EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)

Volume 41 (2022), Number 2

Dynamic Reflections Using Eye-Resolution Point Rendering

Abstract

We describe an improvement to the recently developed view independent rendering (VIR), and apply it to dynamic cube-mapped reflections. Standard multiview rendering (MVR) renders a scene six times for each cube map. VIR traverses the geometry once per frame to generate a point cloud optimized to many cube maps, using it to render reflected views in parallel. Our improvement, eye-resolution point rendering (EPR), is faster than VIR and makes cube maps faster than MVR, with comparable visual quality. We are currently improving EPR's run time by reducing point cloud size and per-point processing.

1. Introduction

As computer graphics hardware has improved, so has its interactive imagery, moving from line drawings to filled polygons, to textured surfaces with specular reflections. However, further improvements in visual realism — effects such as soft shadows, depth of field, and object reflections — have been hindered by current hardware, which requires multiple model traversals to render the many views needed to sample area lights, different focal depths, and reflections.

View-independent rasterization (VIR) avoids the complexity of multiple rendering passes [MWH17] by using points as a display primitive. For every frame, it carefully transforms input triangles into a point cloud specialized to the current set of views. VIR then renders these views in parallel using the point cloud, with an order of magnitude fewer passes over the geometry. In this research we:

- Introduce eye-resolution point rendering (EPR), a new sampling technique based on VIR, making sampling more parsimonious while remaining "watertight" (without holes).
- Apply EPR to dynamic cube-mapped reflections.
- Show that with a single pipeline pass, EPR generates cube maps with similar quality and superior speed to VIR and MVR.

2. Related Work

3. Eye-Resolution Point Rendering

EPR incorporates a fundamental change to VIR's sampling scheme. Where VIR ensures at least one point will be projected to every pixel in all off-screen buffers, EPR guarantees that at least one point will find its way to every pixel in the viewport (the eye's view). This reduces the size of the point cloud, an important improvement for environment mapping, which has more diversity of views than shadow mapping. On the other hand, this means that a single EPR point must often cover many pixels in off-screen buffers.

To form the point cloud, the geometry shader estimates each triangle's sampling density in the eye's view, and sets up a matching, unique rasterization. For reflection maps, this density is inversely proportional to the shortest distance light travels as it moves from the triangle via any reflective object to the eye. We conservatively bound this distance by approximating each reflective object with a bounding sphere. Each point has location, size, and orientation derived from its triangle, which the fragment shader uses to splat it across several buffer pixels. We dynamically manage off-screen buffer sizes to reduce point oversampling (lowering buffer pixels per point).

3.1. Dynamic Reflections using EPR

Cube mapping [BN76, Gre86] displays a reflective object by placing six images of the scene viewed from the object into off-screen buffers, and then mapping those images onto the object using specular reflection. If the scene is dynamic, off-screen imagery must be updated every frame; and if other objects are nearby, every reflective object requires a different cube map. EPR generates several cube maps per frame with a single pipeline pass.

4. Results and Discussion

We compared EPR's cube-mapped reflections to VIR's MVR's using OpenGL 4.5 on a PC with an Intel i5-7600K @ 3.80 GHz

#Reflec Objects *# tris*	EPR #points (pt gen in ms)	EPR total (ms) (× Faster)	VIR #points (pt gen in ms)	VIR (ms)	MVR (ms)
1 *1.57 M*	3.08*M* (1.50)	3.01 (×0.91)	6.50*M* (2.56)	4.05	2.74
5 *1.78 M*	2.79*M* (1.88)	9.82 (×1.26)	7.24*M* (3.24)	15.06	12.38
10 *1.93M*	2.45*M* (2.48)	20.41 (×1.35)	7.14*M* (3.83)	29.34	27.45
20 *2.39M*	2.49*M* (4.83)	50.83 (×1.24)	7.64*M* (5.79)	82.04	63.05

Table 1: Speed comparison of EPR, VIR and MVR for a breakfast scene with 1.5M non-reflective triangles and 1-20 reflective objects. As the number of reflective objects varies, we report total time, along with point cloud size and generation time for EPR and VIR.

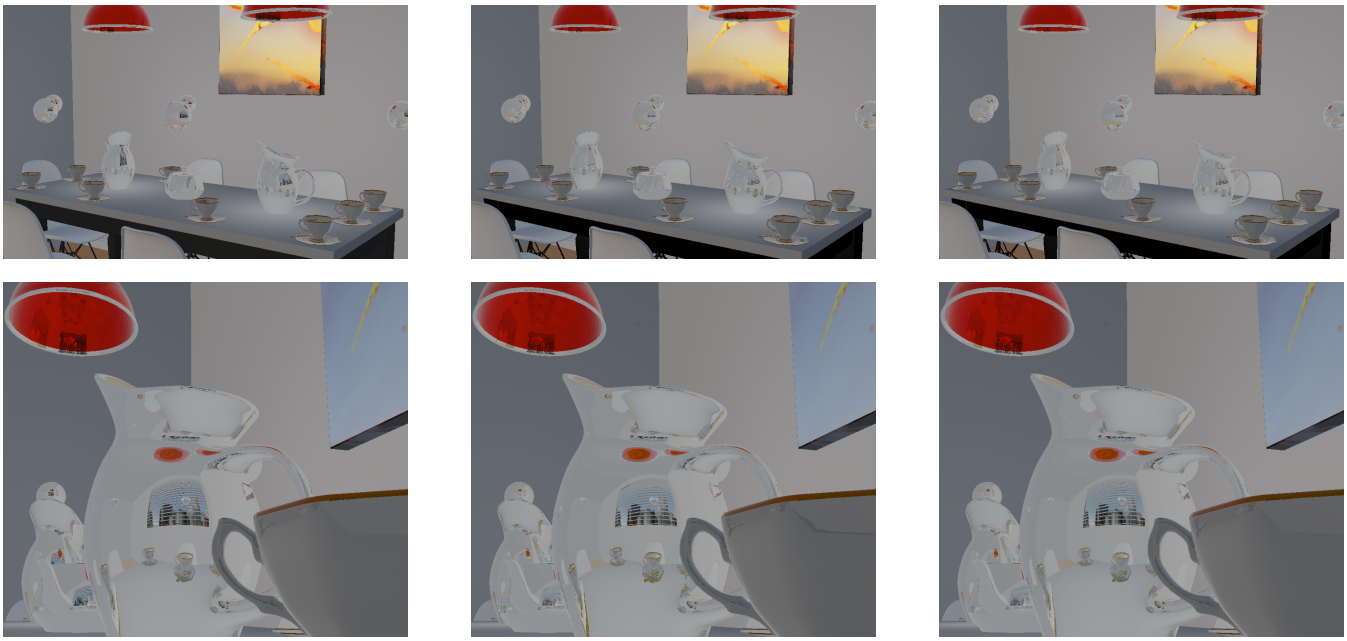

Figure 1: Cube mapped reflections generated using MVR (left), EPR (middle) and VIR (right); for distant (top) and close-up (bottom) views. All use 20 cube maps. The images in the top row were generated with average frame times of *63 ms* (MVR), *51 ms* (EPR) and *82 ms* (VIR).

CPU and an NVIDIA 1080Ti GPU, running Windows 10 OS. Scenes were dynamic, with each reflective object rotating around itself twice (720 degs), while the eye also revolved around the scene twice. Lights remained stationary. Our cube maps used 32-bit unsigned integer buffers, with a maximum adaptive resolution of 512^2. To compare performance, we averaged GPU run-time and the number of points generated over 1256 frames of execution, with each technique generating the same views at different rates.

Table 1 shows results for a breakfast room scene with varying numbers of reflective objects [McG17]. The leftmost column shows the number of triangles per scene, and the number of reflective (cube-mapped) objects in it (non-reflective objects required $1.5M$ triangles). The next two columns show EPR's point cloud size along with the time required to generate that point cloud, and the total time to generate point clouds and make cube maps. For comparison, the next two columns report VIR's point cloud size and generation time, while the last column reports MVR's total time. Under EPR's total time we report its performance improvement as the ratio of MVR time over EPR time, highlighted in blue and red. EPR renders these dynamic, complex cube maps up to 1.35 times faster than MVR, and up to 1.64 times faster than VIR. Figure 1 shows cube mapped reflections generated by MVR, EPR, and VIR. Despite being faster than MVR, EPR's reflections have similar visual quality, and remain stable during animation.

EPR generated its point cloud roughly 1ms more quickly than VIR, and its point cloud was nearly three times smaller. Despite this, EPR was only moderately faster than VIR. This is likely due to the greater fragment shader loads required for EPR to splat points across many buffer pixels, rather than just one pixel in VIR.

5. Conclusions and Future Work

EPR generates sparser point clouds than VIR, while remaining watertight. We show that it can generate high-quality environment maps, at speeds slightly faster than a standard MVR cube map.

We are currently implementing several optimizations that will improve EPR's speed advantage by reducing point cloud size further, and particularly by reducing processing per point. For example, we might use a less conservative bound for the distance of a triangle from the eye, or create a point cloud hierarchy as described in [HREB11]. To reduce fragment shader load, we are experimenting with image hierarchies and depth culling. We will also explore alternative mappings to the pipeline that make use of feedback paths to enable hardware-supported vertex splatting.

In future work, we will explore the use of EPR with other multi-view effects such as soft shadows, defocus blur, or motion blur. Finally, we plan to examine the use of EPR with light field displays, which demand tens or hundreds of views in every frame.

References

[BN76] BLINN J. F., NEWELL M. E.: Texture and reflection in computer generated images. *Commun. ACM 19*, 10 (1976), 542–547.

[Gre86] GREENE N.: Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications 6*, 11 (1986), 21–29.

[HREB11] HOLLANDER M., RITSCHEL T., EISEMANN E., BOUBEKEUR T.: Manylods: Parallel many-view level-of-detail selection for real-time global illumination. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 1233–1240.

[McG17] MCGUIRE M.: Computer graphics archive, July 2017. URL: `https://casual-effects.com/data`.

[MWH17] MARRS A., WATSON B., HEALEY C. G.: Real-time view independent rasterization for multi-view rendering. In *Eurographics (Short Papers)* (2017), pp. 17–20.