*Voicu Popescu, Chunhui Mei, Jordan Dauble, and Elisha Sacks*
Purdue University

# Reflected-Scene Impostors for Realistic Reflections at Interactive Rates

## Reflections—a difficult problem

- Every reflector is a portal onto a world which is as rich as the directly observed scene and which has complex image formation laws

2

## Prior work—vast

| | |
|---|---|
| Ray tracing | Image-Based Rendering |
| Feed-forward reflection rendering | Approximation of reflected scene |

3

## Problem of rendering reflections

- Compute
  - Intersection with reflector
  - Reflected ray
  - Intersection with reflected scene
  - antialiasing

4

## Problem of rendering reflections

- Compute
  - Intersection with reflector
  - Reflected ray
  - Intersection with reflected scene
  - antialiasing

"OpenGL"

???

5

## Reflected-scene approximation

- Reflected scene replaced with approx. that provides
  - Fast intersection with ray
  - Antialiasing

6

## Reflected-scene approximation

- Example: environment mapped reflections
  - Reflected scene infinitely far away
  - Straight forward intersection with ray
  - Antialiasing computed in 2D (mipmapping)

7

## Reflected-scene approximation

- Example: environment mapped reflections
  - Reflected scene infinitely far away
  - Straight forward intersection with ray
  - Antialiasing computed in 2D (mipmapping)
  - Drastic approximation, incorrect results close to the reflector

8

## Our approach

- Approximate reflected scene with impostors
  - Considerable prior work on impostors
  - Reflector surface prevents desired viewpoint from getting too close to the impostor
  - Reflection distortion hides impostor artifacts

9

## Impostor requirements

- Impostor has to provide
  - Fast construction
  - Fast intersection with ray
  - Antialiasing

10

## Results: billboard impostors

11

## Results: depth image impostors

12

## Billboard impostors

- Replace reflected object with billboard
- Higher order reflections
  - Reflective billboards (normal mapped quads)

13

## Billboard impostors

- Impostor has to provide
  - Fast construction YES
  - Fast intersection with ray YES
  - Antialiasing YES

14

## Pixel algorithm

- For *D* diffuse, *R* reflective billboards, and maximum reflection order *K*
  - Compute reflected ray *r*
  - For reflection order 1 to *K*
    - Intersect with (*D*+*R*-1) billboards
    - If no intersection
      - return *EM(r)*
    - Else if intersection with diffuse billboard $DB_i$
      - return $DB_i(r)$
    - Else if intersection with reflective billboard $DB_i$
      - $r = DB_i(r)$

15

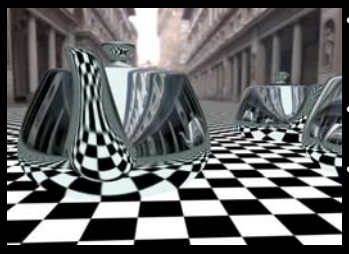
## Pixel algorithm

- For *D* diffuse, *R* reflective billboards, and maximum reflection order *K*
  - Compute reflected ray *r*
  - For reflection order 1 to *K*
    - Intersect with (*D*+*R*-1) billboards
    - If no intersect
      - return *EM(r)*
    - Else if interse
      - return $DB_i(r)$
    - Else if intersection with reflective billboard $DB_i$
      - $r = DB_i(r)$

$O(K*(D+R))$

16

## Example: 4 teapots



- *D* = 1, *R* = 4, *D*+(*R*-1)+*D* = 5 intersections / pix
- 12 second order reflections
- 40fps

17

## Example: table scene

- *D* = 2, *R* = 2, *D*+(*R*-1)+*D* = 5 intersections / pix
- 2 second order reflections
- 33 fps

18

## Example: table scene

- *D* = 2, *R* = 2, *D+(R-1)+D* = 5 intersections / pix
- 2 second order reflections
- 33 fps

19

## Example: table scene

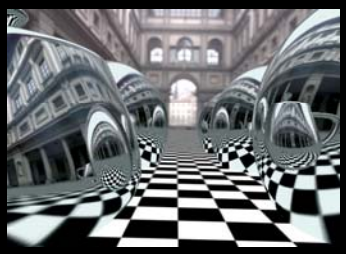
20

## Example: pushing-it scene

- *D* = 2, *R* = 9, *D+(R-1)+D* = 11 intersections / pix
- 72 second order reflections
- 11 fps

21

## Example: pushing-it scene



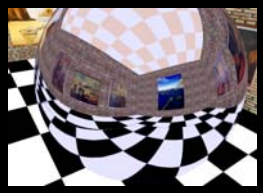
- *D* = 2, *R* = 9, *D+(R-1)+D* = 11 intersections / pix
- 72 second order reflections
- 6 fps

22

## Example: pushing-it scene

23

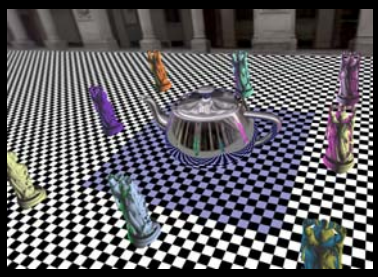
## Problem



Transition from impostor to environment map (*red* in left image) is discontinuous.

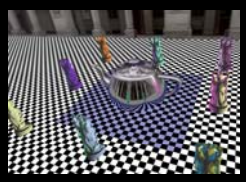
24

## Solution: ray morphing



25

## Solution: ray morphing



26

## Solution

*Left*—continuous transition. *Right*—morph region (*green*), environment map (*red*).

27

## Ray morphing

28

## Attenuation w/ distance

29

## Fresnel

30

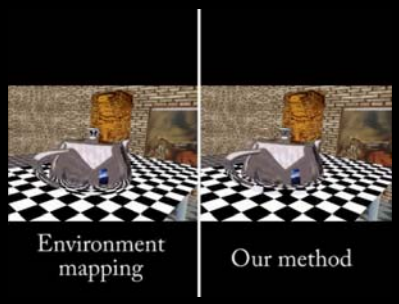
## Combined effects

31

## Animation and materials

32

## Comparison to env. mapping




33

## Billboard limitations

- No support for objects very close to the reflector
- Limited accuracy
  - Flat reflection
  - Lack of motion parallax

34

## Depth image impostors

- Impostor has to provide
  - Fast construction YES
  - Fast intersection with ray ???
  - Antialiasing YES

35

## Depth image—ray intersection

Epipolar-like constraints: intersection computed as 1D search
Still too many steps along epipolar segment

36

## Simplified Rotated Depth Maps

Pre-rotate depth map.
All rays ever needed project to rows.
Pre-simplify rows.

37

## Simplified Rotated Depth Maps

38

## SRDM construction cost

| *Number of segments* | 8 | 16 | 32 | 64 |
|---|---|---|---|---|
| *Construction time* [ms] | 210 | 300 | 480 | 980 |

Rigid body transformations, color updates, and reflector updates do not require reconstruction.

39

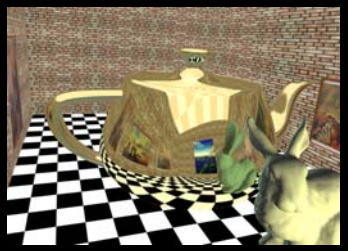
## Depth image impostor results

40

## Depth image impostor results

41

## Depth image impostor results



42

## Depth image impostor results



## Depth image impostor results



## Depth image impostor results



## SRDM under-sampling

One rotated depth map every $20^{o}$, $10^{o}$, $3^{o}$, and $2^{o}$, respectively.



## Depth image impostor results

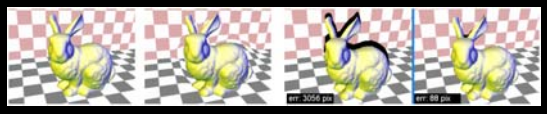


## Conclusions

- The reflected-impostor approach works
  - Fast, realistic
  - Increased modeling effort
- Rendering reflections reduced to the lesser problem of rendering w/ impostors

## Future work

- Other types of impostors
  - occlusion-resistant



49

## Future work

- Other types of impostors
- Other BRDFs
- Self-reflections
- Constructing the SRDMs on the GPU

50

## Acknowledgments


- Funding & equipment
  - NSF, Intel, Microsoft, Computer Science Purdue, Visualization Laboratory Purdue
- Stanford 3D Scanning Rep. for models
- Paul Debevec for environment maps
- Our graphics group at Purdue for miscellaneous but important help


51