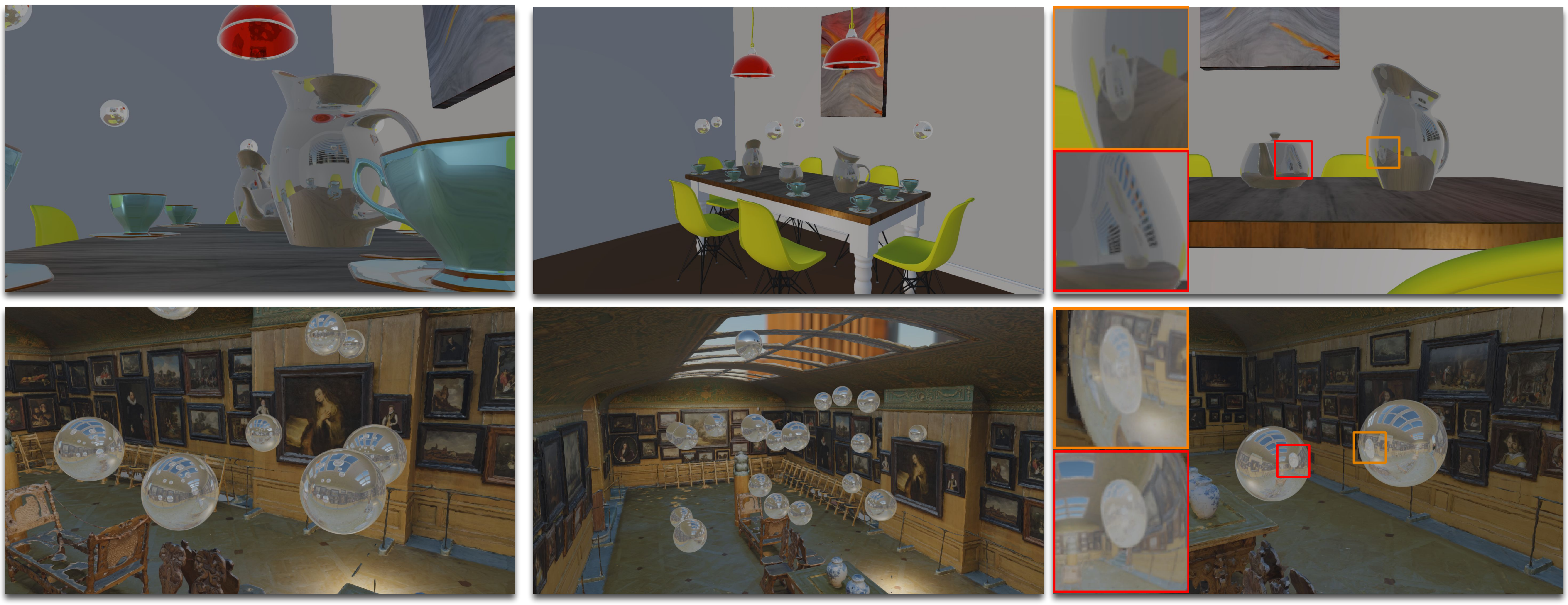
Eye-Based Point Rendering for Dynamic Multiview Effects

Ajinkya Gavane & Benjamin Watson

Email: asgavane@ncsu.edu, bwatson@ncsu.edu



Cube map reflections generated by eye-based point rendering (EPR) for dining (top row) and gallery (bottom row) scenes with camera close (left) and far (middle). Last column demonstrate reflections of reflections with EPR.

Introduction

Eye-based point rendering (EPR) is a new algorithm that combines the geometric parallelism of view independent rendering (VIR) with new eye-resolution shading efficiencies throughout the pipeline to make real-time multiview effects more practical. We demonstrate this by applying EPR to dynamic cube-mapped reflections and omnidirectional soft shadows. Standard Multi-view Rendering (MVR) renders a scene six times for each cube map. VIR improves matters, but cannot reach real-time speeds when views are diverse and shading complex. EPR introduces eye-resolution constraints of shader computation: in each frame, it limits points, off-screen buffer sizes, lighting and reflection calculations to the sampling rate of the eye's viewport.

Objective

Show that EPR makes multiview effects like dynamic cube-mapped reflections practical. It can render a complex scene with over 2 million triangles and 20 moving, reflective objects in real time.

Method

- EPR guarantees that at least one point appears in every pixel in the viewport (the eye's view).
- The geometry shader conservatively finds a triangle's needed sampling density using the distance to the eye via any reflective object, and setting up rasterization to match.
- Each point has location, triangle vertices, normal, material, size, and view visibility string. The fragment shader puts these points in multiple storage buffers for later splatting by the compute shader.
- The compute shader, for each point and view pairing, splats the point into that view's off-screen buffer by writing only the point's reference.
  - The splat defined can be slightly larger than an eye pixel, so we clip each splat against the edges of its triangle and fill it only if it lies inside the triangle.
- The lighting for each off-screen buffer is deferred and resolved only when lighting for a pixel in the eye's view is being computed. We call this recursively deferred rendering **doubly deferred**

Results

Scene	#Refl objs (#tris)	EPR: point gen + view gen (#points)	EPR: point gen + view gen + sampled deferred	VIR tot time (#points)	MVR tot time	× times
Dining Scene	1 (1.57M)	4.71 (1.35M)	5.82	7.31 (6.47M)	4.05	0.70
	8 (1.92M)	10.15 (2.02M)	12.21	54.18 (17.64M)	21.90	1.79
	16 (1.92M)	14.99 (2.27M)	17.31	124.17 (26.98M)	48.39	2.80
	20 (2.39M)	18.04 (2.37M)	20.60	138.71 (28.74M)	61.45	2.98
Gallery Scene	1 (1.01M)	9.36 (1.95 M)	10.32	6.85 (5.29M)	4.11	0.40
	8 (1.09M)	12.62 (2.49M)	14.80	21.55 (8.51M)	29.45	1.99
	16 (1.19M)	16.92 (2.62M)	19.95	41.69 (11.04M)	59.52	2.98
	20 (1.24M)	19.48 (2.69M)	23.06	53.20 (11.91M)	75.05	3.26

Comparing speed of EPR vs. VIR and MVR, for the dining and gallery scenes with 1-20 reflective objects and 1.5M and 1.0M non-reflective triangles, respectively. We report total time, along with point cloud size and generation time for EPR on NVIDIA 1080Ti GPU

Conclusion and Discussion

- EPR generates sparser point clouds than VIR.
- EPR generates high-quality environment maps, 3.3x faster than MVR, and 6.7x faster than VIR.
- EPR preserves more details in the reflections while MVR blurs them.
- EPR enables drastic speedups of recursive reflection without any additional passes or buffers.
- We plan to follow up this work by:
  - introduce adaptive sampling that depends on the solid angle subtended by the reflection vectors within the eye pixel when computing environment map reflection in the eye's deferred shading pass.
  - use of EPR with other multi-view effects such as defocus and motion blur. We also plan to examine the use of EPR for foveation and with light field displays

REFERENCES

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

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

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

HOLLANDER M.,et al: Manylods: Parallel many-view level-of-detail selection for real-time global illumination. In Computer Graphics Forum(2011), 30, pp. 1233–1240.

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