Dynamic GI with Self-Occlusion on GPU or CPU from many lights

KGC November, 2011

by Wolfgang Engel, Igor Lobanchikov

Confetti Special Effects

- Think-Tank for game and movie related industries
- Clients: INTEL, Qualcomm, many game developers (“Engine Tuner”)
- Provides software solutions for games, movies and tools for GPU manufacturers
- http://www.conffx.com

Motivation

What we are going to show

- … is a Dynamic GI system that
  - works on static and dynamic objects like animated characters
  - does not require any pre-processed data
    -> game objects are destructible, 24 hour cycle is possible, artist iteration time == real-time
  - works on a large number of lights
    -> many point lights
- Volume calculations run on GPU or CPU
  -> load balancing if you have multi-core CPU
  -> works faster on CPU with 4+ core machines -> in that case very light on the GPU

CONFETTI

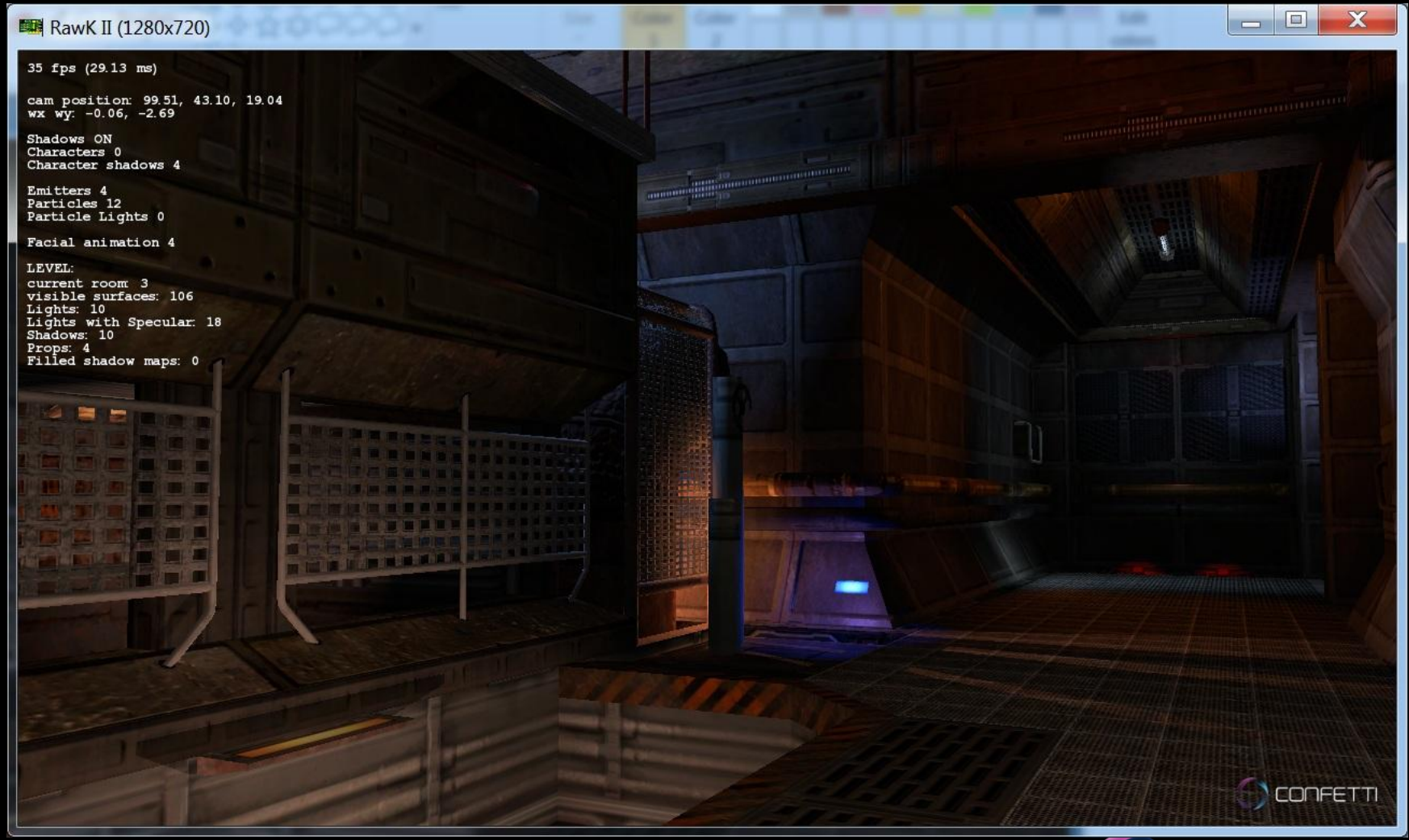
CONFETTI

CONFETTI

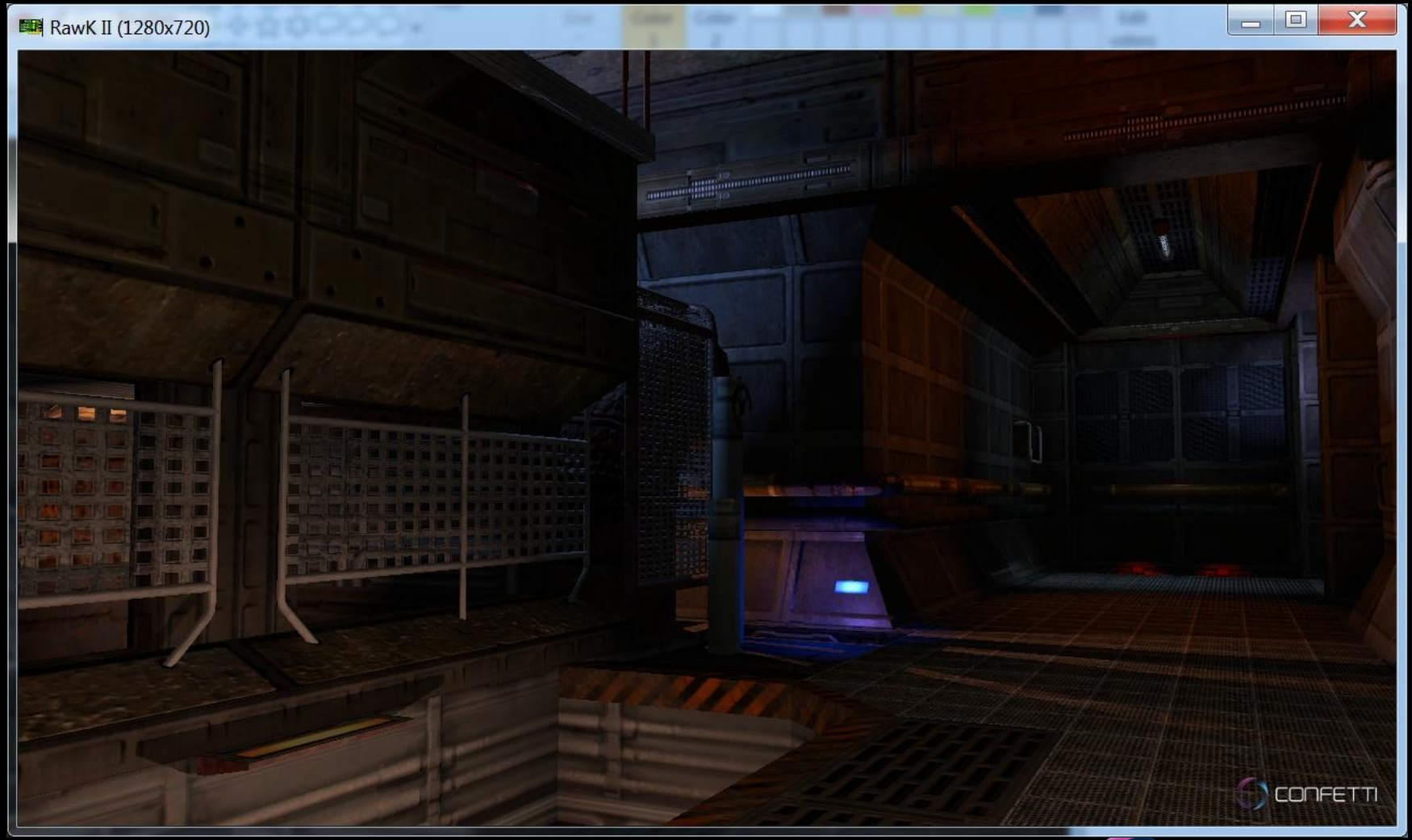

dark
dark
CONFETTI


RawK II (1280x720)
CONFETTI


RawK II (1280x720)
CONFETTI


RawK II (1280x720)
35 fps (29.13 ms)
cam position: 99.51, 43.10, 19.04
wx wy: -0.06, -2.69
Shadows ON
Characters 0
Character shadows 4
Emitters 4
Particles 12
Particle Lights 0
Facial animation 4
LEVEL:
current room 3
visible surfaces: 106
Lights: 10
Lights with Specular: 18
Shadows: 10
Props: 4
Filled shadow maps: 0
CONFETTI

CONFETTI


RawK II (1280x720)
CONFETTI
CONFETTI


RawK II (1280x720)
CONFETTI
CONFETTI


RawK II (1280x720)
CONFETTI

Agenda

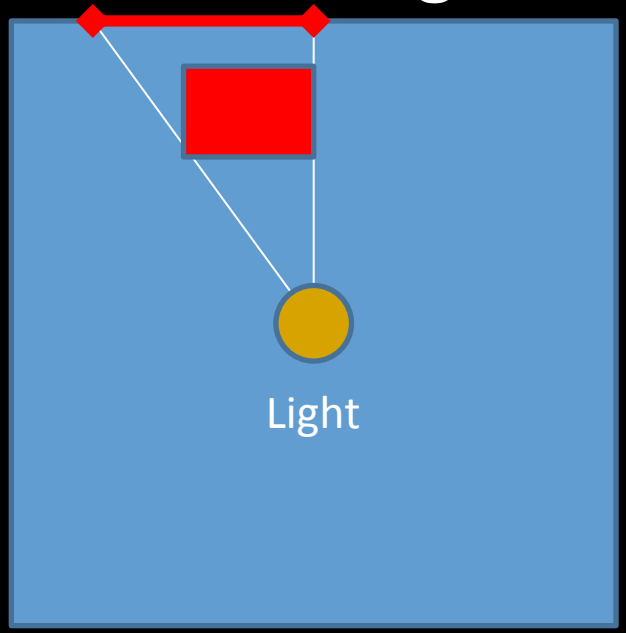
- Capturing light data in Cube Reflective Shadow Maps (CRSM)
- Light Propagation Volume (LPV)
  - Injection Phase
  - Propagation Phase
- Apply Lights

Capturing Light Data

- Cube Reflective Shadow Maps (CRSM) filled up with
  - Depth
  - Normal
  - Color data from the light color and the albedo color

Capturing Light Data

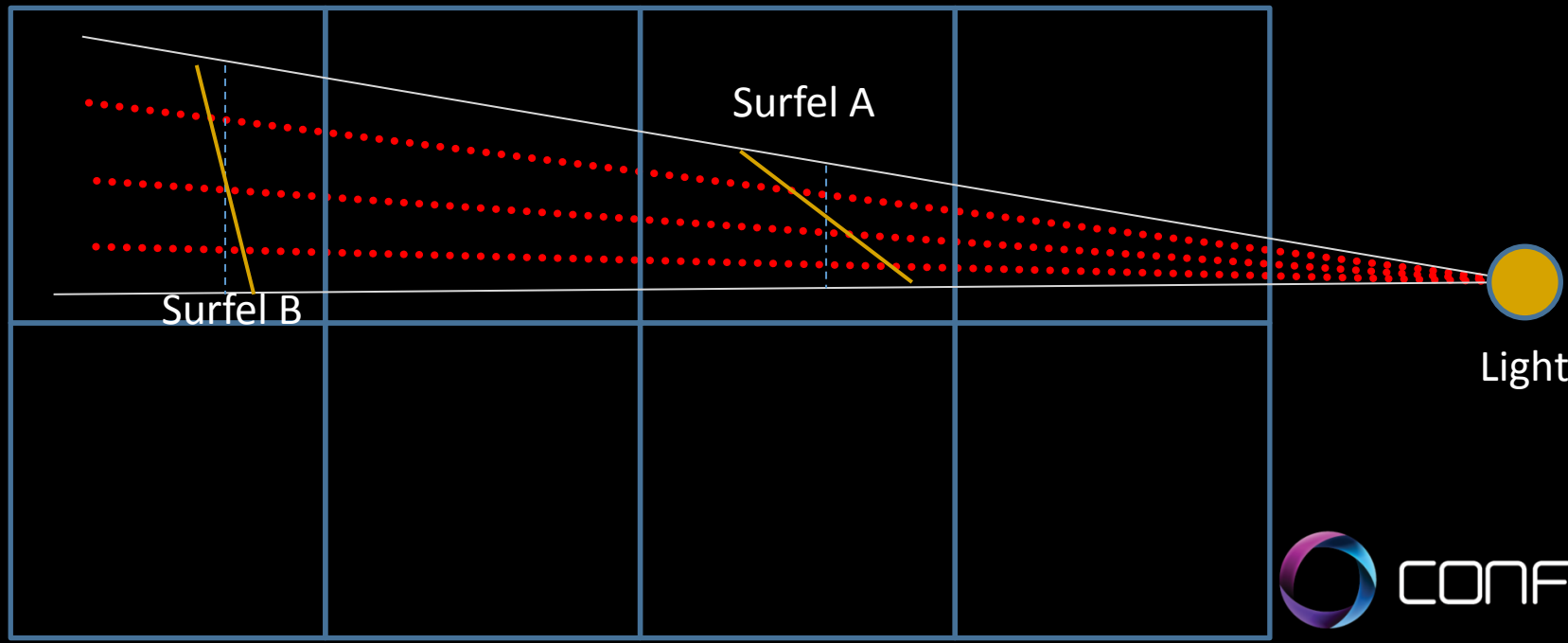
- Similar to regular RSM: capture the light surrounding into RSM [Dachsbacher]
- Projection and rendering is different



CONFETTI

Capturing Light Data

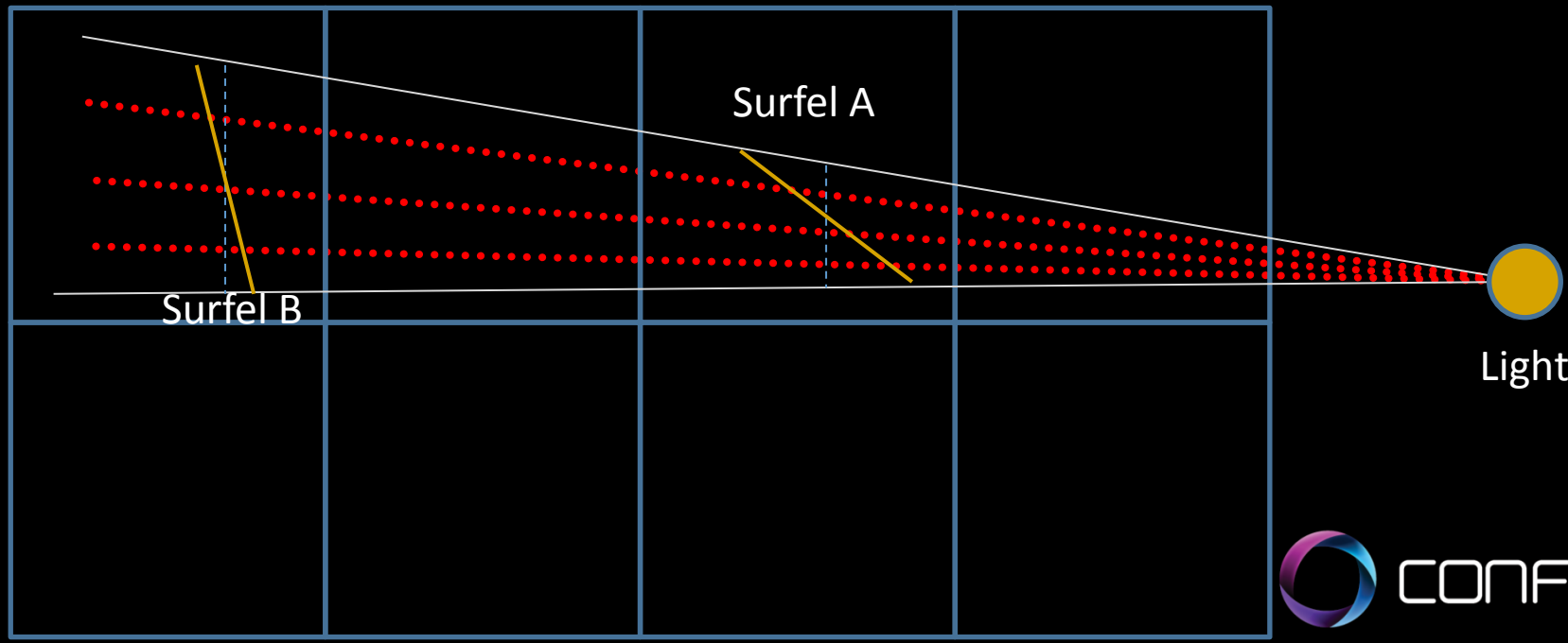
- Flux amount per surfel depends only on solid angle; not surfel orientation or distance



CONFETTI

Capturing Light Data

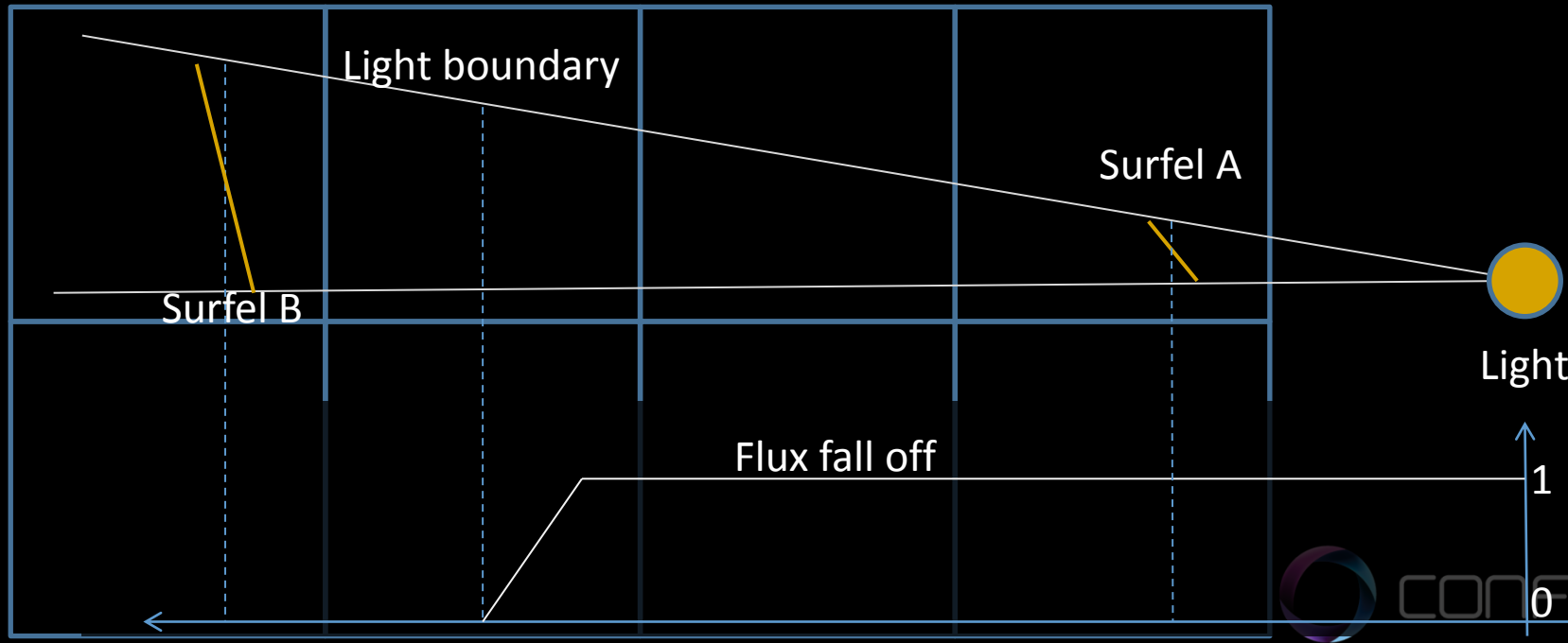
- Same amount of photons (red doted lines) will pass through surfel A compared to B for every RSM texel



CONFETTI

Capturing Light Data

- Simple flux fall-off function

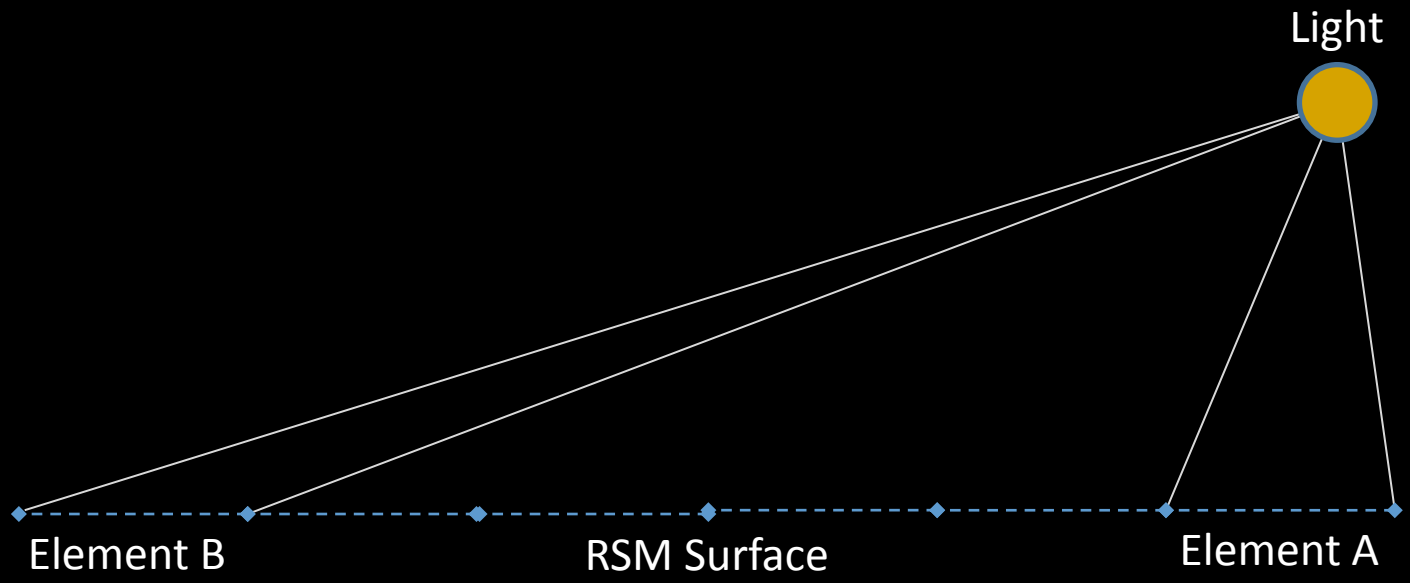


Capturing Light Data

- Simple flux fall-off function
  - Surfel with increasing distance can become bigger than the LPV cell
  - Far outside the area of influence of the light

Capturing Light Data

- Solid angle is not uniform across the RSM



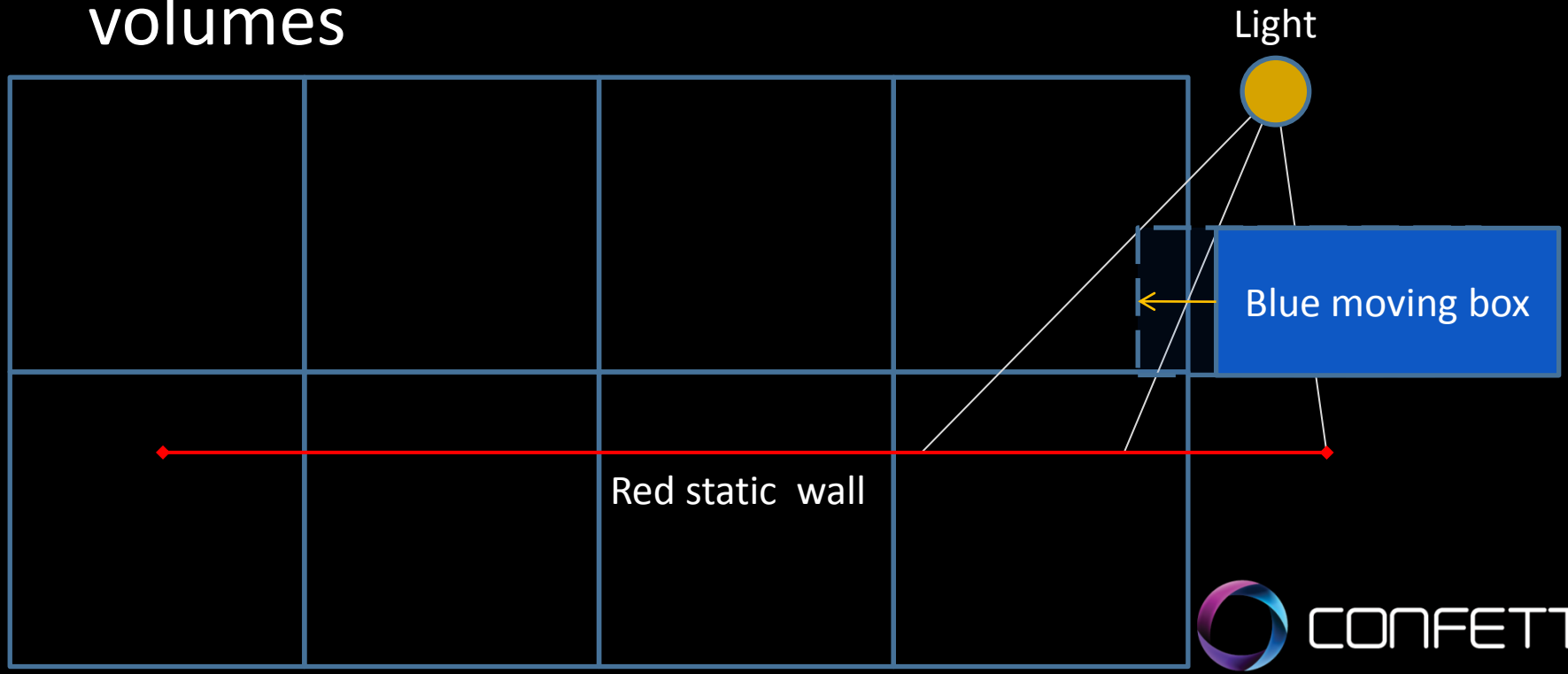
CONFETTI

Capturing Light Data

- Variable CRSM resolution
- We differ between static and dynamic CRSM
  - Static CRSM -> only receives static geometry
  - Dynamic CRSM -> also receives dynamic geometry like characters
  - The nature of the CRSM changes in case a dynamic character approaches the area of influence of the light

Capturing Light Data

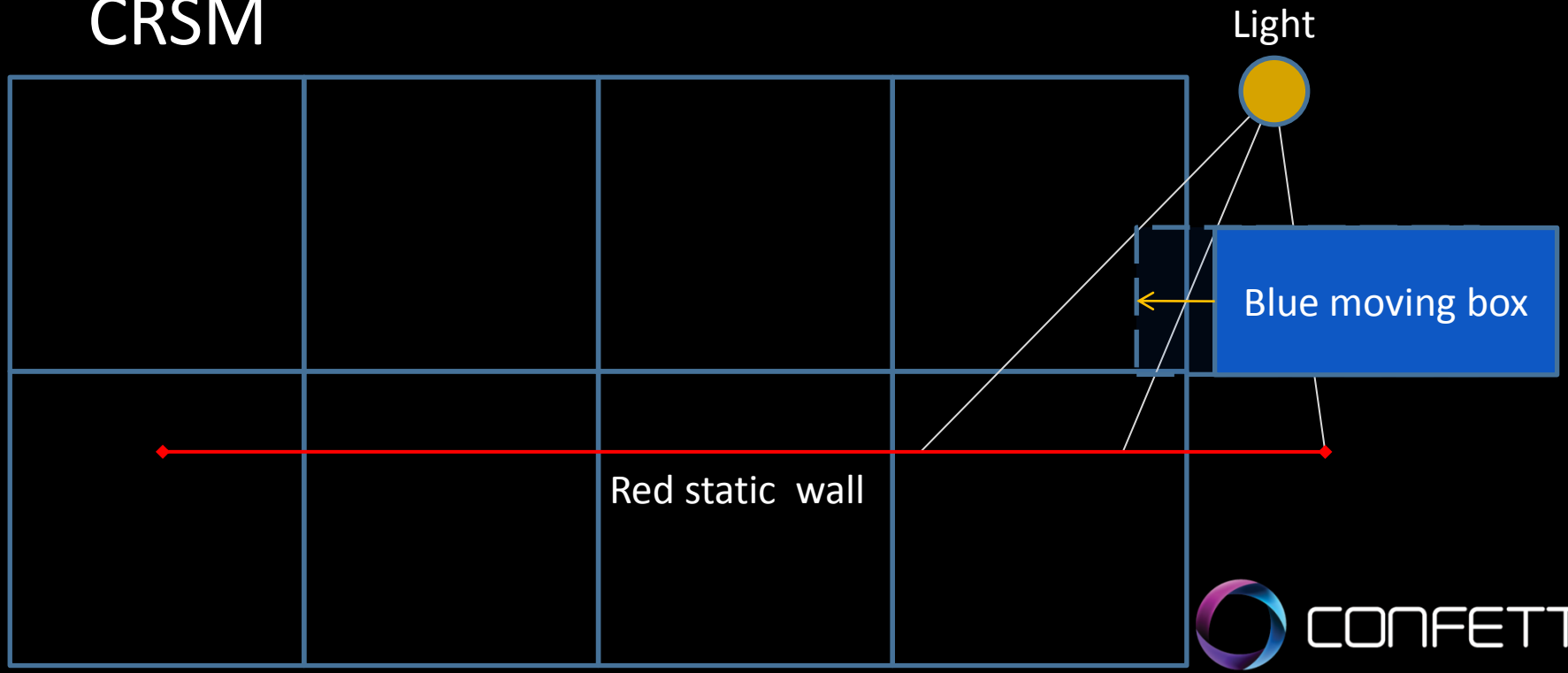
- LPV - blue cubic cells – single layer of stack of volumes



CONFETTI

Capturing Light Data

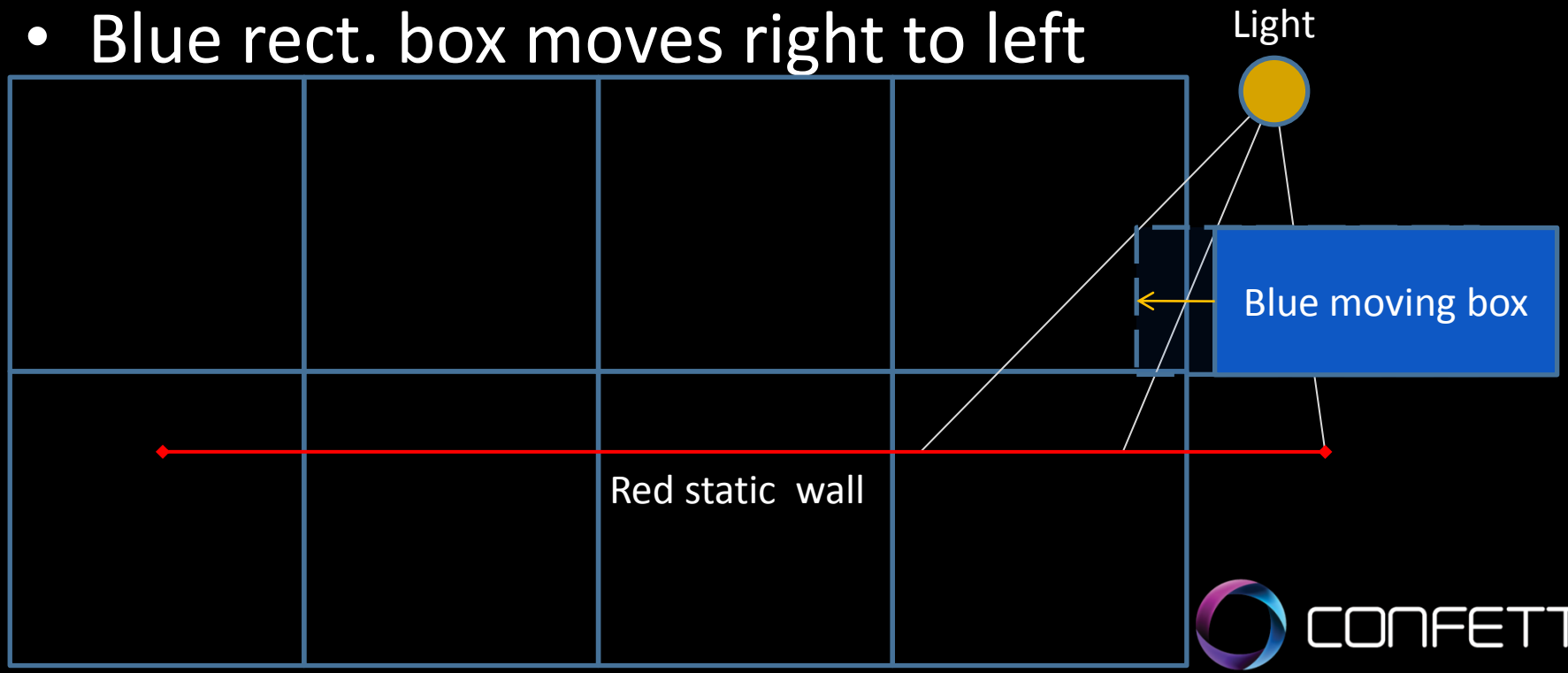
- White rays – boundaries of single pixel in CRSM



CONFETTI

Capturing Light Data

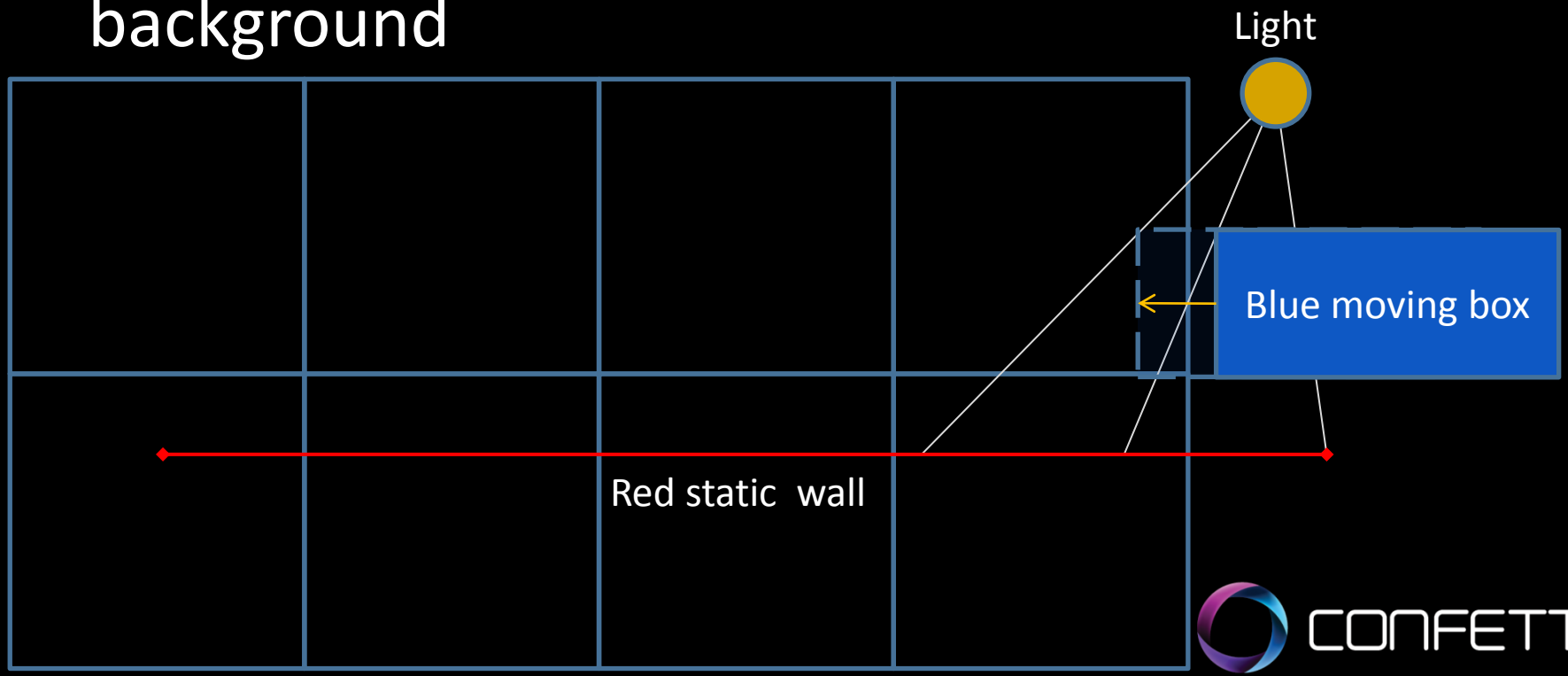
- Red line is a wall from above
- Blue rect. box moves right to left



CONFETTI

Capturing Light Data

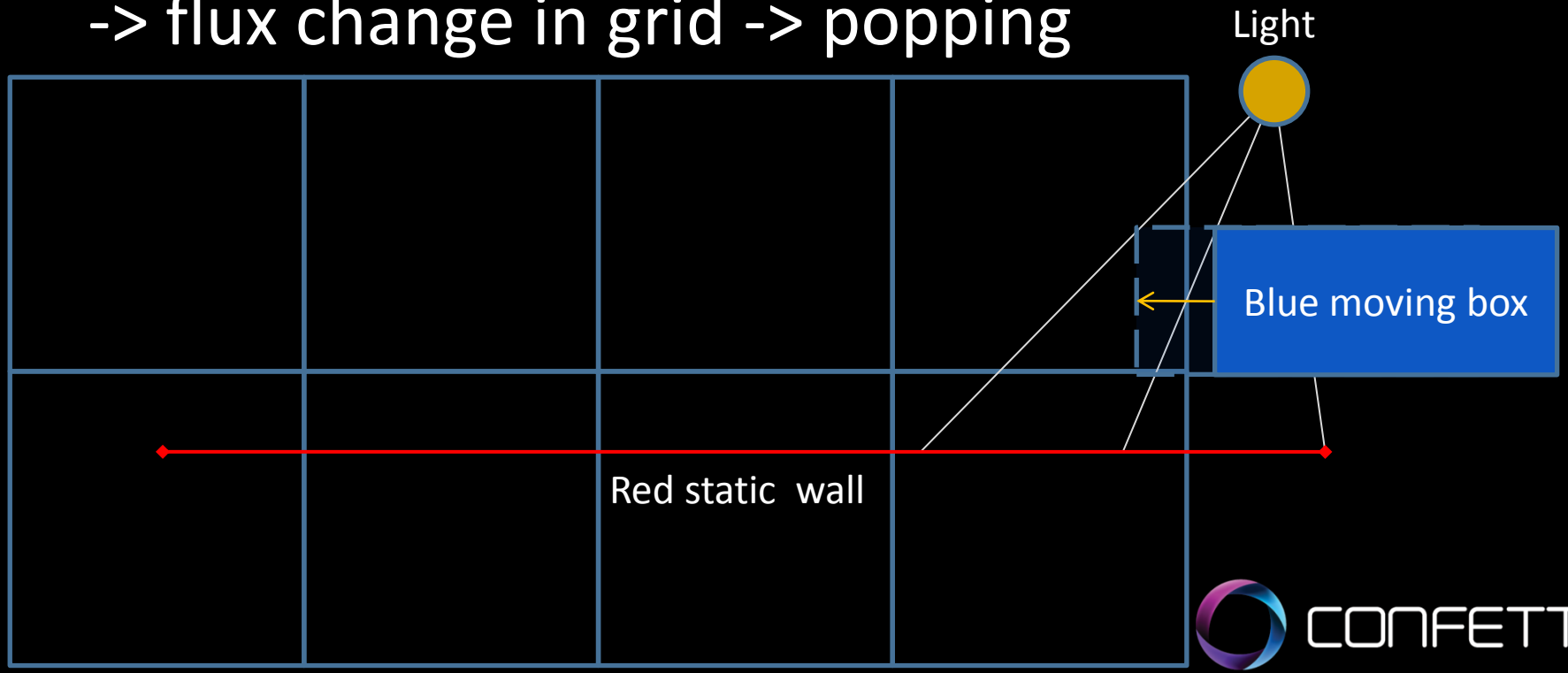
- Blue box is dynamic object -> occludes background



CONFETTI

Capturing Light Data

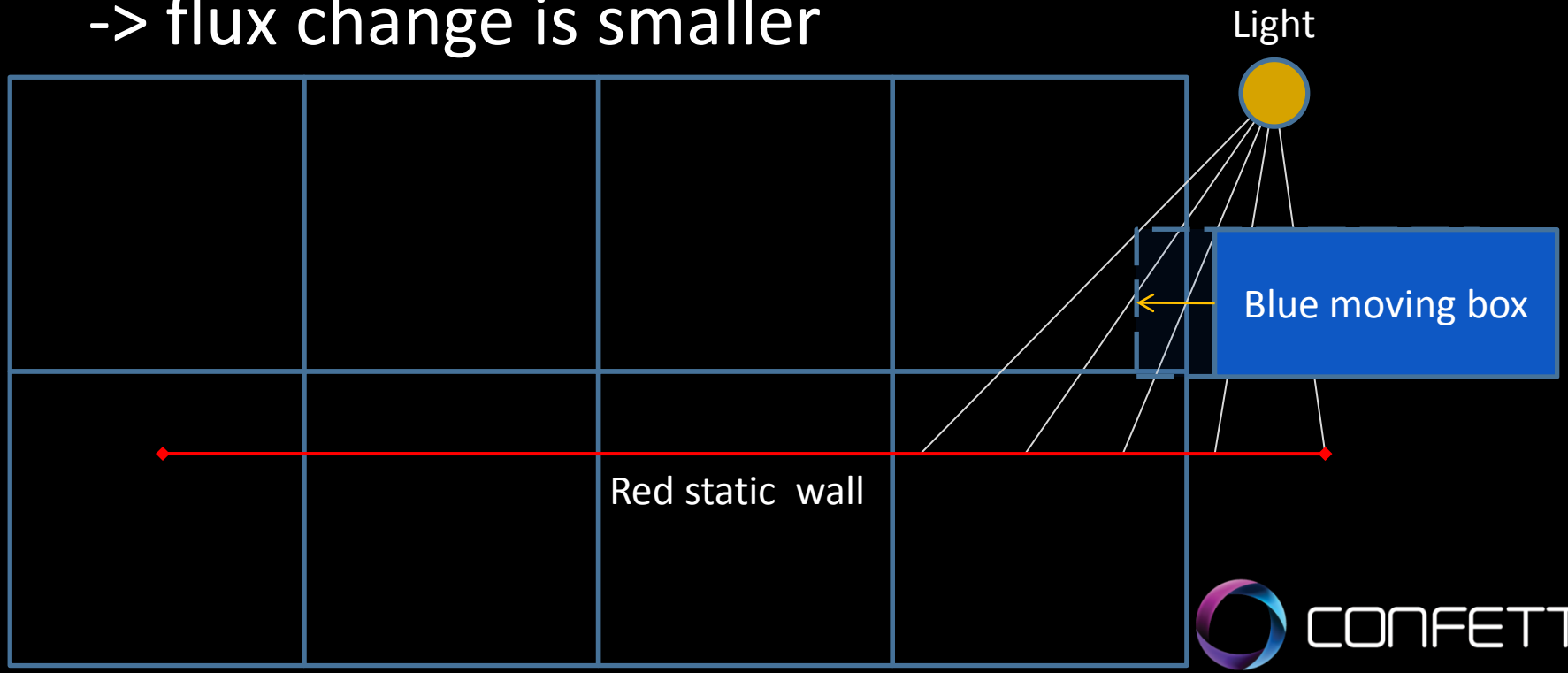
- Sufel color / position change
  -> flux change in grid -> popping



CONFETTI

Capturing Light Data

- Solution: increase RSM resolution
  -> flux change is smaller



CONFETTI

Capturing Light Data

- Variable CRSM resolution
  - Non-moving objects (static) CRSM: surfel size < LPV cell size
    - Still need to fill LPV cells continuously
    - <= 16x16 on each cube face (overall 18 kB per light; probably GPU allocates more in the moment)
  - Moving objects (dynamic) CRSM: surfel size << LPV cell size
    - Need to make surfel flux less to reduce LPV cell flux changes to reduce popping
    - = 64x64 on each cube face (overall 288 kB)

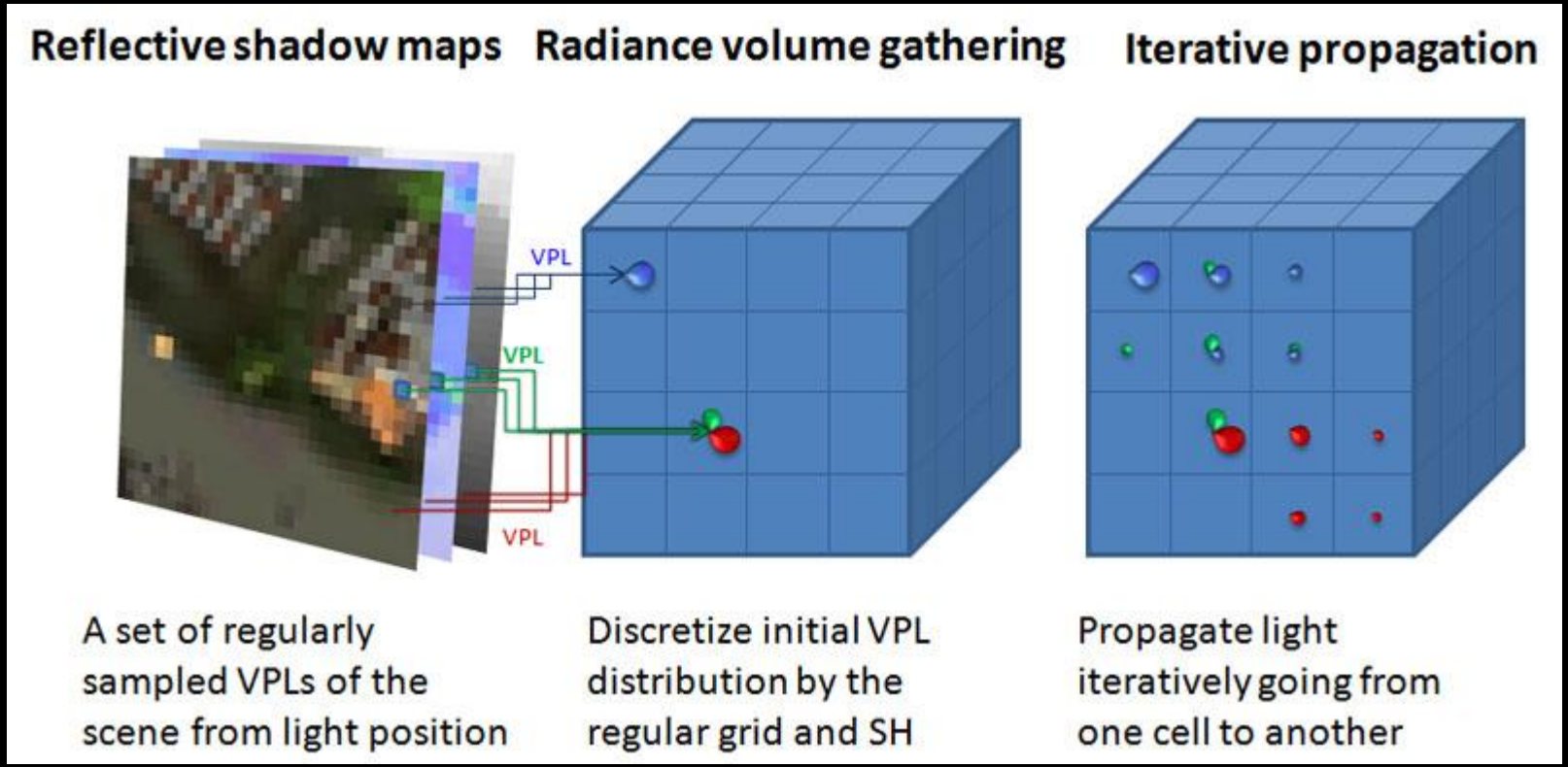
Capturing Light Data

- GI Data cache
  - Fastest pixels are those GPU doesn't render
  - Static CRSM is a cache
  - Dynamic CRSM is updated as soon as object reaches the area of influence

Light Injection in LPVs

- Light Propagation Volumes
  - Based on CRSM data the reflected flux is calculated
  - Compressed as two bands of the sh basis (i.e. four coefficients for R, G and B each)
    -> SH represents the directional distribution of intensity
  - … and stored in 3 Volume textures for R, G and B

Light Injection in LPVs



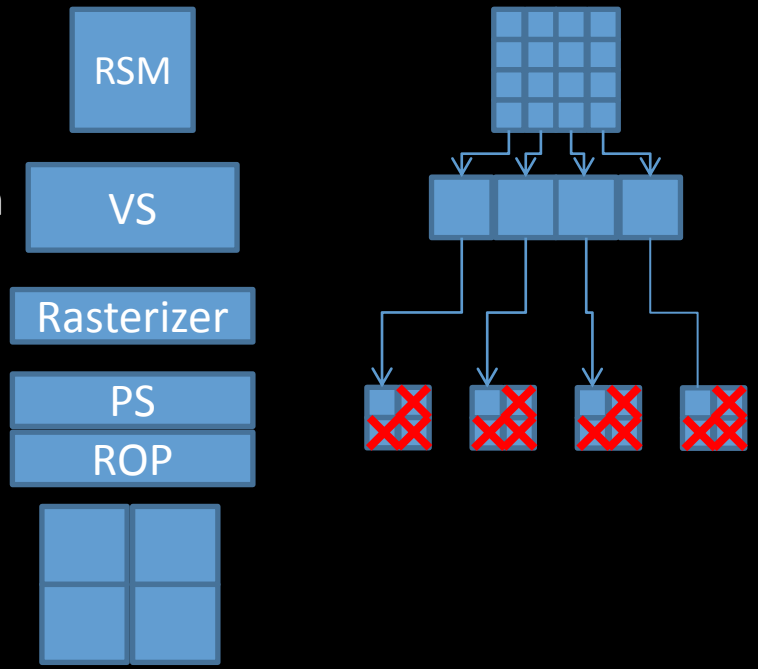
CONFETTI

Light Injection in LPVs

Points generated for each surfel

Calcs Surfel Position

PS/ROP runs on single pixel per quad



GS addresses slices in 3D texture

CONFETTI

Light Injection in LPVs

- Light Propagation Volumes
  - Distribution of SH compressed flux with spatial discretization
    - Flux distribution is approximated by SH cosine lobe representation
      -> Coefficients are scaled according to “how they are reflected by the surface” this is the magnitude of cosine lobe

Light Injection in LPVs

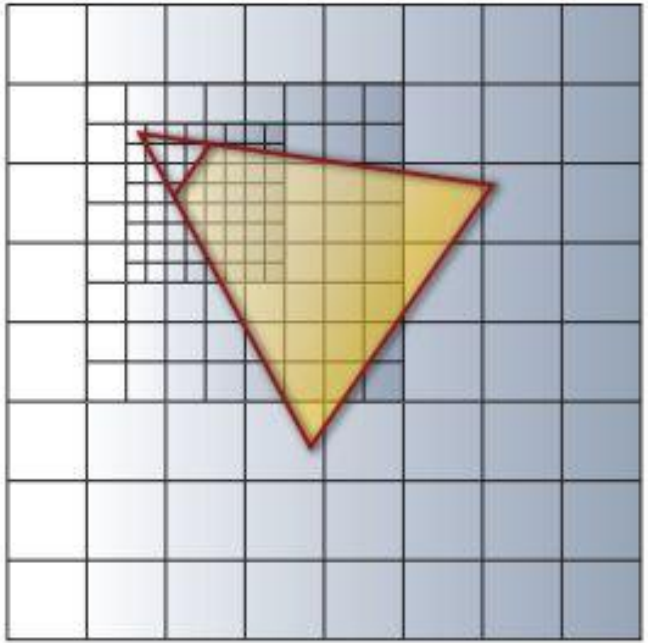
- Light Propagation Volumes
  - Distribution of SH compressed flux with spatial discretization
    - If the point light points away from the cell's center, we do not want to add his contribution to this cell, but rather to the next cell in order to avoid self-lighting and shadowing

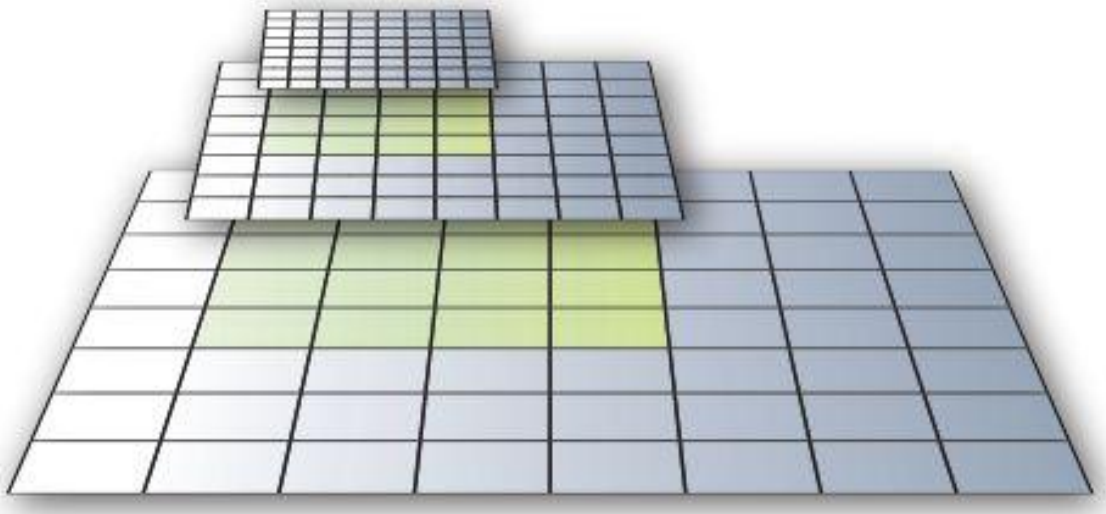
      -> Offset injection position in the normal direction by the half cell size in order to reduce self-illumination and shadows, which is an appropriate approximation in practice

Light Injection in LPVs

- Cascaded LPVs
  - There are 3x3 LPVs each 32x32x32 called Cascaded LPVs (+3x occlusion volume textures)
  - Cascaded LPVs are situated generally in front of the camera similar to Cascaded Shadow Maps [Engel05]
  - All volumes cover the camera position and bigger volume contains the smaller one
  - Only close lights are in the Cascade -> worst case in all three cascades

Light Injection in LPVs





Cascaded LPVs

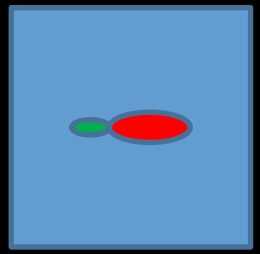
CONFETTI

Light Propagation in LPV

- Energy is propagated to the six neighbors along the axial directions
- All subsequent propagation steps take the LPV from the previous iteration as input and propagate as in the first iteration

Light Propagation in LPV

- Initially each cell contains the current light flow

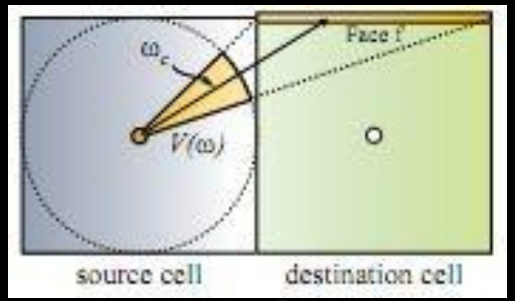


CONFETTI

Light Propagation in LPV

- This flow is propagated to the neighbor cells by converging (integrated over sphere) the flux flow with the 1st order SH approximation of the face visibility function
- The result is a scalar which modulates the same face visibility approximation

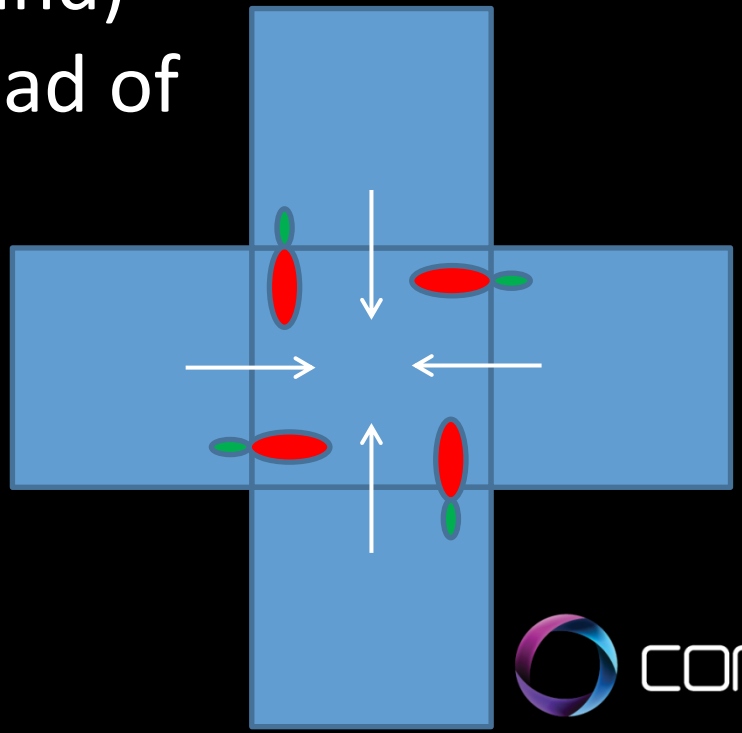
-> good enough approximation



CONFETTI

Light Propagation in LPV

- Propagate flow from neighbors to center (not the other way round)
  -> gathering data instead of scattering
  -> faster on DX10/11



CONFETTI

Apply Lights

- For each of the Cascaded LPV we apply the lights
- The radiance volume is rendered directly into the diffuse and specular light accumulation buffer on top (Light Pre-Pass [Engel])
- This way it is possible to compose complex layered lighting

Apply Lights

- For every pixel on screen we re-construct its position and look-up the LPV using the pixel's world position

Apply Lights

- SH trilinear interpolation of spatially approximated radiance may cause serious artifacts like unwanted self-illumination and light bleeding
- Needs to be dampened

Apply Lights

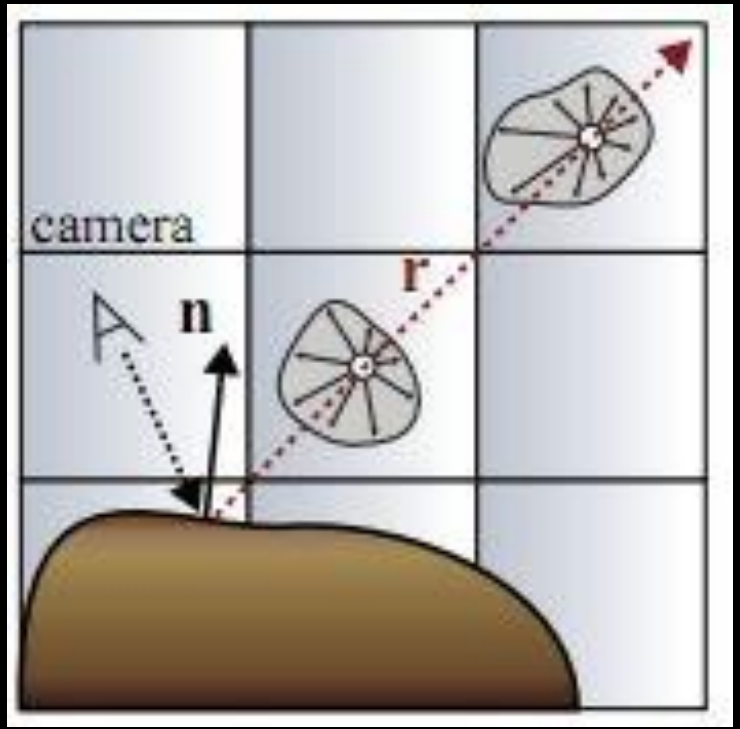
- This can be done by comparing primary light propagation directions [Kaplanyan]
- We project the gradient vector on the light flow direction vector and smoothstep the result
  -> this is the dampening factor

Apply Lights

- “Specular” == Glossy reflections [Kaplanyan]
- Tracing the source volume texture with regular steps towards a reflected direction
- Apply integration over the cone with respect to viewing angle
  -> glossiness depends on the initial cone angle of the outgoing reflected ray
- Fresnel is applied following the same idea

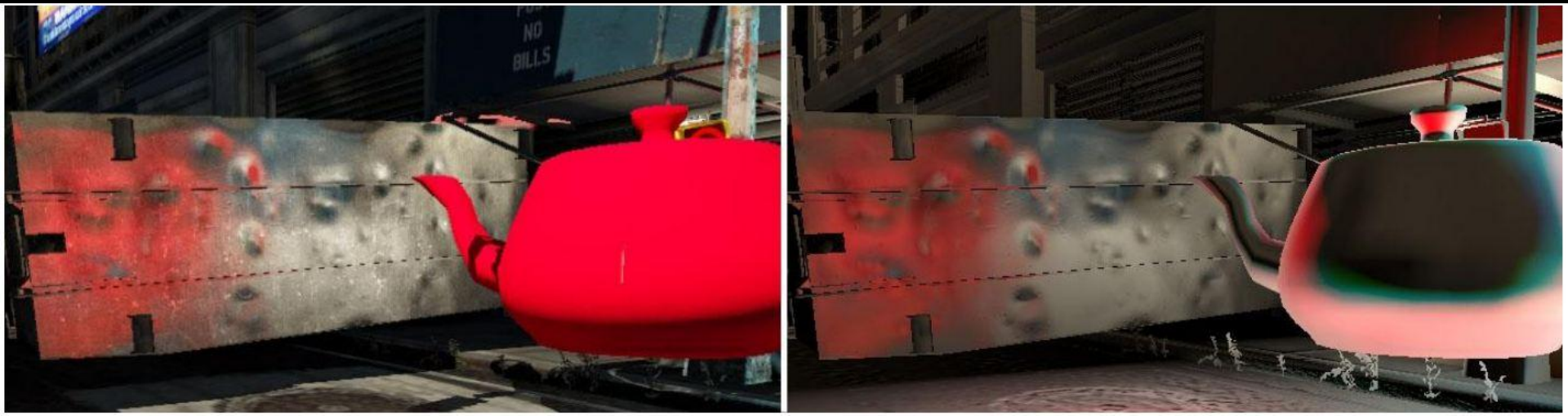
Apply Lights

- Ray marching in the LPV



CONFETTI

Apply Lights



CONFETTI

Feature Summary

- Allows randomly destructible Geometry
- No “typical” streaming bandwidth consumed -> open-world games
- Artist iteration time: real-time -> less cost in production
- Quality is scalable:
  - Bigger CRSM, bigger LPVs -> more memory for caching

Summary

- Future: optional will be higher quality algorithms
  -> expensive parts of the algorithm are cached and only processed during start-up (no offline compilation)

Thank you

Acknowledgements

- We would like to thank Carsten Dachsbacher and Anton Kaplanyan for discussions and support
- Peter Santoki and the whole team that worked on RawK II and the first RawK (check out the credits)

wolf@conffx.com

CONFETTI

References

- [Dachsbacher] Carsten Dachsbacher, Marc Stamminger, “Reflective Shadow Maps”, http://www.vis.uni-stuttgart.de/~dachsbcn/download/rsm.pdf http://www.vis.uni-stuttgart.de/~dachsbcn/publications.html
- [DachsbacherSii] Carsten Dachsbacher, Marc Stamminger, “Splatting Indirect Illumination”, http://www.vis.uni-stuttgart.de/~dachsbcn/download/sii.pdf
- [Engel05] Wolfgang Engel, “Cascaded Shadow Maps”, pp. 197–206, ShaderX5, Charles River Media, 2005
- [Engel] Wolfgang Engel, “Designing a Renderer for Multiple Lights: The Light Pre-Pass Renderer”, pp. 655–666 , ShaderX7, Charles River Media, 2007
- [Kaplanyan] Anton Kaplanyan, Wolfgang Engel, Carsten Dachsbacher, “Diffuse Global Illumination with Temporally Coherent Light Propagation Volumes”, GPU Pro 2, AK Peters 2011

CONFETTI

Miscellaneous

- Occlusion [Kaplanyan]
- SH projections of blocking potential written into a separate geometry volume (GV)
- Only works with one surface per grid cell