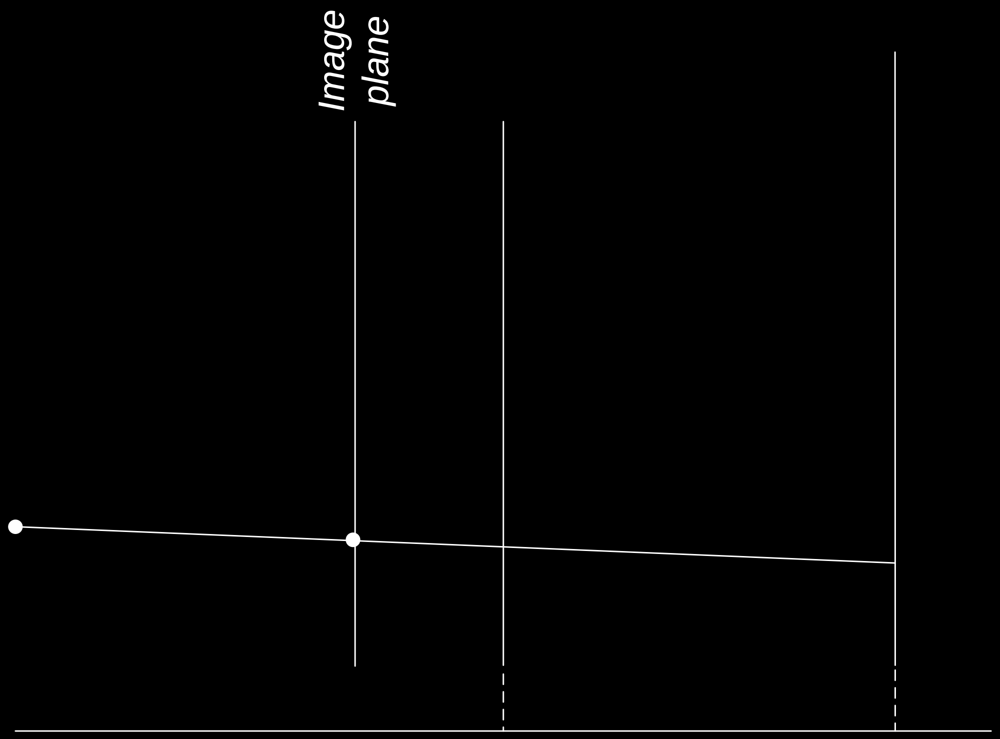
# Outline

- Introduction
- Occlusion camera model
- OCRI construction
- Rendering with OCRI
- Discussion

PURDUE
UNIVERSITY

# Occlusion Camera Model

3D radially distorted planar pinhole camera



*(PPHC, $u_0$, $v_0$, $z_n$, $z_f$, $d_n$, $d_f$)*

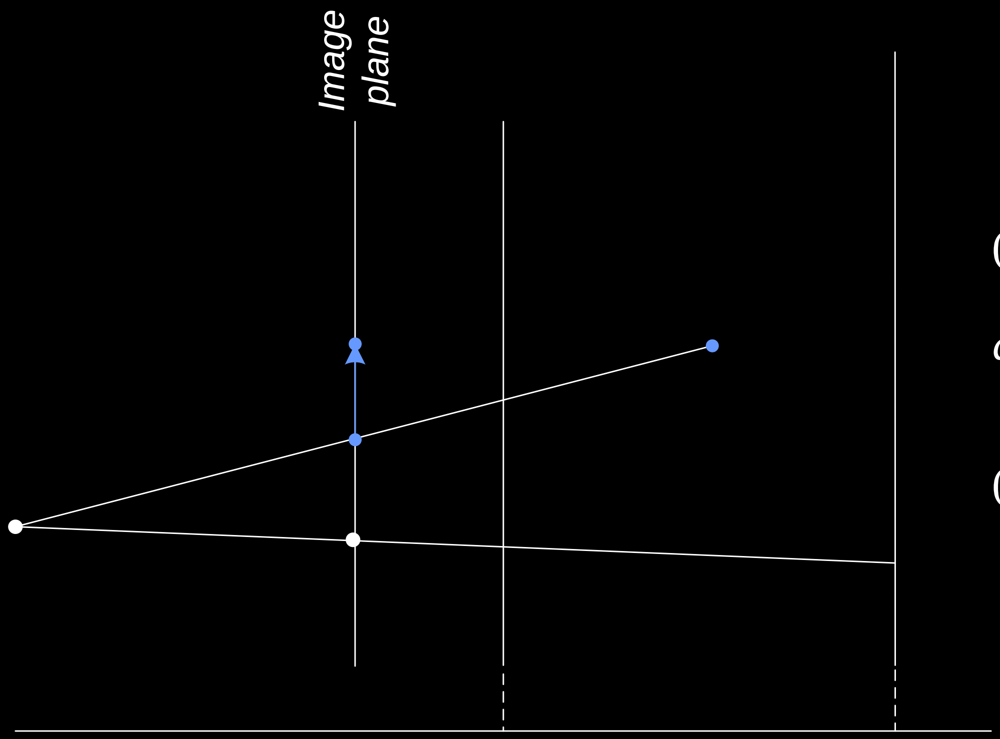
*PPHC* planar pinhole camera

*($u_0$, $v_0$)* pole pixel coordinates

*($z_n$, $z_f$)* near and far distortion planes

*($d_n$, $d_f$)* near and far distortion magn.

# Projection

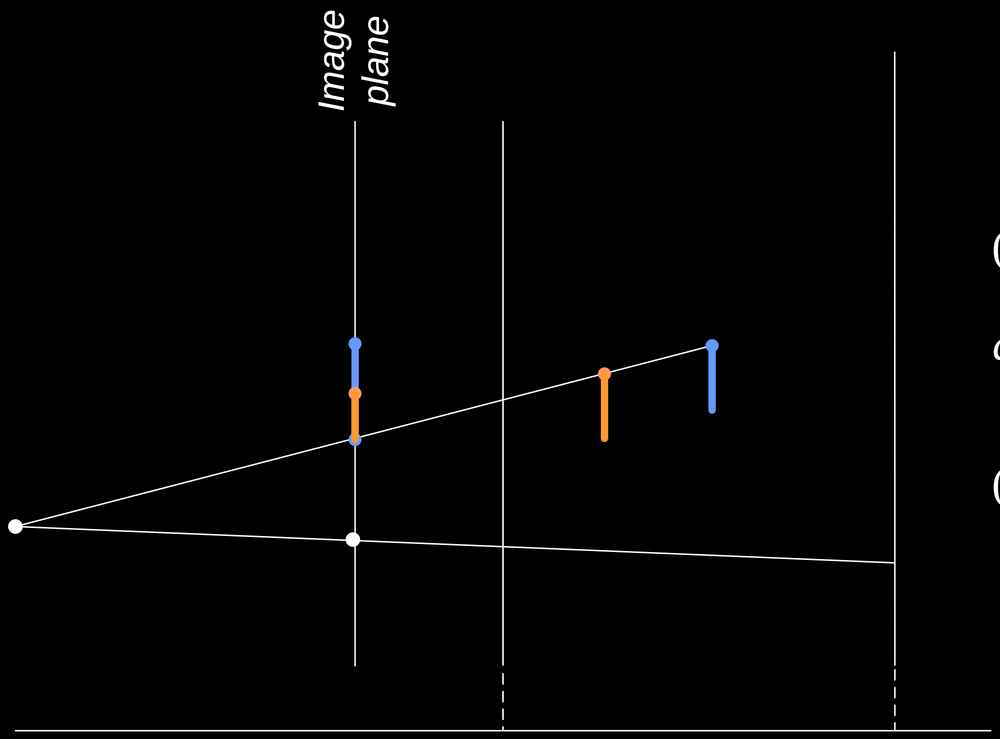


$$(u_u, v_u, z) = PPHC(P)$$

$$d(z) = d_n + \frac{1/z_n - 1/z}{1/z_n - 1/z_f}(d_f - d_n)$$

$$(u_d, v_d) = (u_u, v_u) + \frac{(u_u - u_0, v_u - v_0)}{\left\|(u_u - u_0, v_u - v_0)\right\|} d(z)$$

# Projection

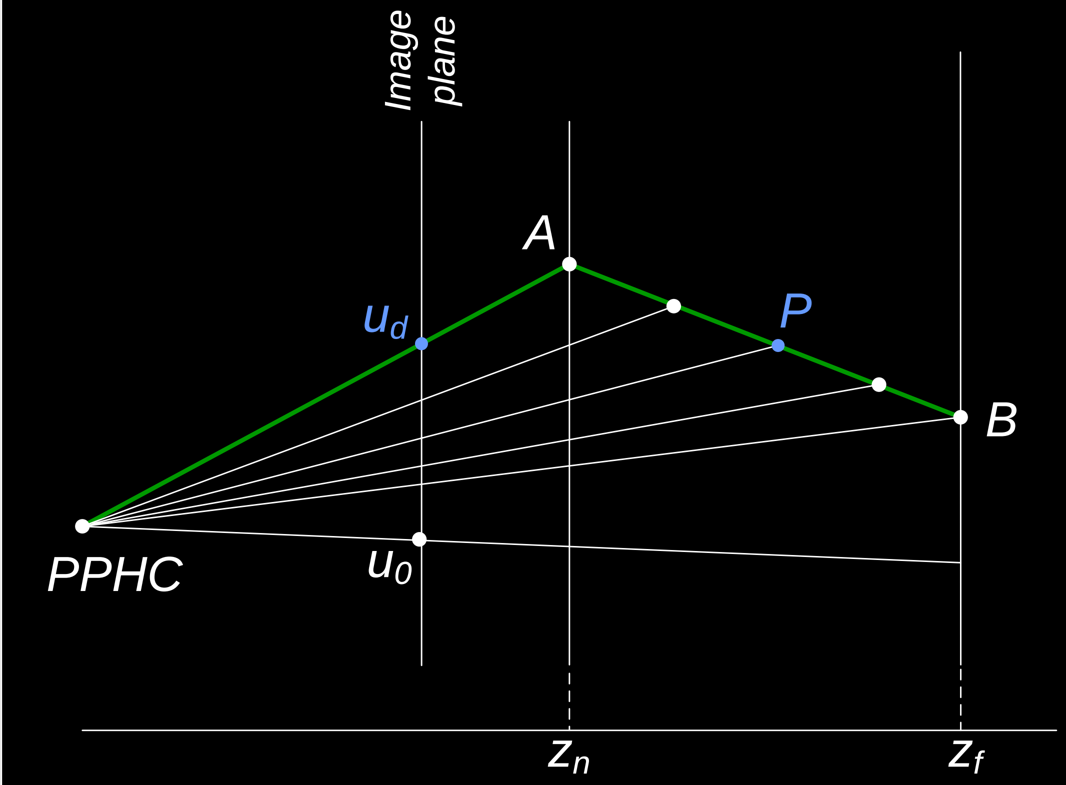


$$(u_u, v_u, z) = PPHC(P)$$

$$d(z) = d_n + \frac{1/z_n - 1/z}{1/z_n - 1/z_f}(d_f - d_n)$$

$$(u_d, v_d) = (u_u, v_u) + \frac{(u_u - u_0, v_u - v_0)}{\|(u_u - u_0, v_u - v_0)\|} d(z)$$

# Camera rays

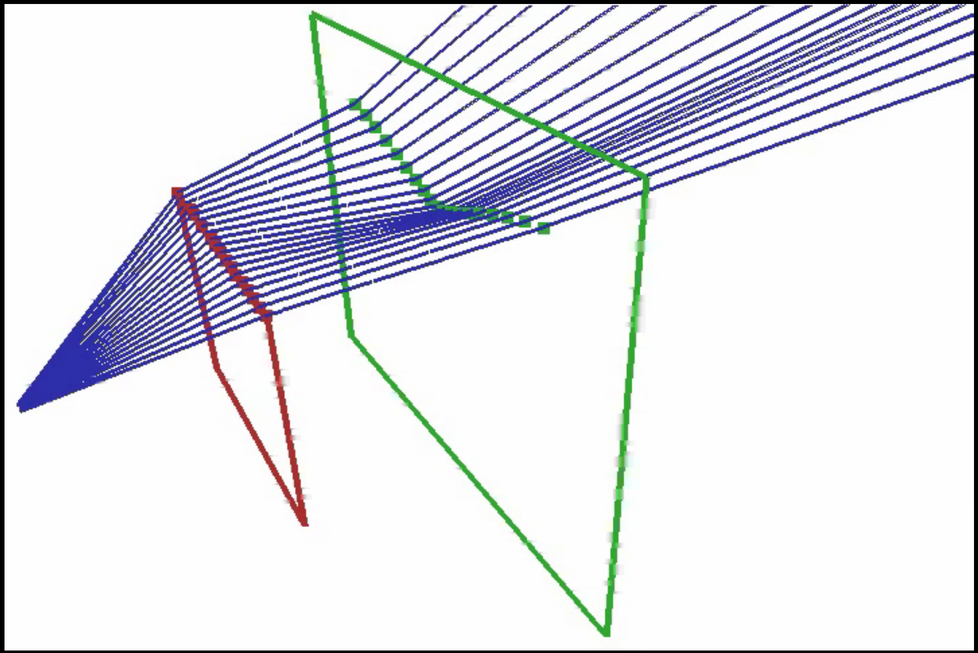


- *Ray(u, v)* = locus of 3D points that project at *(u, v)*

- Line segment because distortion magnitude linear in *1/z*

- Could be a curve

# Camera rays



# Outline

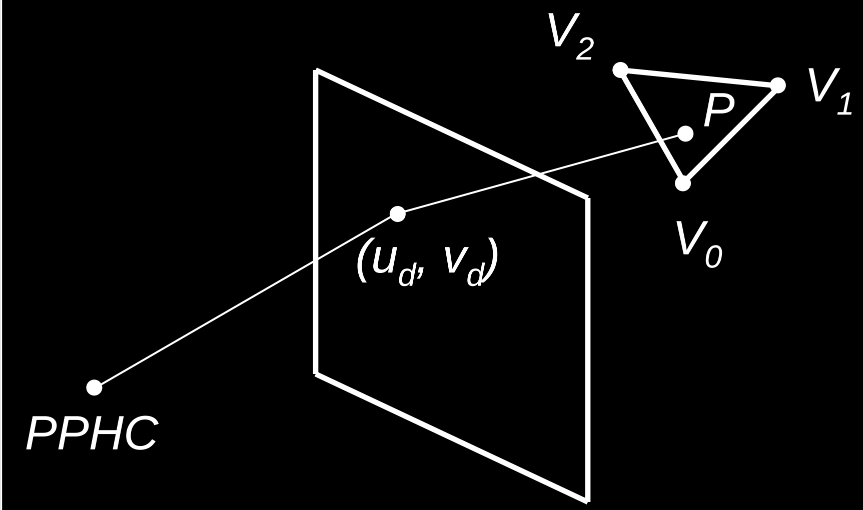
- Introduction
- Occlusion camera model
- OCRI construction
- Rendering with OCRI
- Discussion

# OCRI construction

- Render scene with occlusion camera
  - Challenge: lines image as curves
  - Solution 1: subdivide scene triangles
  - Solution 2: rasterize in distorted domain

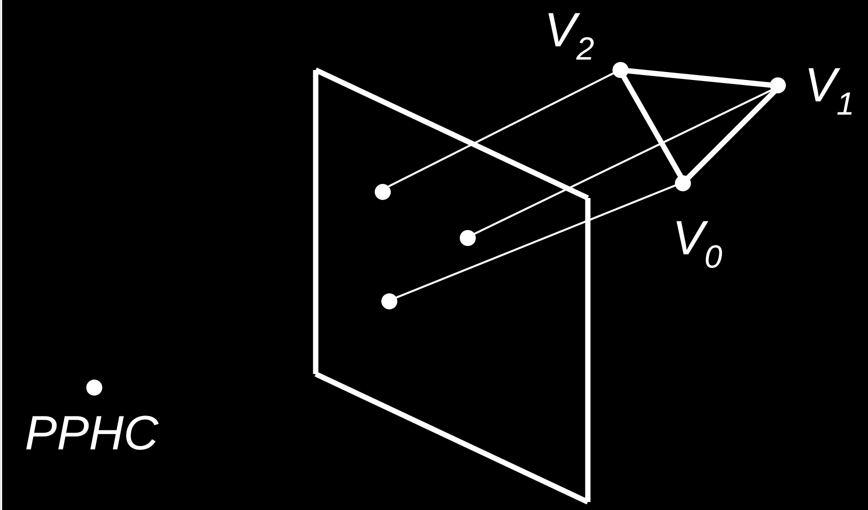
# Distorted domain rasterization

- Possible
  - Given a scene triangle $t(V_0V_1V_2)$ and a pixel $(u_d, v_d)$
  - There is at most one $P$ in $t$ that projects at $(u_d, v_d)$



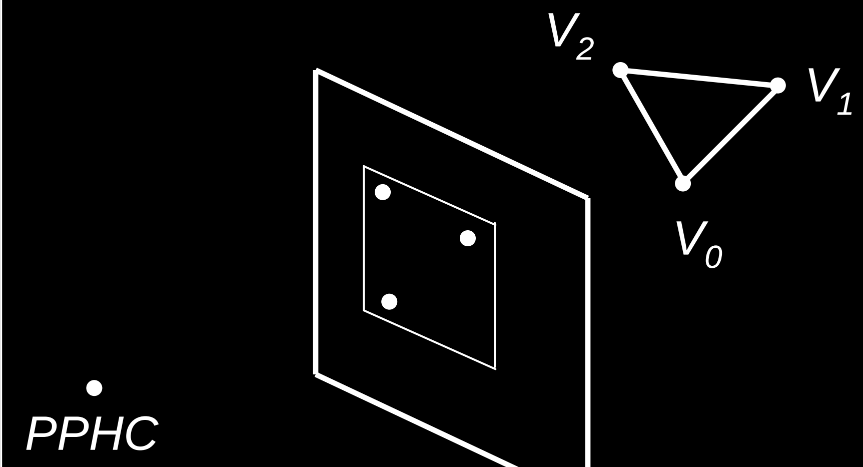
# Distorted domain rasterization

- Algorithm
  - Project $V_0$, $V_1$, and $V_2$

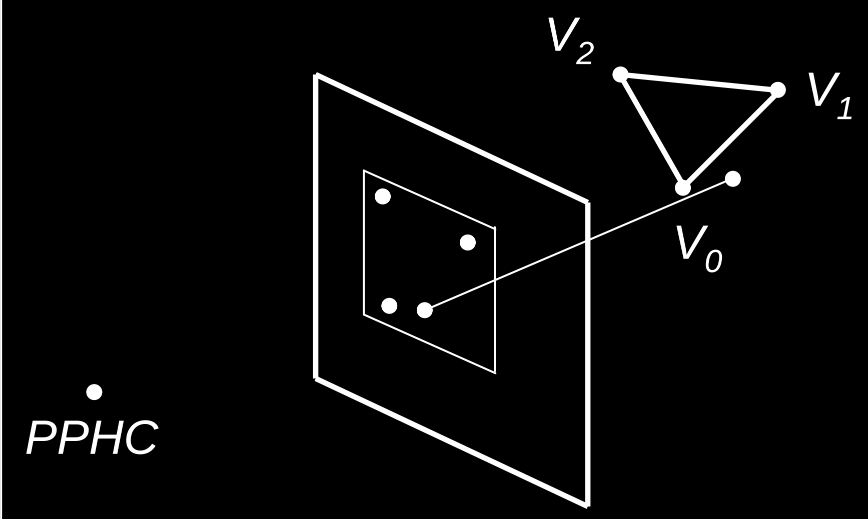


# Distorted domain rasterization

- Algorithm
  - Project $V_0$, $V_1$, and $V_2$
  - Approximate bbox

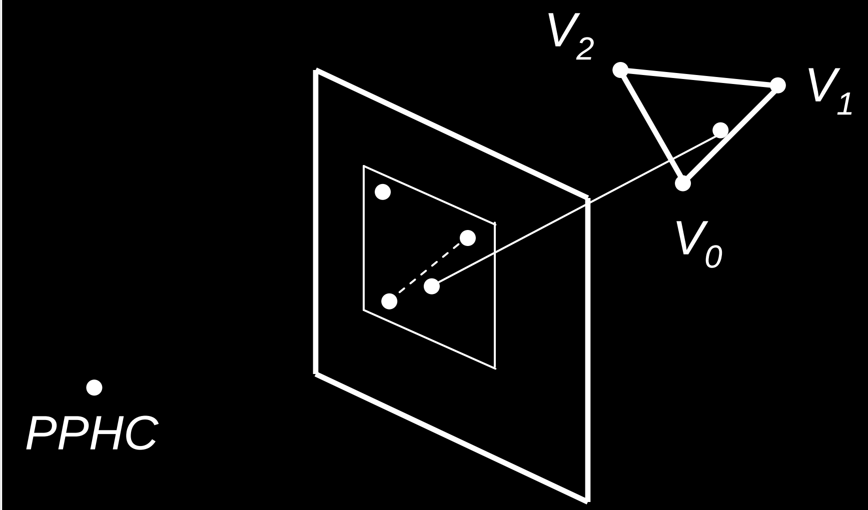


# Distorted domain rasterization



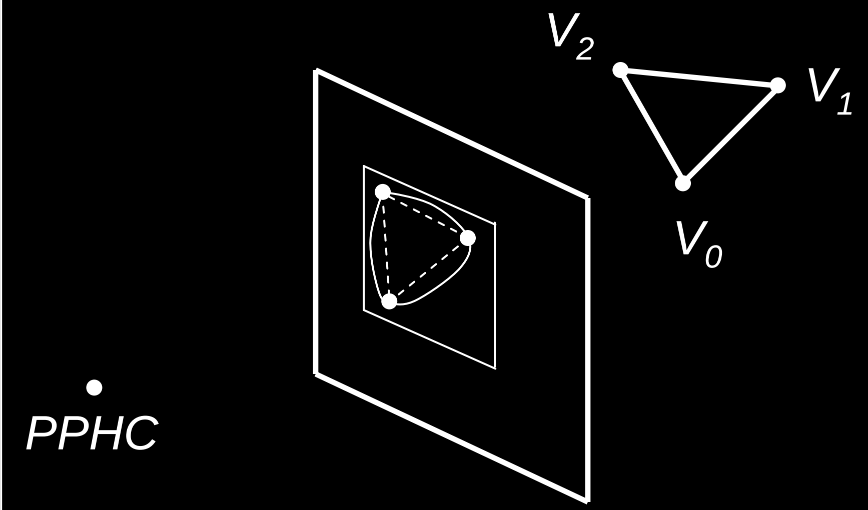
- Algorithm
  - Project $V_0$, $V_1$, and $V_2$
  - Approx. bound. box
  - For all $(u_d, v_d)$ in *bb*
    - Unproject
    - If outside triangle, discard

# Distorted domain rasterization



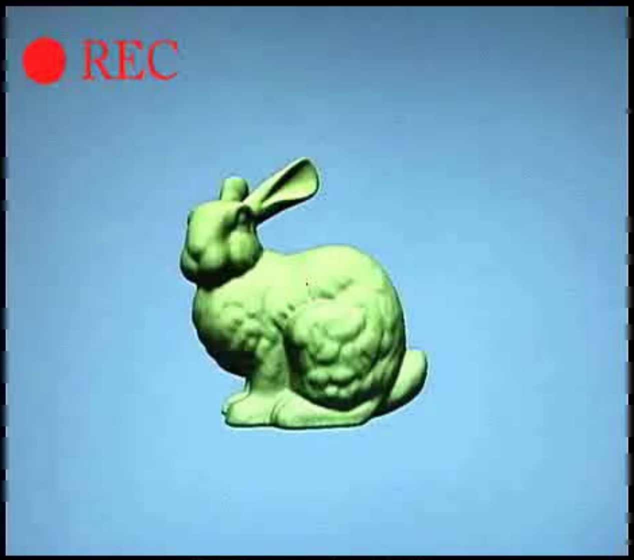
- Algorithm
  - Project $V_0$, $V_1$, and $V_2$
  - Approx. bound. box
  - For all $(u_d, v_d)$ in *bb*
    - Unproject
    - If outside triangle, discard

# Distorted domain rasterization



- Algorithm
  - Project $V_0$, $V_1$, and $V_2$
  - Approx. bound. box
  - For all $(u_d, v_d)$ in *bb*
    - Unproject
    - If outside triangle, discard
    - Z-buffer
    - Shade

# OCRI construction on GPU



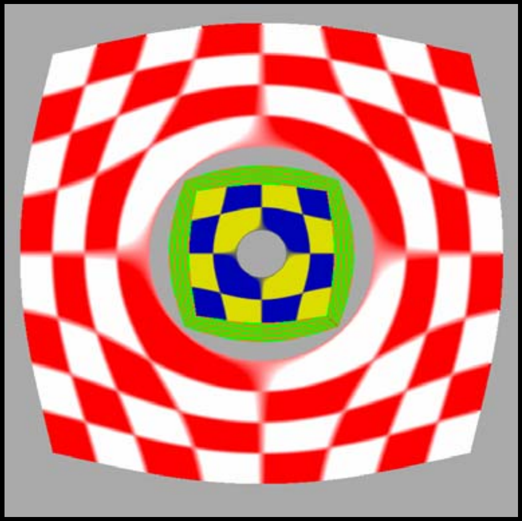
70K triangles 11fps

# Outline

- Introduction
- Occlusion camera model
- OCRI construction
- Rendering with OCRI
- Discussion

# OCRI samples

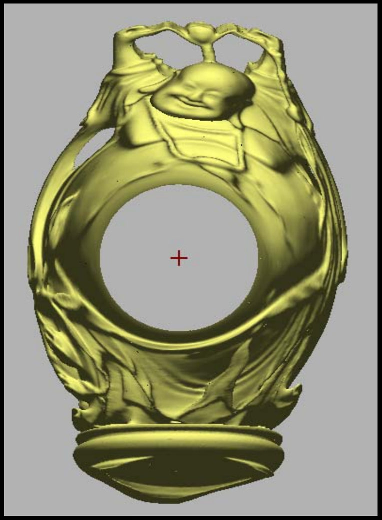
- Occlusion camera + framebuffer + zbuffer = implicit, regular mesh of *(x, y, z, R, G, B)* samples



*OCRI*

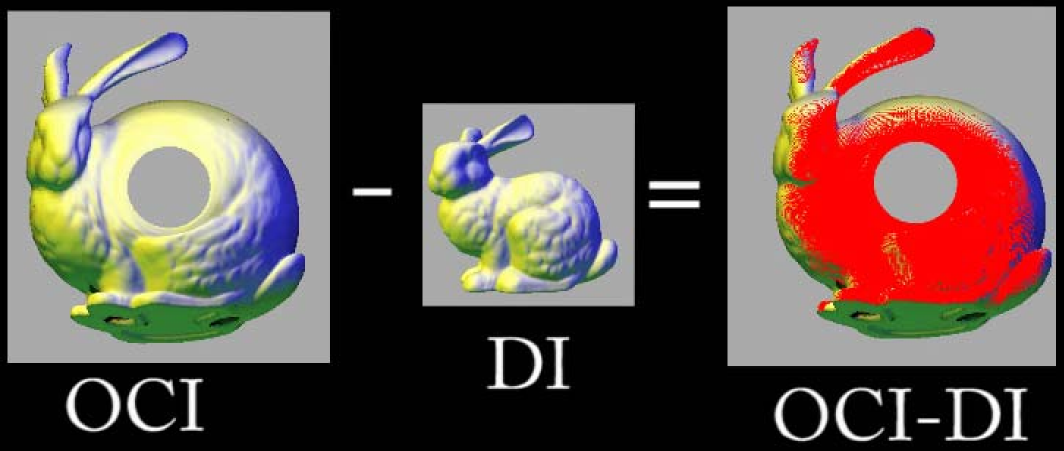
# Complex geometry

- OCRI does not guarantee that all disocclusion errors are avoided
- OCRI could be missing samples visible from reference view



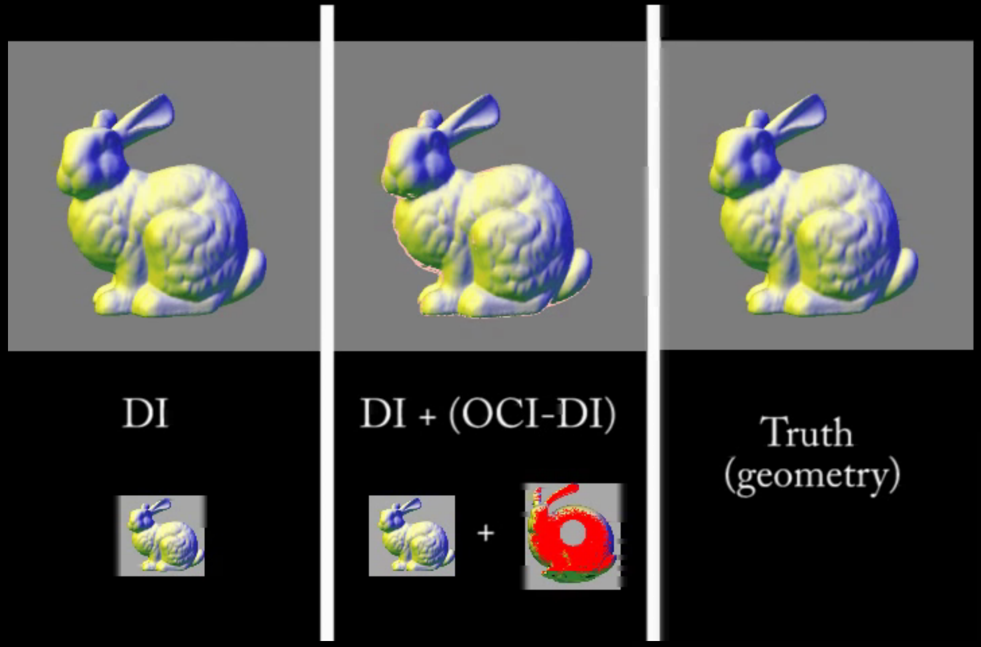
# RI = DI + (OCI-DI)

- Depth image enhanced with occlusion camera image contribution

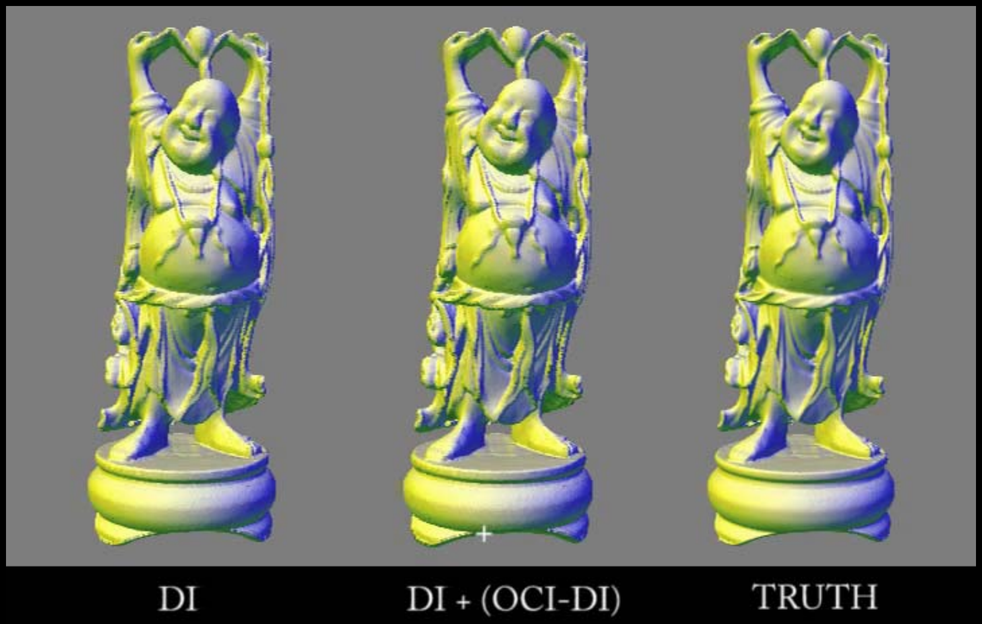


# RI = DI + (OCI-DI)

- Depth image enhanced with occlusion camera image contribution



# Happy Buddha example

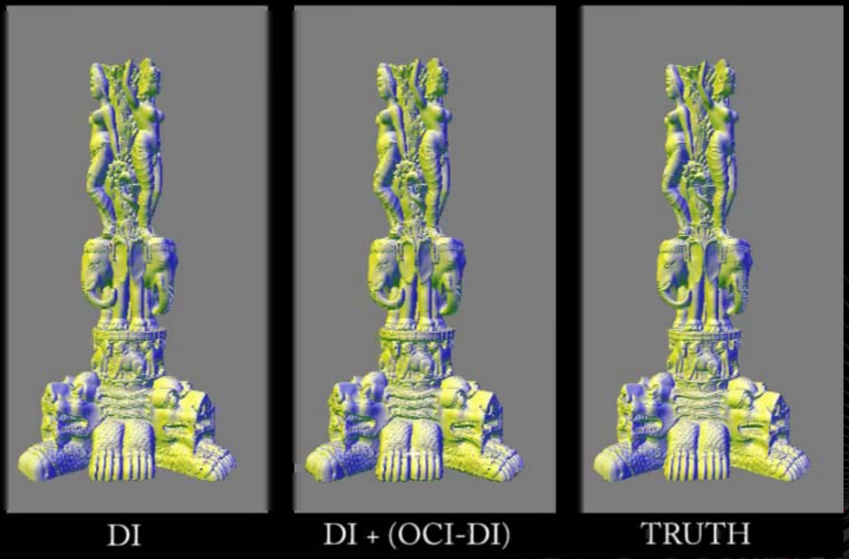


# Discussion

- Occlusion cameras: a novel class of cameras
  - gather hidden samples, likely to be needed
  - trade *(u, v)* resolution for resolution along same ray
  - are defined by the reference view *& geometry seen*
  - are not pinholes, but offer unambiguous projection
- This paper: single-pole occlusion camera
- Future: fine grain control of distortion

# Acknowledgments

- Members of Purdue Graphics and Visualization Laboratory
- Our reviewers
- Funding, HW, and SW
  - NSF, Purdue University Visualization Center, IBM, Intel, Microsoft



DI

DI + (OCI-DI)

TRUTH