Dynamic Sky Dome

KGC November, 2011

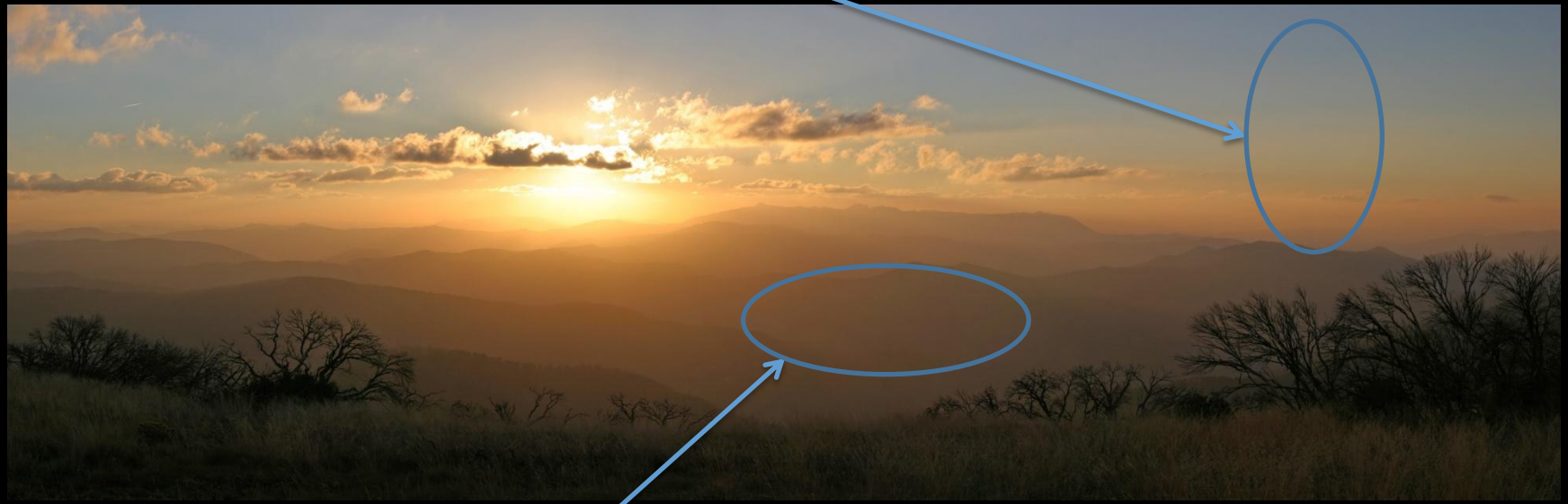
by Igor Lobanchikov

Agenda

- Dynamic Sky Dome lighting model
  - Introducing the rendering equation
  - Fog integration
  - Sun shafts
- Dynamic clouds rendering
- Bonus trick

Effects to Achieve

- Rayley scattering
- High chromatic dependency



- Mie scattering
- Achromatic

© David Iliff

CONFETTI

Sky Dome Rendering History

- [PSS99][HP03][O'N05]
- Single scattering only
- Or viewable from ground only
- Or all of the above
- Or not real-time

Current Approach

- Multiple scattering
- Aerial perspective
- Viewable from space
- LUTs are pre-computed on GPU

Rendering Equation

$$L(x, v, s) = (L_0 + R[L] + S[L])(x, v, s)$$

- ***x*** – position, ***v*** – view direction, ***s*** – sun direction
- Final lighting includes:
  - L_0 – attenuated direct sun light
  - $R[L]$ – light reflected at point (x_0) and attenuated
  - $S[L]$ – light scattered toward the viewer between x and x_0
- Usually computed by numerical integration

CONFETTI

Direct Sun Light

$$L_0(x, v, s) = T(x, x_0) L_{sun}, \text{ or } 0$$

- Direct sunlight attenuated by the transmittance function $T(x, x_0)$ before reaching viewer at x
- Accounts for occlusion by the horizon
- Smooth shadow transition to hide sharp border on the clouds

Reflected Light

$$R[L](x, v, s) = T(x, x0)\, I[L](x_0, s)$$

- Light reflected at x_0 and attenuated by transmittance towards x
- I[L] == 0 at the top atmosphere boundary

Inscattered Light

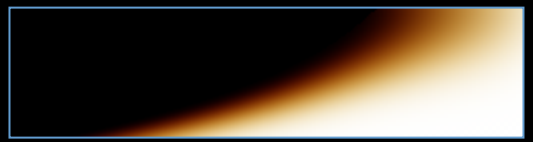
$$S[L](x,v,s)=\int_{x}^{x_0} T(x,y)J[L](y,v,s)dy$$

- Light scattered between x and x_0 toward x

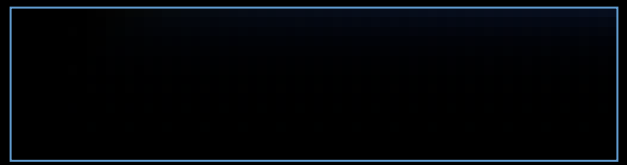
Look-up Tables (LUT)

- Re-parametrise functions and store them in LUT
- Pre-compute LUTs on GPU

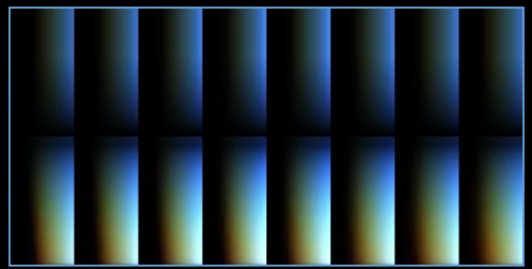
Transmittance (r, μ)



Irradiance (r, μS)

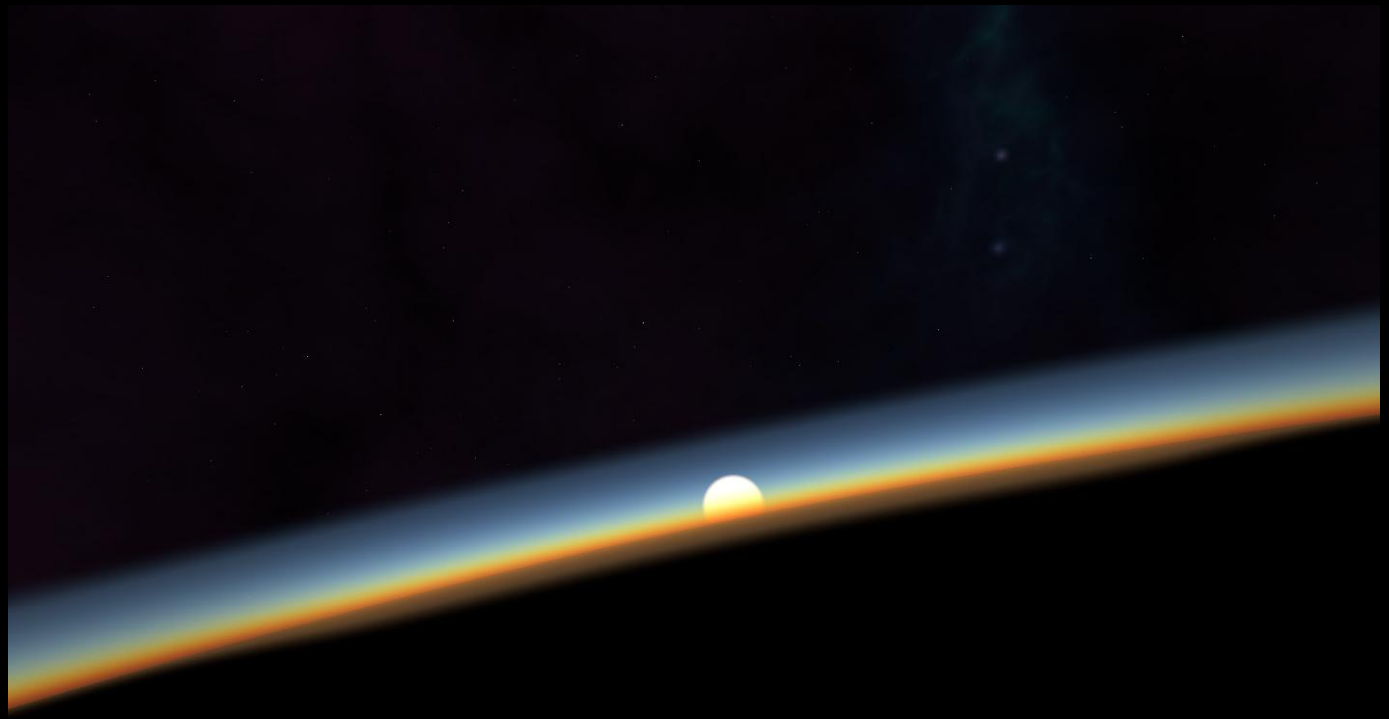


Inscatter (r, μ, μS, ν)



CONFETTI

View from Space



CONFETTI

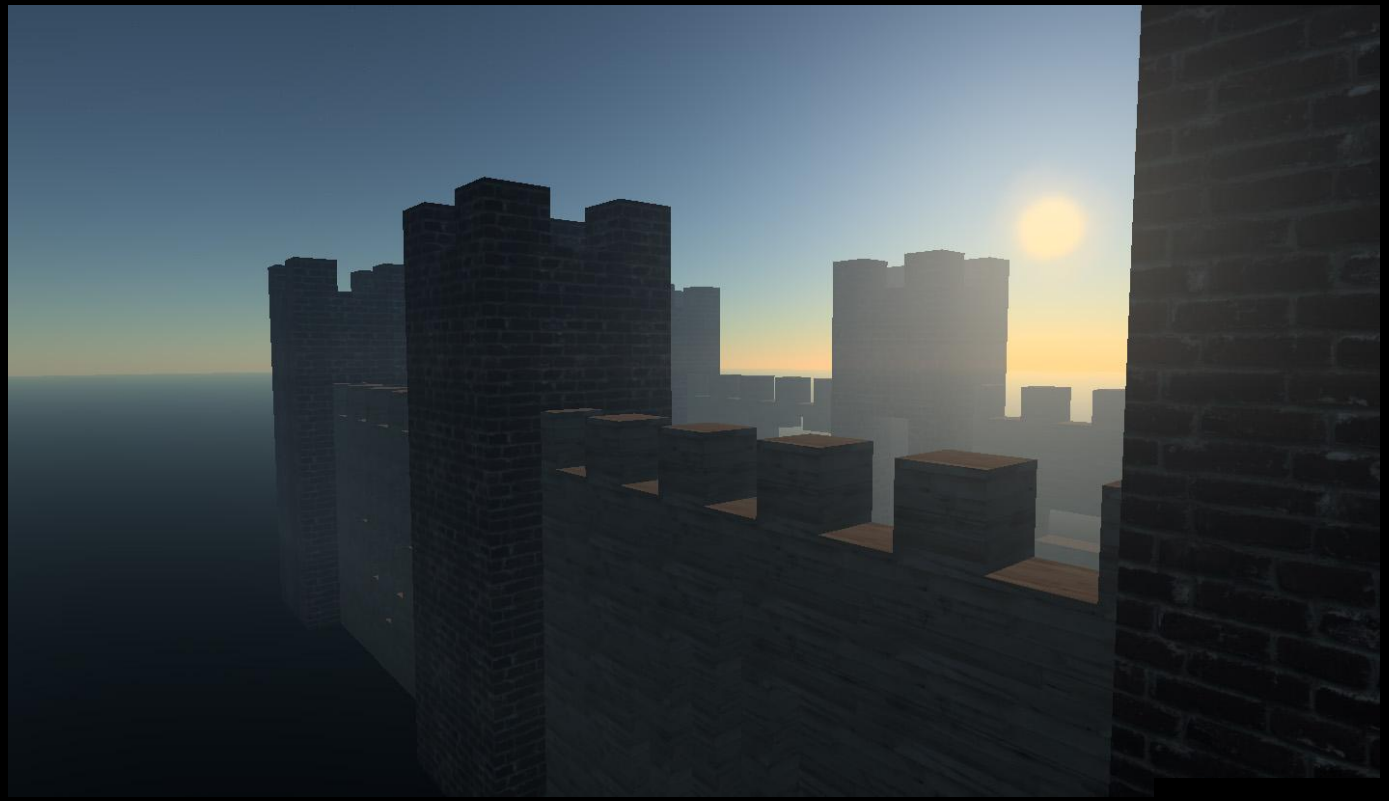
Simple Fog vs Inscatter

- Fast
- Change parameters in real time
- Accounts for transmittance only
- Looks fine

- Slower
- Precompute
- Accounts for transmittance and inscatter
- Extremely realistic

CONFETTI

Inscatter

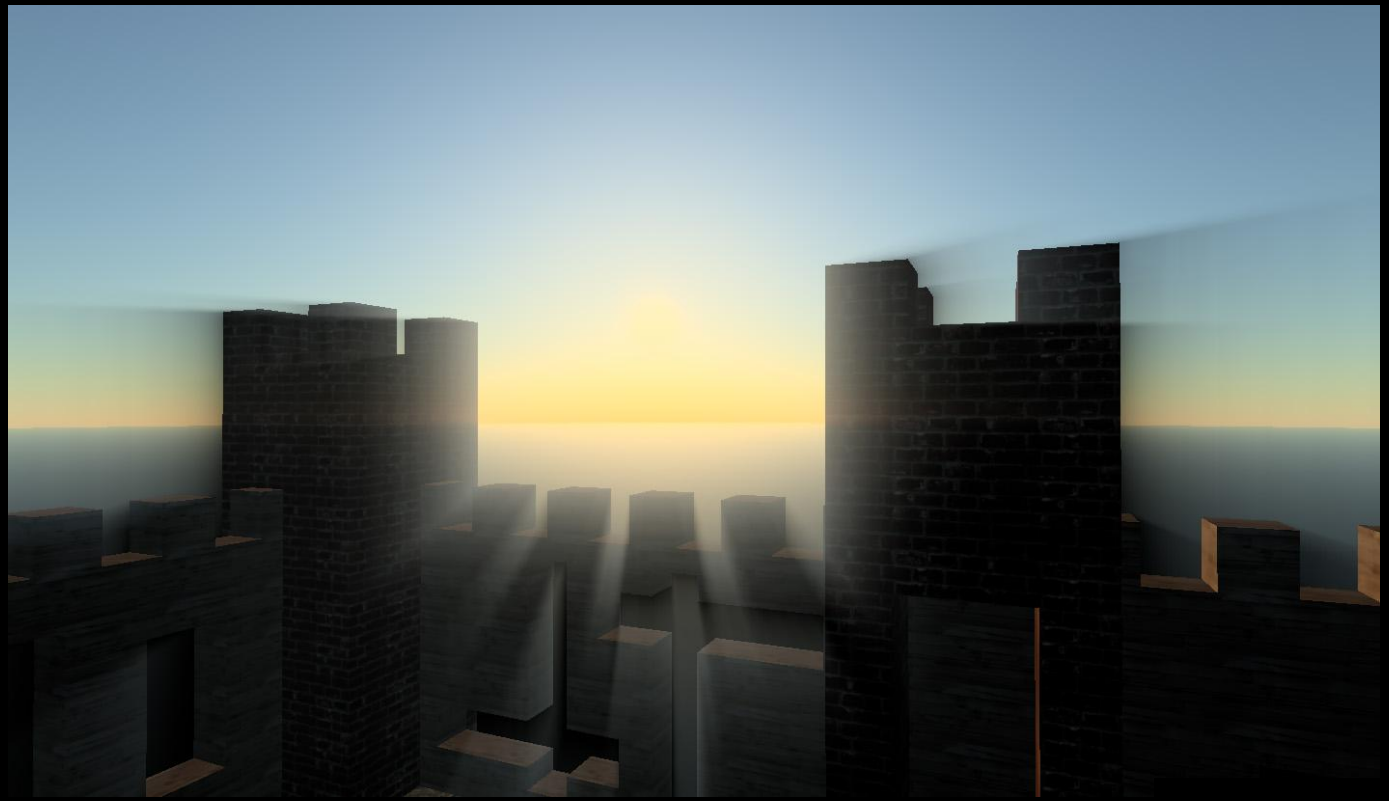


CONFETTI

Sun shafts

- Use industry-standard mask generation
- Radial blur it
- Modulate inscattered light with the mask
- Cheap, known, works for all types of geometry

Sun shafts



CONFETTI

Clouds

- Need volumetric realistically looking clouds
- Use group of particles to represent a single cloud
- Use the same lighting equation to blend nicely into the sky
- Use impostors to save on expensive per-pixel lighting computations

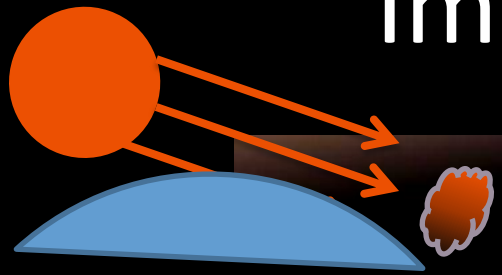
Clouds lighting

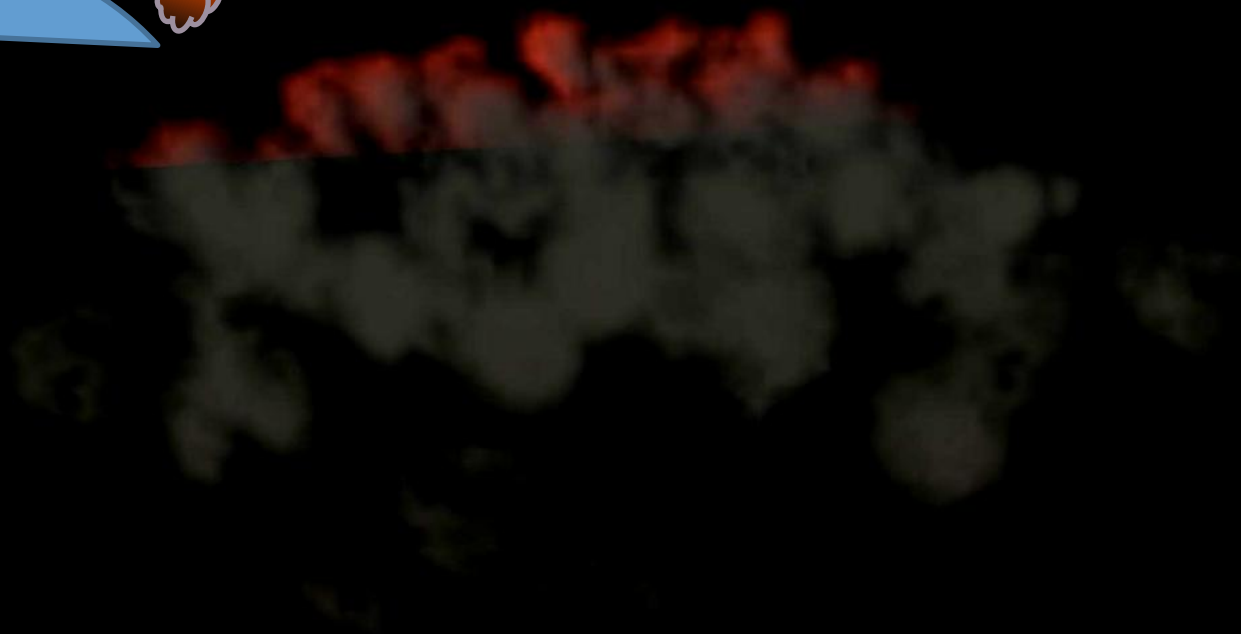
- Need x_0 to apply lighting equation
  - Use world-space impostor's pixel position
- Apply inscattered light and irradiance as to a regular object.
- To simulate sun direct light scattering use similar to [WENZEL07] technique
  - March in sun direction in impostor space to accumulate light extinction factor to attenuate sun light
  - Add rim light to simulate scattering on the cloud's edge

Clouds lighting

CONFETTI

Impostor Pixel Position





CONFETTI

Impostor Pixel Position

- Need better pixel position reconstruction:
  - Sun shadow is a crisp horizontal line
  - Can’t pack a group of clouds into a single impostor or have a very large cloud

Impostor Pixel Position

- Accumulate impostor-space linear depth per pixel
  - Simple min doesn’t work well – must account for transparency, linear interpolation problem on borders
  - Simple additive blend doesn’t work – depth on cloud edges blend with impostor clear depth value

Impostor Pixel Position

Min blend

Additive blend

CONFETTI

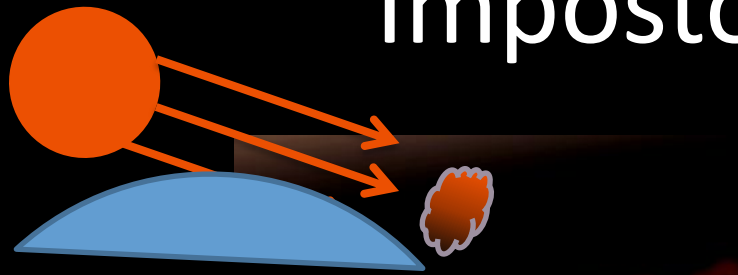
Impostor Pixel Position

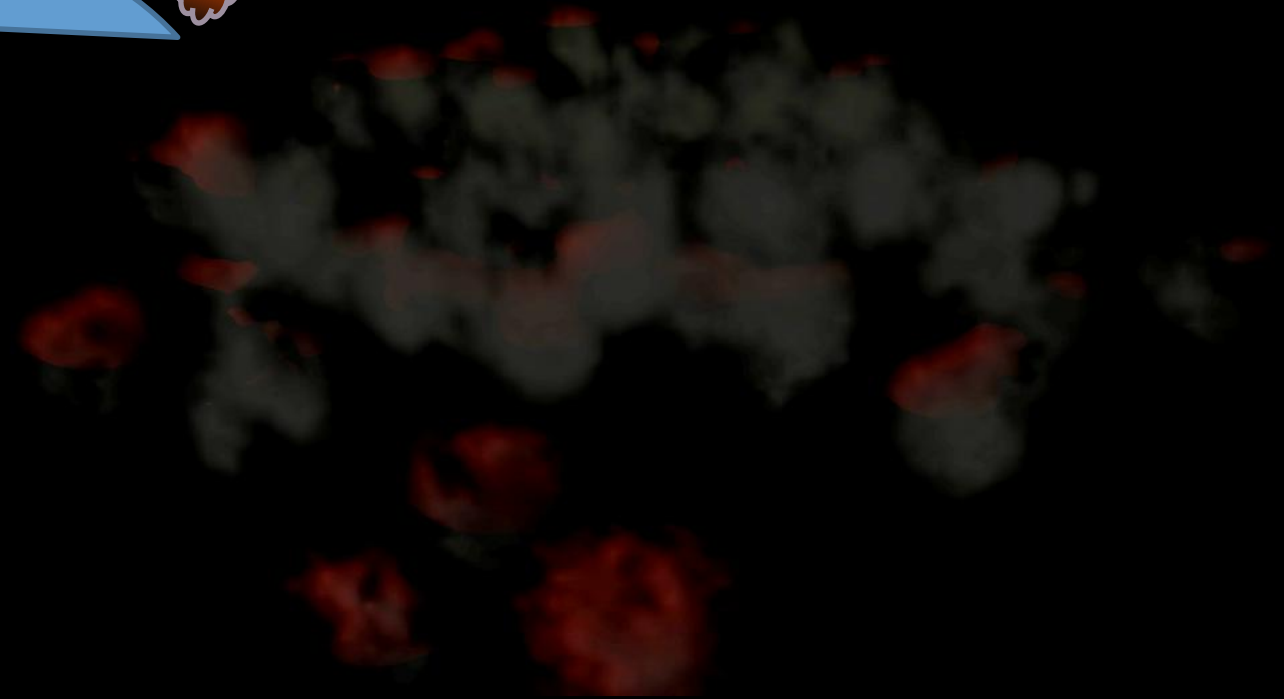
- Store both accumulated depth and normalization weight
  - DepthAcc = OldDepthAcc*(1-newDepthWeight) + NewDepth*NewDepthWeight
  - DepthWeightAcc=OldDepthWeightAcc*(1-newDepthWeight) + NewDepthWeight
  - Final depth = DepthAcc/DepthWeightAcc
- Use particle transparency as DepthWeight => Need 2 FP16 channels to store both impostor depth and density
- Linear interpolation is not linear anymore, but it is monotonous and works
- Have two more spare channels – store normal or per-particle ambient occlusion for more sophisticated lighting?

Impostor Pixel Position

CONFETTI

Impostor Pixel Position





CONFETTI

Polarization

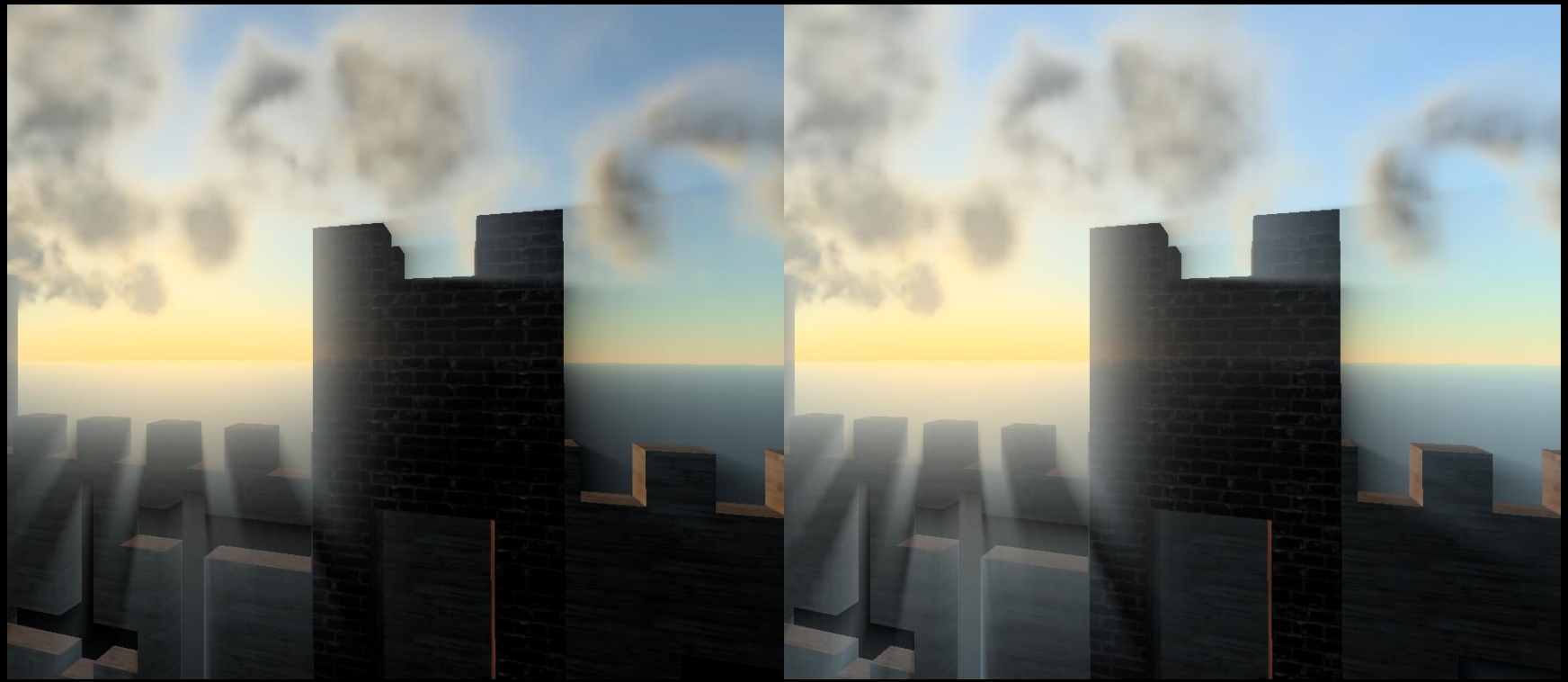
- Sky light is polarized
  - Max polarization at view-sun angle == 90^0
  - Min polarization at view-sun angle == 0^0 and 180^0
- Simulate effect of camera with polarized filter attached to add more contrast and depth to the scene

CONFETTI

Polarization

ON

OFF



CONFETTI

Questions ?

CONFETTI

Reference

- [PSS99]Preetham, A. J., Shirley, P., and Smits., B. E.: A Practical Analytic Model for Daylight. In Siggraph 1999, Computer Graphics Proceedings (Los Angeles, 1999), Rockwood A., (Ed.), Addison Wesley Longman, pp. 91–100.
- [HP03] Hoffman, N., and Preetham, A. J.: Real-time Light-Atmosphere Interactions for outdoor scenes. *Graphics Programming Methods* (2003), pp. 337–352.
- [O'N05] O'NEIL S.: Accurate atmospheric scattering. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (2005), Addison-Wesley* Professional.
- [BrunetonNeyret2008] Bruneton, E. and Neyret, F. 2008. Precomputed Atmospheric Scattering. EGSR 2008. Computer Graphics Forum, 27(4), June 2008, pp. 1079-1086.
- [WENZEL07] WENZEL C. 2007. Real-time Atmospheric Effects in Games Revisited. Conference Session. GDC 2007. March 5-9, 2007, San Francisco, CA.

CONFETTI