

Gaussian Splatting

- Trains a 3D scene representation from images and camera positions
- Uses blobs known as Gaussians with opacity and view-dependent colors, encoded as spherical harmonics
- Photorealistic results, but “cheats” with geometry and lighting
- Scene editing is more practical
- Realtime rendering

Problem for Reflections

- Appearance is learned but not scene lighting, material properties or geometry.
- Reflections are baked-in
- Spherical harmonics are not suited for modelling reflective surfaces
- Scene edits lead to wrong reflections

Method

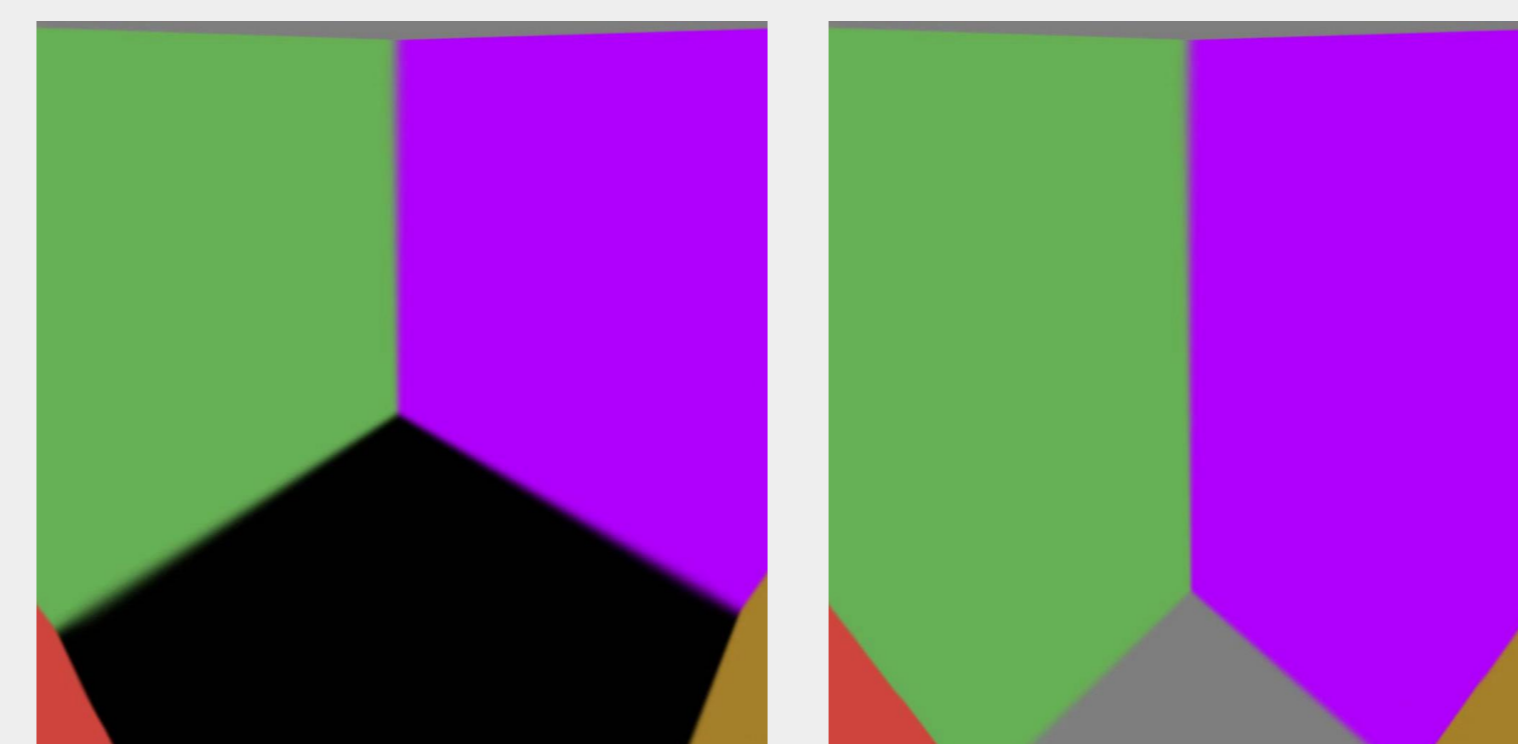
- Use techniques from conventional 3D rendering to calculate reflections and apply them to Gaussian representations.
- Planar reflection for flat mirrors, and cubemaps for non-flat mirrors.
- Bounding box to specify Gaussians as mirrors
- Mask to indicate mirror pixels.
- Composite the reflection image together with the rendered image.

Implementation

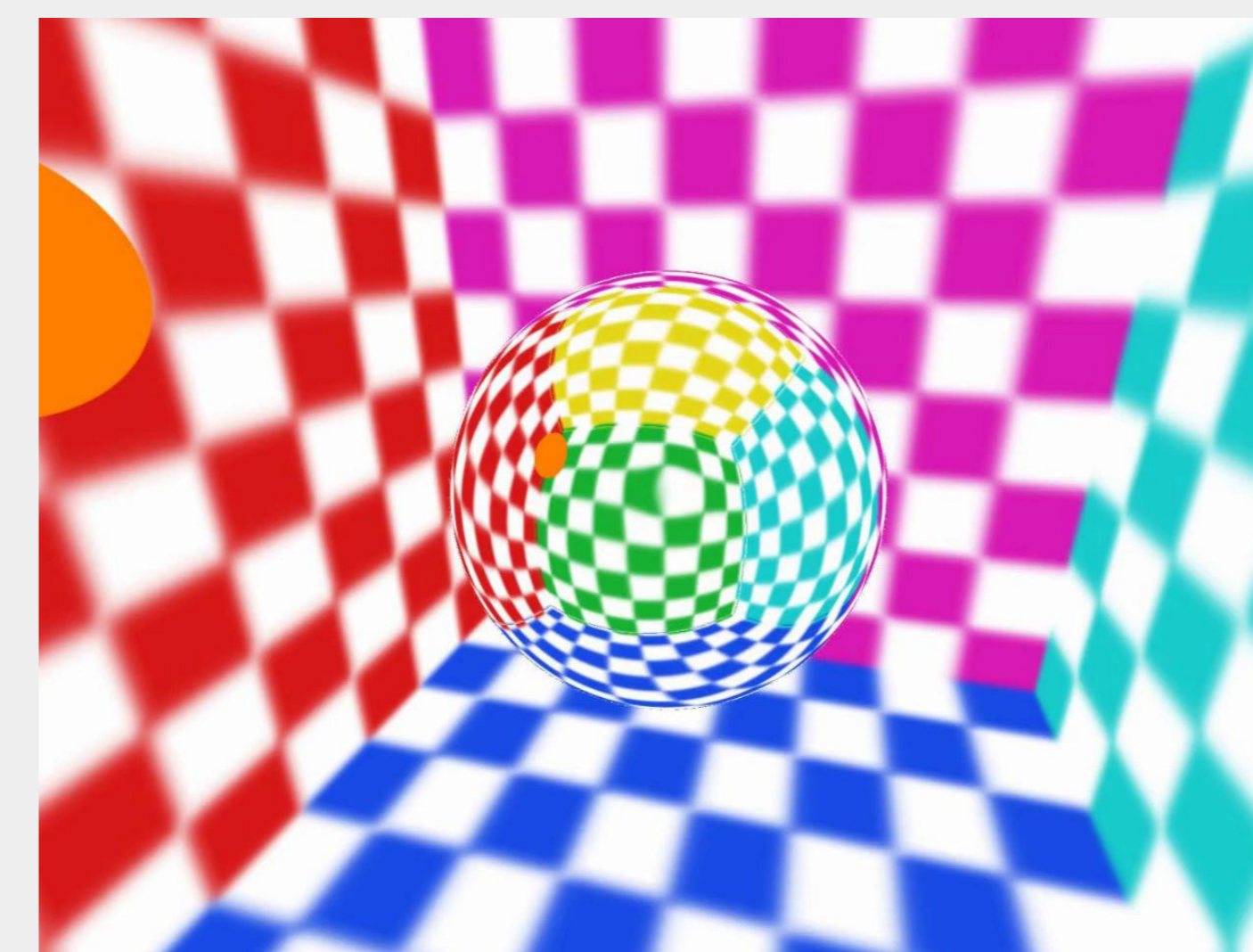
- Key observation: With more training iterations, Gaussians become flatter
- Thin axis can act as an approximation for a surface normal
- Planar reflection: Shared normal for all Gaussians.
- Calculate mirror normal from PCA on per-Gaussian normals.
- Reflect incident ray around mirror normal for reflection viewing angle.
- Position new camera behind mirror, cull all Gaussians between mirror camera and render reflected image.
- Cubemaps: Pre-render images in six fixed direction and cache.
- Sample from cached cubemaps to provide the reflection

Synthetic Scenes

- Testing planar reflection
- Room with colored walls.
- Mirror floor



- Testing cubemap reflection
- Checkerboard walls
- Off-center floating orange sphere



Results

- Planar reflection
- Room scene
- Turn part of wall into a mirror
- Reflection is visible and accurate.
- Floaters occluding the mirror slightly.
- Jagged edges because of Gaussian shapes.
- Cubemap reflection
- Turn an object into a mirror
- Inconsistent result caused by the normals of the surface being inaccurate.

