

**Temporal  
Radiance Caching**



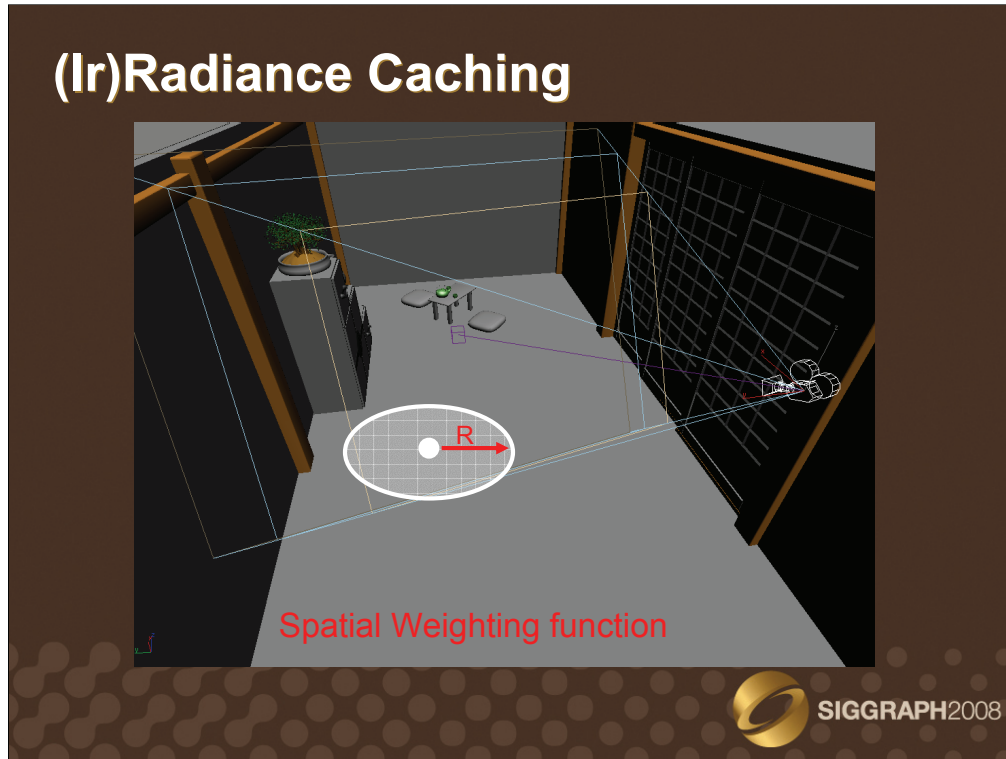
**SIGGRAPH2008**

Pascal Gautron

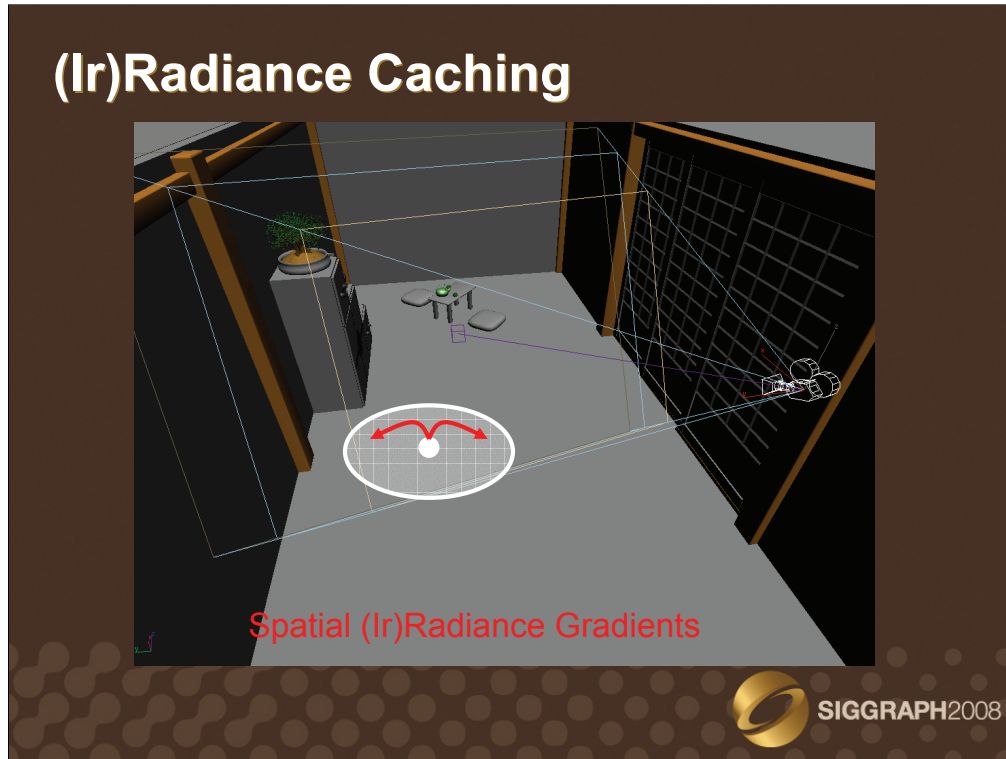
R&D Engineer

Thomson Corporate Research

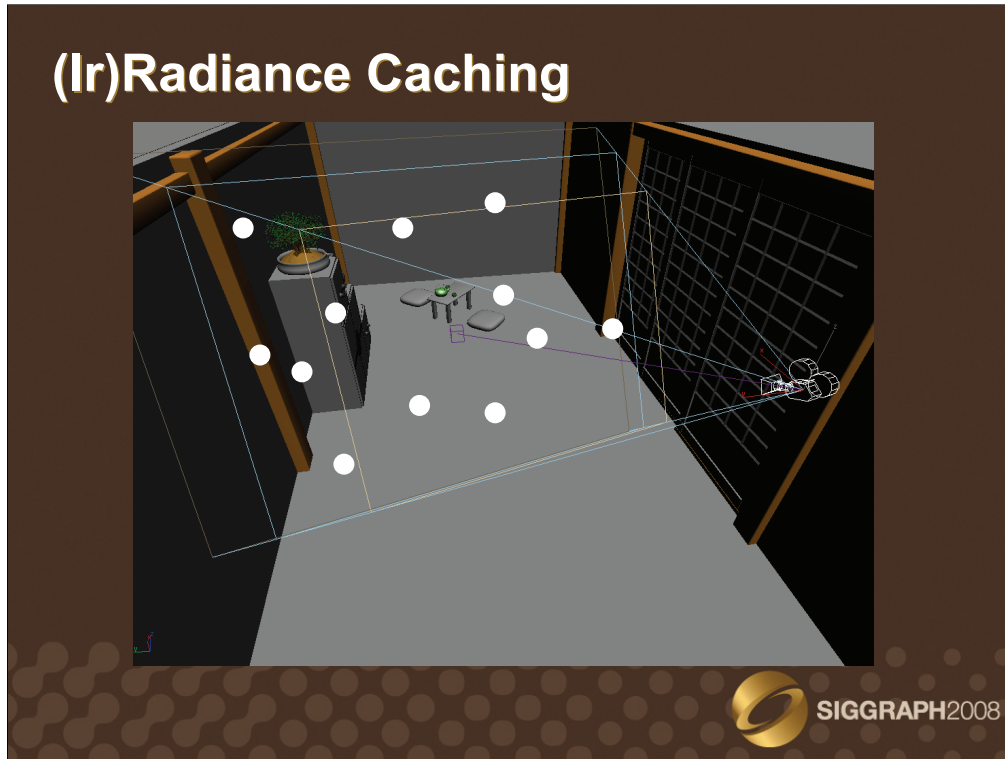
France



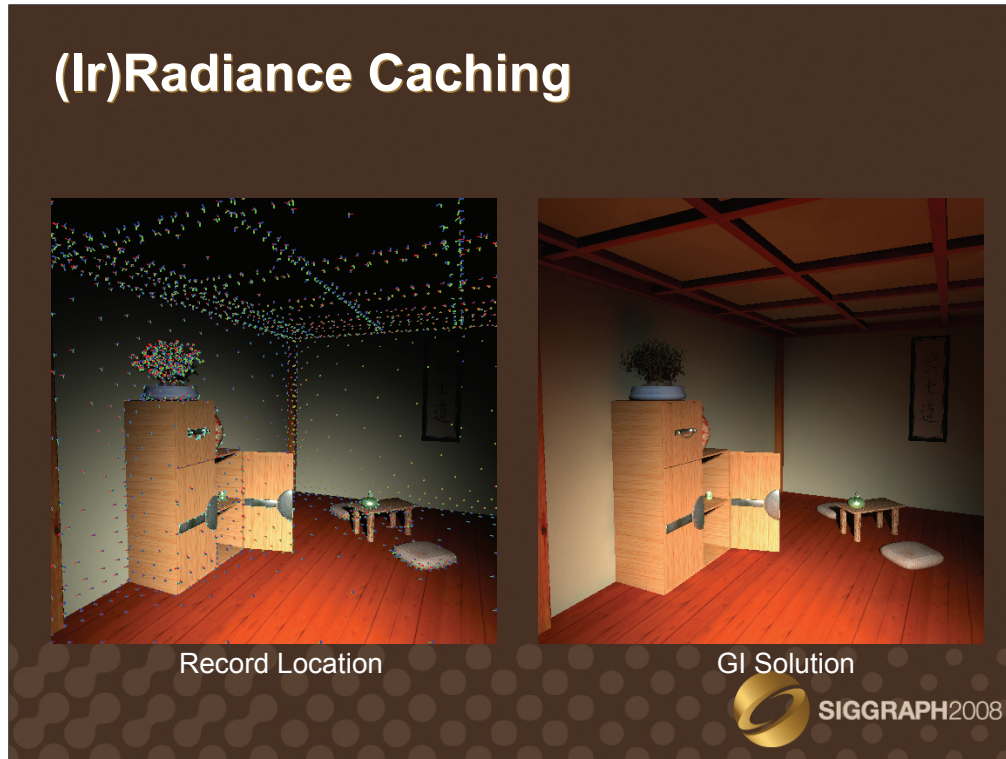
As explained in the previous parts of this course, the zone of influence of a given irradiance record is defined by a spatial weighting function.



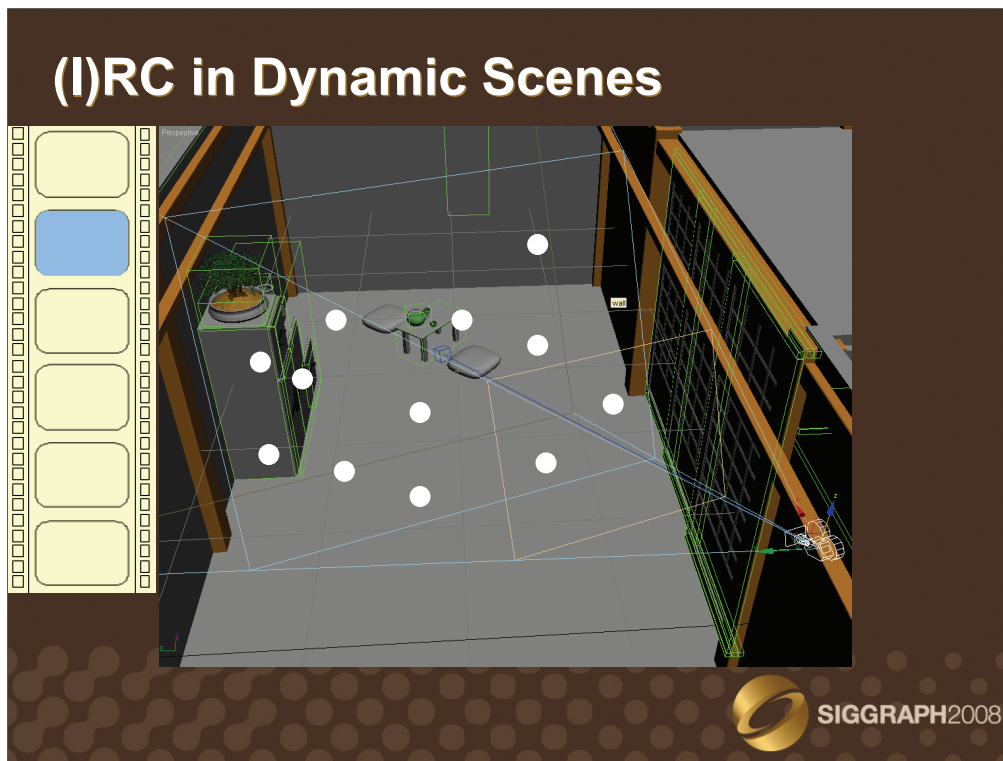
The contribution of a record within its zone of influence is estimated using irradiance gradients [Ward92].



The lighting of the visible points is then computed explicitly at the location of the records, and **extrapolated** for all the other visible points.

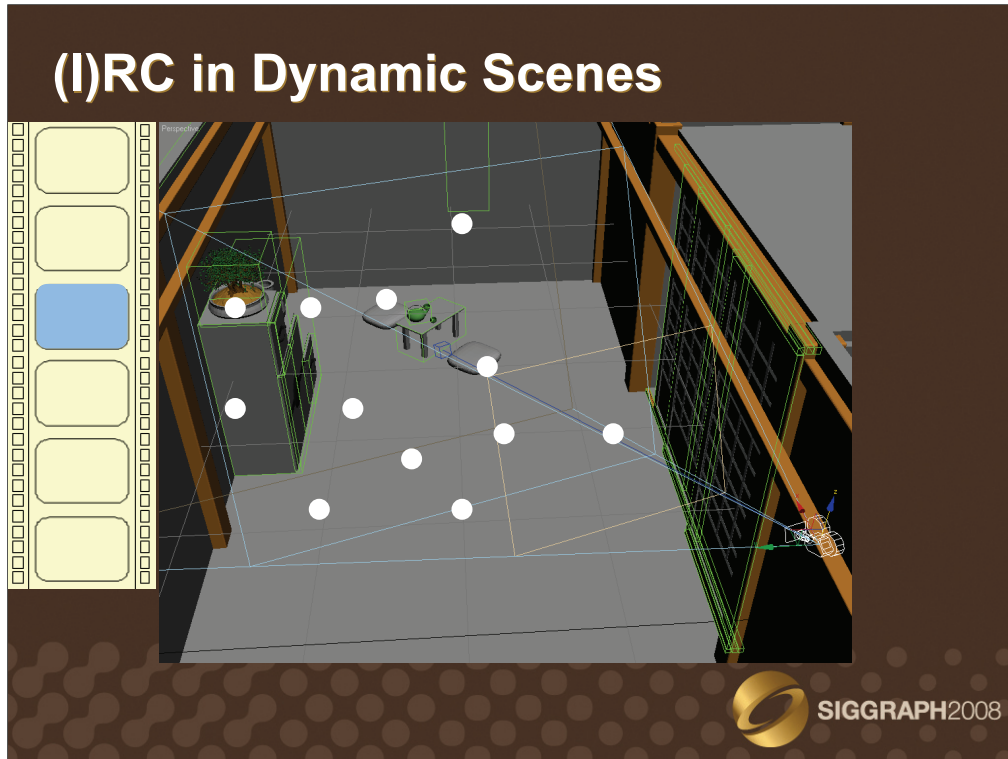


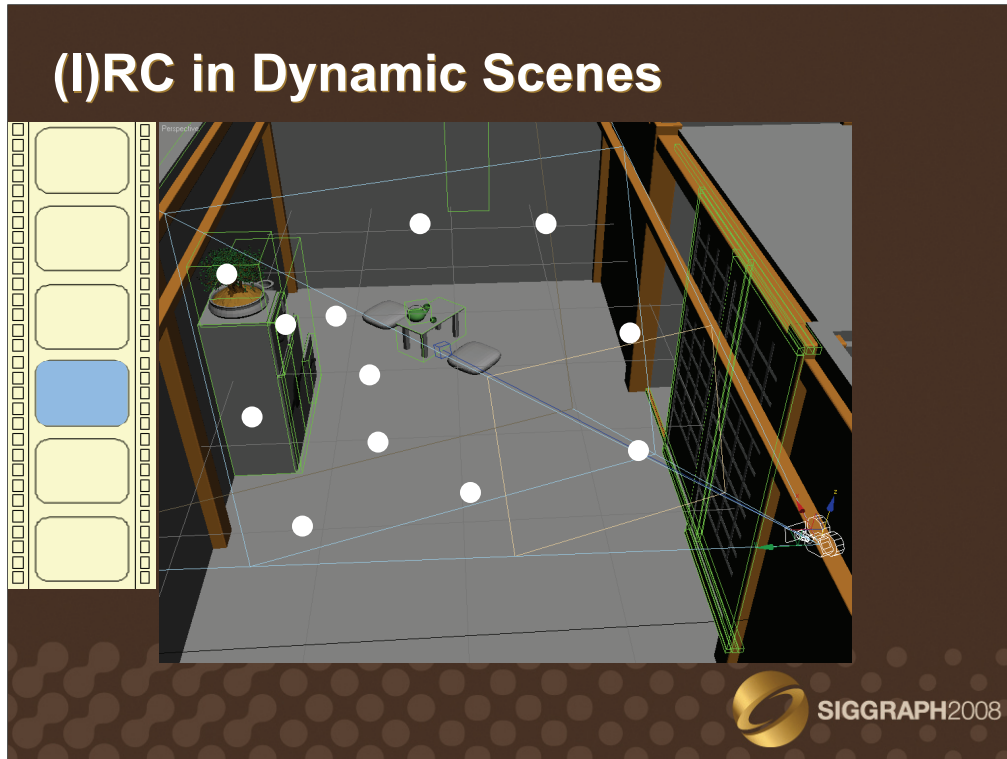
In a single image, this method provides high quality results even when using a very sparse sampling. However, since the estimated lighting is generally obtained through extrapolation from the location of the records, the result depends on the **distribution** of the records.



In dynamic scenes, the indirect lighting must be changes in each frame. A simple method for animation rendering using irradiance caching is the entire recomputation of the global illumination solution for each frame. The camera and objects being dynamic, the distribution of records is likely to change across frames, yielding flickering artifacts. A video illustrating this method can be found on [MyWebSite].

Furthermore, the indirect lighting tends to change slowly across frames. The indirect lighting computed at a given frame may thus be reused in several subsequent frames without degrading the quality. However, the lighting may change very quickly in some areas of the scene, and be nearly-static elsewhere. The method described in this course leverages these observations by assigning a distinct lifespan to each record in the cache.







# Outline

## Temporal (ir)radiance interpolation scheme

Flying Kite - No temporal coherence  
Rendering time: 5109s

Flying Kite - Interpolated gradients  
Rendering time: 783.4s Speedup: 6.52x

Temporal weighting function  
Fast estimate of future indirect lighting  
Temporal gradients

SIGGRAPH2008

More precisely, the method described in this course is an extension of the irradiance caching interpolation scheme to the temporal domain. To this end, we devise a temporal weighting function to determine the lifespan of a record, e.g. the number of frames in which a record can be reused without degrading the rendering quality. The contribution of a record within its lifespan is estimated using temporal irradiance gradients.

## Temporal Weighting Function

Estimate the temporal change rate of indirect lighting



The spatial weighting function described in [Ward88] is based on an estimate of the change of indirect lighting with respect to displacement and rotation. The temporal weighting function is thus based on an estimate of the **temporal change** of indirect lighting across frames.

## Temporal Weighting Function

Estimate the temporal change rate of indirect lighting

$$\begin{aligned}\frac{\partial E}{\partial t}(t_0) &\approx \frac{E_t - E_{t+1}}{\delta_t} \\ &= E_0(\tau - 1)\end{aligned}$$



$$\tau = E_{t+1}/E_t$$




Let us consider the irradiance  $E_t$  at current frame, and the irradiance  $E_{t+1}$  at next frame. The temporal change of indirect lighting can be estimated by a numerical estimation of the temporal derivative of the lighting. We define the value  $\tau$  as the ratio of the future and current lighting.

## Temporal Weighting Function


Inverse of the temporal change rate of indirect lighting

$$w_k^t(t) = \frac{1}{(\tau - 1)(t - t_0)} > 1/a^t$$

⇒  $\delta_k(t) = \frac{a^t}{(\tau - 1)}$



**Problem :**  
**Lifespan  $\delta_k$  is determined when the record is created**

$$\tau = E_{t+1}/E_t$$


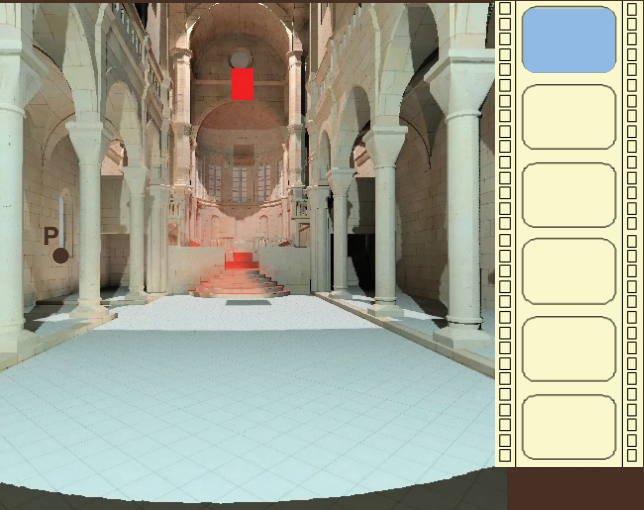
Using a derivation similar to [Ward88], we obtain a temporal weighting function defined as the inverse of the change of indirect lighting over time. The lifespan of a record is thus adapted to the **local change** of indirect lighting: fast changes yield a short lifespan, while records can be reused across many frames if the temporal changes are slow. The user-defined threshold value  $a^t$  conditions the temporal accuracy of the computation: a high value leads to long lifespans, hence reducing the rendering time at the detriment of the quality. Conversely, a small value ensures frequent updates at the expense of rendering time.

At the end of its lifespan, a record is discarded and replaced by a new record located at the same point. This method strenghtens the temporal coherence of the distribution of records, hence avoiding flickering artifacts due to changes of record distribution.


However, the lifespan is determined using the indirect lighting at current and next frames only. Therefore, later changes do not affect this lifespan, hence harming the reactivity of the algorithm. An overestimation of the lifespan yields residual global illumination effects known as « ghosting artifacts ».

## Lifespan Thresholding

At point P and time t:  
Static environment  
↓  
 $\tau = E_{t+1}/E_t = 1$   
↓  
 $w_k^t(t) = \infty$  for all t  
↓  
Infinite Lifespan



The image shows a 3D rendered scene of a cathedral interior. A red square is placed at a point labeled 'P' on the floor. To the right of the scene is a vertical film strip with several frames. The top frame is blue, and the others are yellow. The scene is lit with a warm, golden light, and the floor is a light blue color.

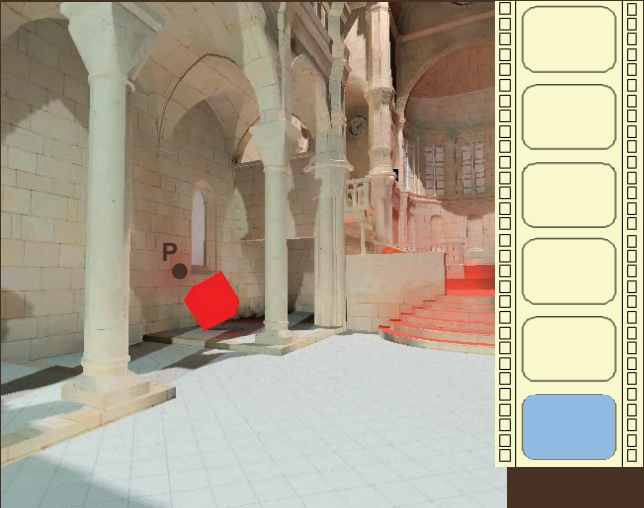


The SIGGRAPH 2008 logo is located in the bottom right corner of the slide.


Let us consider an example: at a point P and time t and t+1, the environment is static. Therefore, the indirect lighting is considered constant. In this case, the value of the temporal weighting function is infinite for any frame. The lifespan of the record is then considered infinite.

## Lifespan Thresholding

At point P and time t:  
Static environment  
↓  
 $\tau = E_{t+1}/E_t = 1$   
↓  
 $w_k^t(t) = \infty$  for all t  
↓  
Infinite Lifespan



|| →  $w_k^t(t) = 0$  if  $t - t_k > \delta_{tmax}$



However, it is easy to show that the lighting at P may change afterwards. Since the weighting function returns an infinite value regardless of the frame, the value of the lighting at P will never be updated, yielding ghosting artifacts. We do not solve this problem, but we simply ask the user to define a maximum lifespan for all records. This maximum lifespan ensures the update of the lighting at least after

## Temporal Weighting Function

Determines the lifespan of the records

Lifespan depends on the local change of incoming radiance

If the environment is static, threshold the lifespan to a maximum value

### However

$$\tau = E_{t+1}/E_t$$

Requires the knowledge of future irradiance



SIGGRAPH2008

The temporal weighting function is used to determine the lifespan of each record based on the local change of incoming radiance over time. The length of the lifespan is shortened or lengthened with respect to the magnitude of the change of indirect lighting. However, the estimate of this change is based on the knowledge of the indirect lighting at next frame,  $E_{t+1}$ . Since this value is generally unknown, next section will devise a method for fast estimation of the future incoming without actual sampling.

# Contributions

## Temporal (ir)radiance interpolation scheme


Flying Kite - No temporal coherence  
Rendering time: 5109s

Temporal weighting function

Fast estimate of future indirect lighting

Temporal gradients


Flying Kite- Interpolated gradients  
Rendering time: 733.4s Speedup: 6.52x



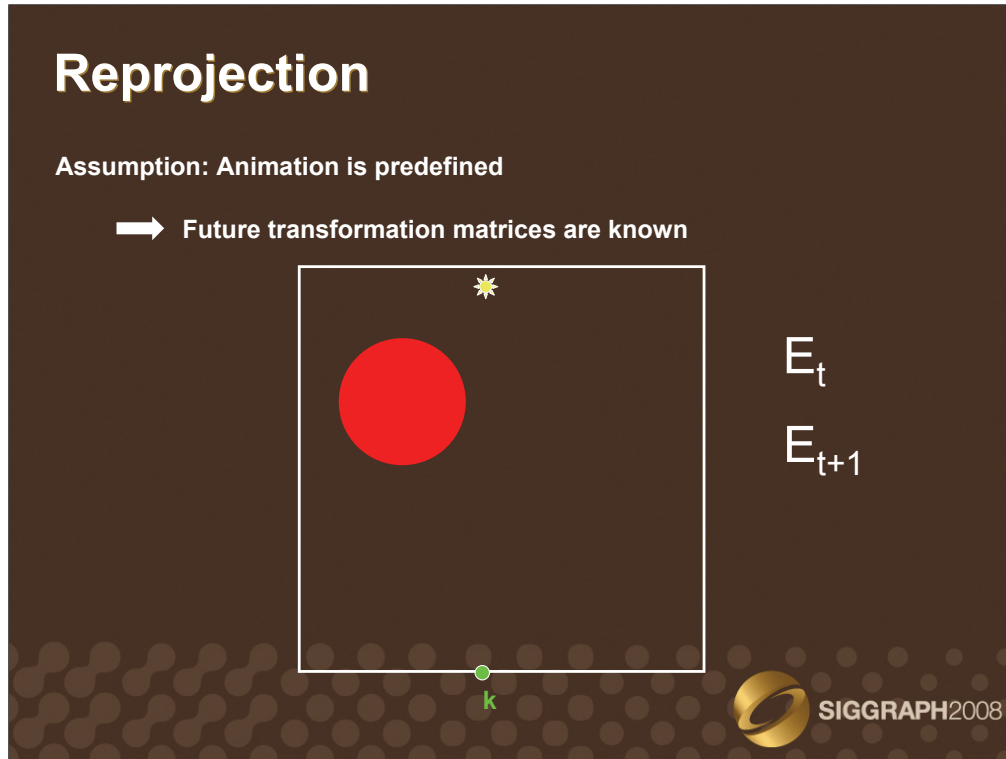


## Future Incoming Lighting

$E(P, t) =$     $\approx$   $E(P, t+1) =$   

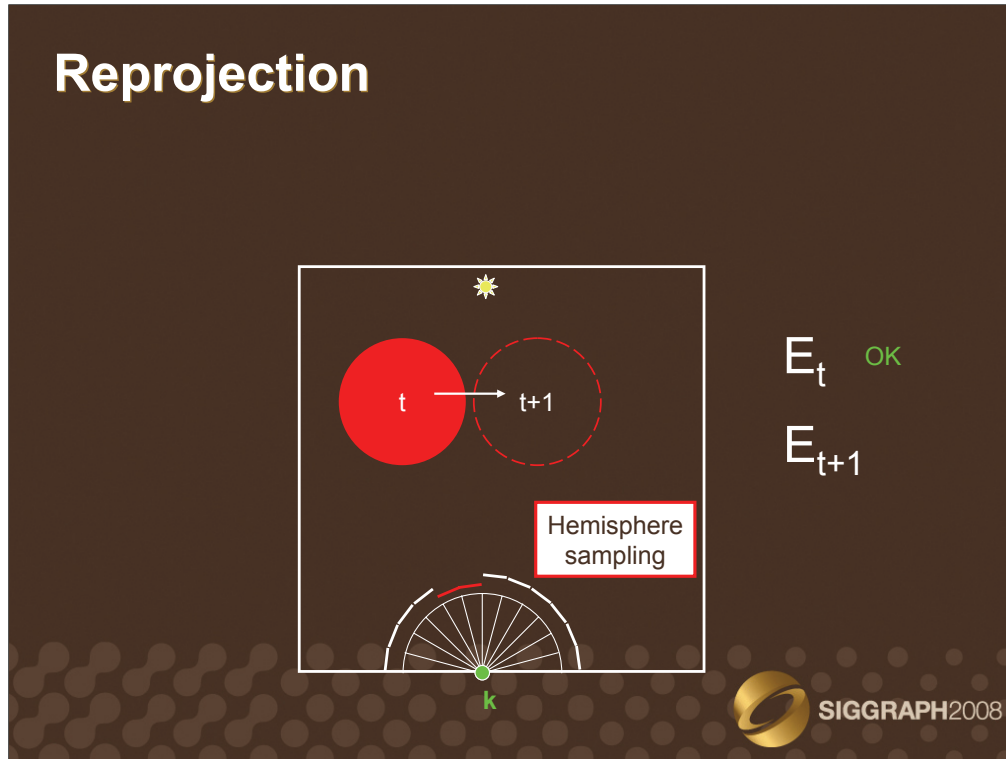
 SIGGRAPH2008

This approach is based on the following observation: between time  $t$  and  $t+1$ , the change of lighting is low. Based on the knowledge of the dynamic properties of surrounding objects, we propose a method based on reprojection to estimate the future incoming lighting using only the data available at current frame.



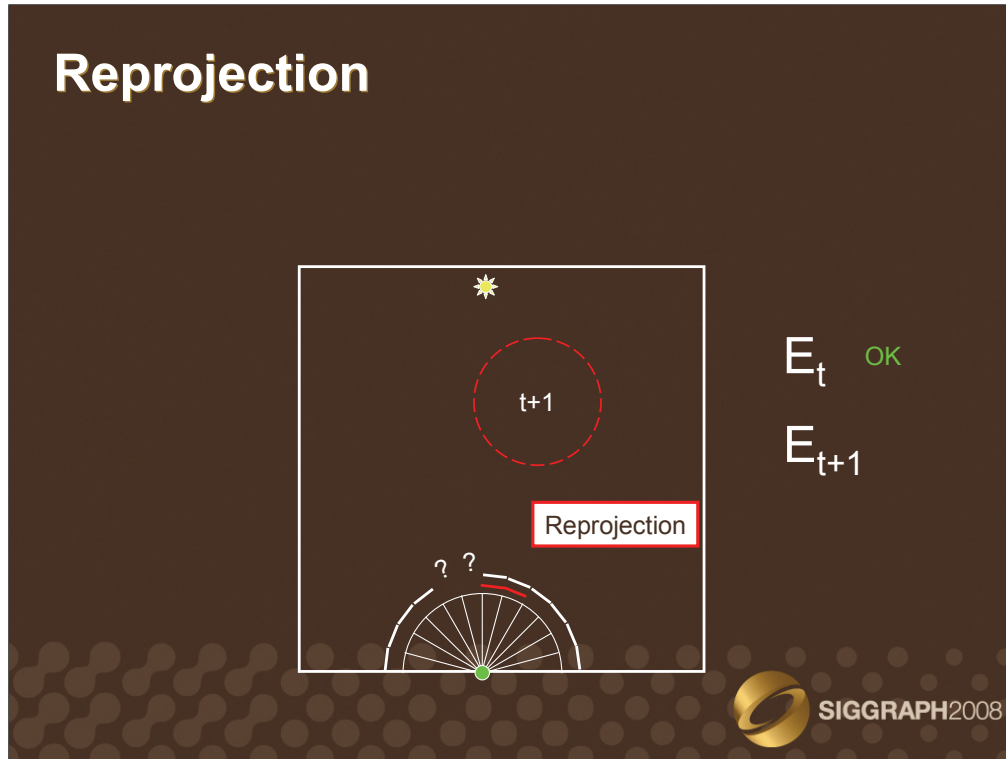
Our reprojection method is similar to the one used in the Render Cache [Walter99]. However, in the Render Cache, the reprojection is used to avoid tracing primary rays from the camera. In our case, the reprojection is only used to estimate the future incoming radiance at the location of a record.

Let us consider a point  $k$  at which we want to create a record.

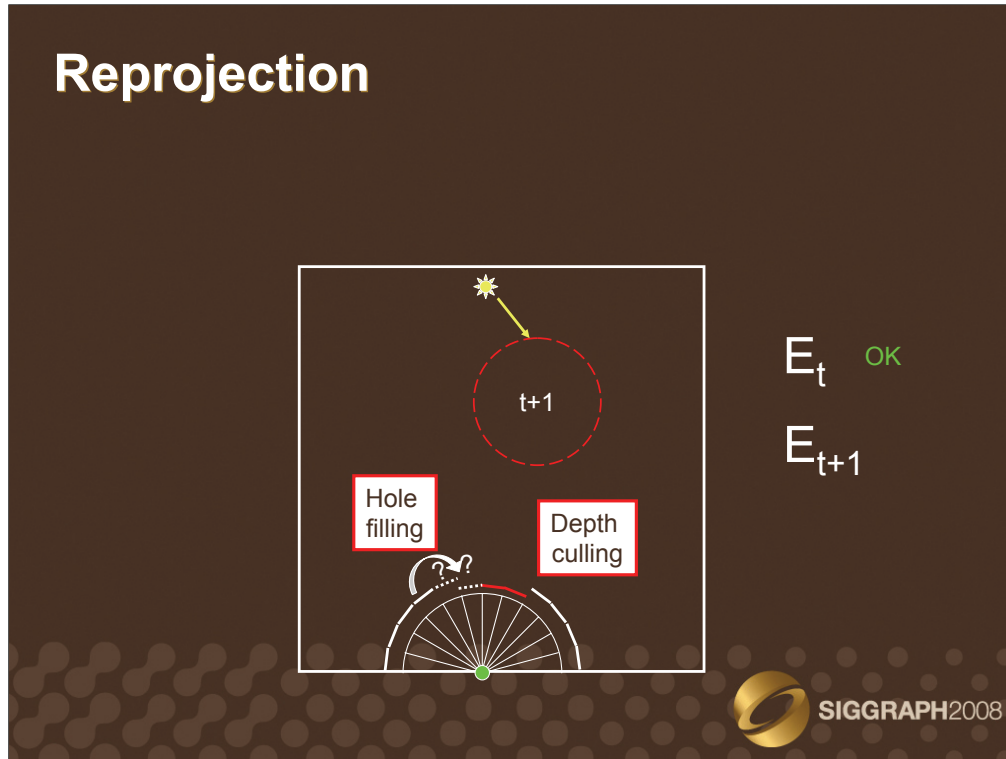


The first step is the computation of the incoming radiance in  $k$  at time  $t$ . This is performed by sampling the hemisphere above  $k$  either using ray tracing or GPU rasterization as explained in the previous chapter.

Assuming the animation is known in advance, the position of the red sphere at time  $t+1$  is known.

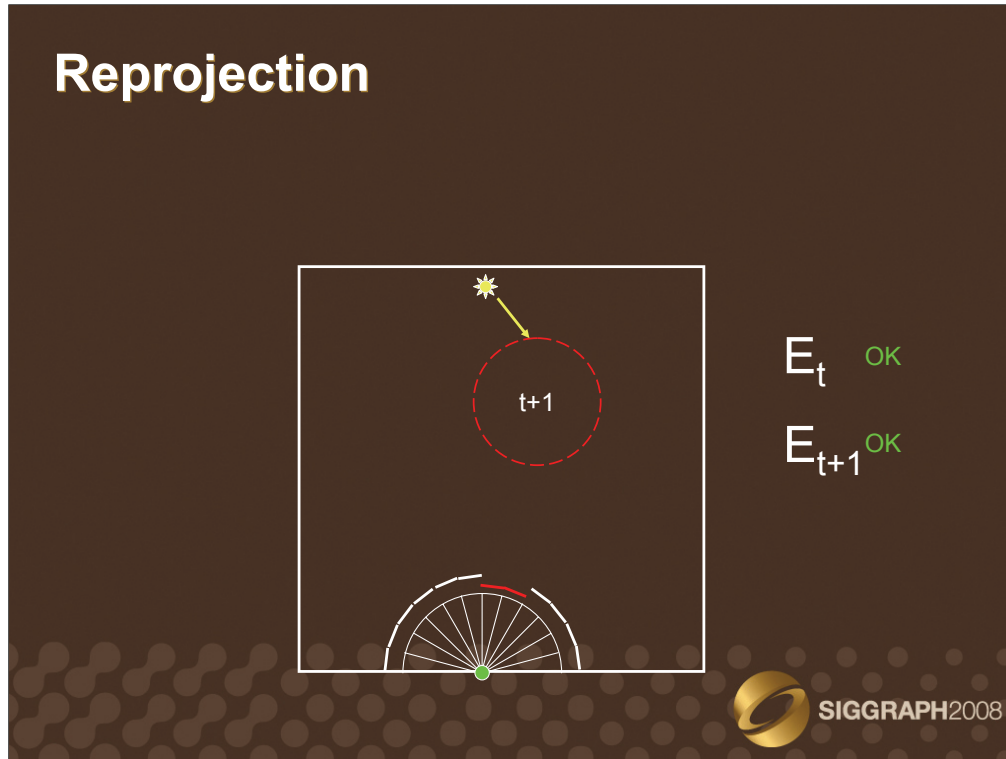


This future position of the red sphere is used to reproject the radiance samples corresponding to the sphere to the new position. This reprojection yields missing information at the former location of the sphere, and overlapping values at the novel position.

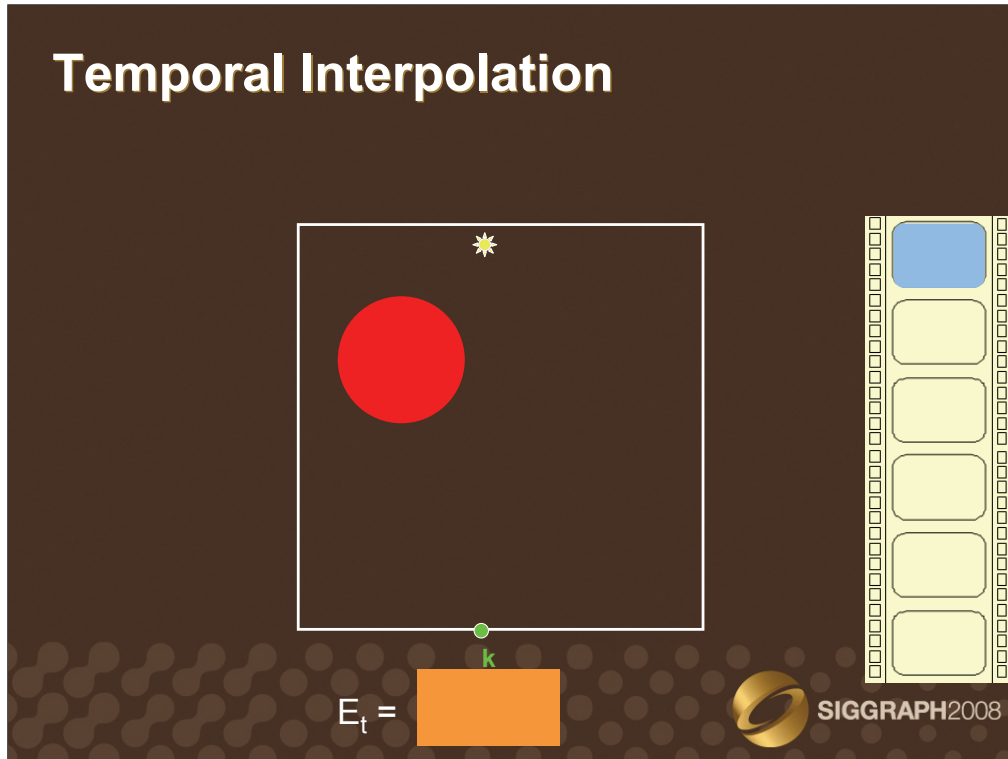


A depth culling step chooses the closest value in the case of overlapping radiance values. In this case, the background values are culled.

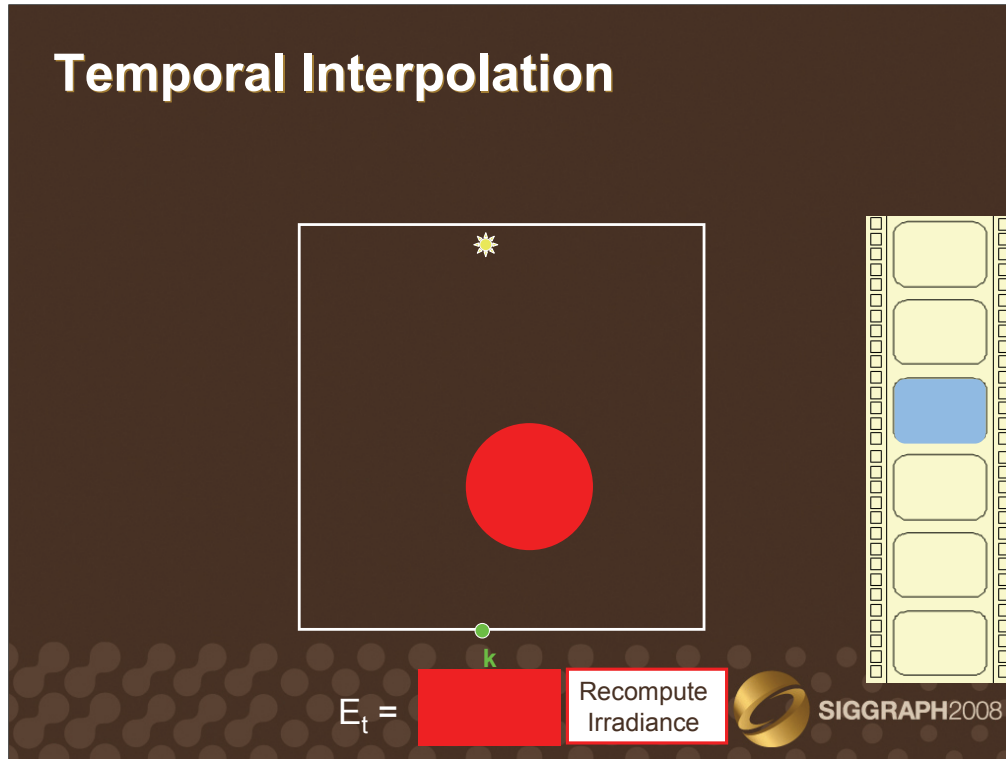
A simple hole-filling is applied in strata where information is missing. Since we only deal with small movements (1 frame), we simply fill the holes using the closest neighboring value.



After depth culling and hole filling, our method provides a reliable estimate of the future incoming lighting. The lifespan of the record created in  $k$  is thus completely defined.



At this point, we know that the value of record k can be reused across n frames.



At the end of the record lifespan, a new record is computed, containing an up-to-date irradiance value. In this case the red sphere got closer to  $k$ , hence the irradiance at time  $t$  and  $t+n$  are noticeably different. A consequence of this brutal replacement is sudden changes in the color of image portions, known as « popping » artifacts.



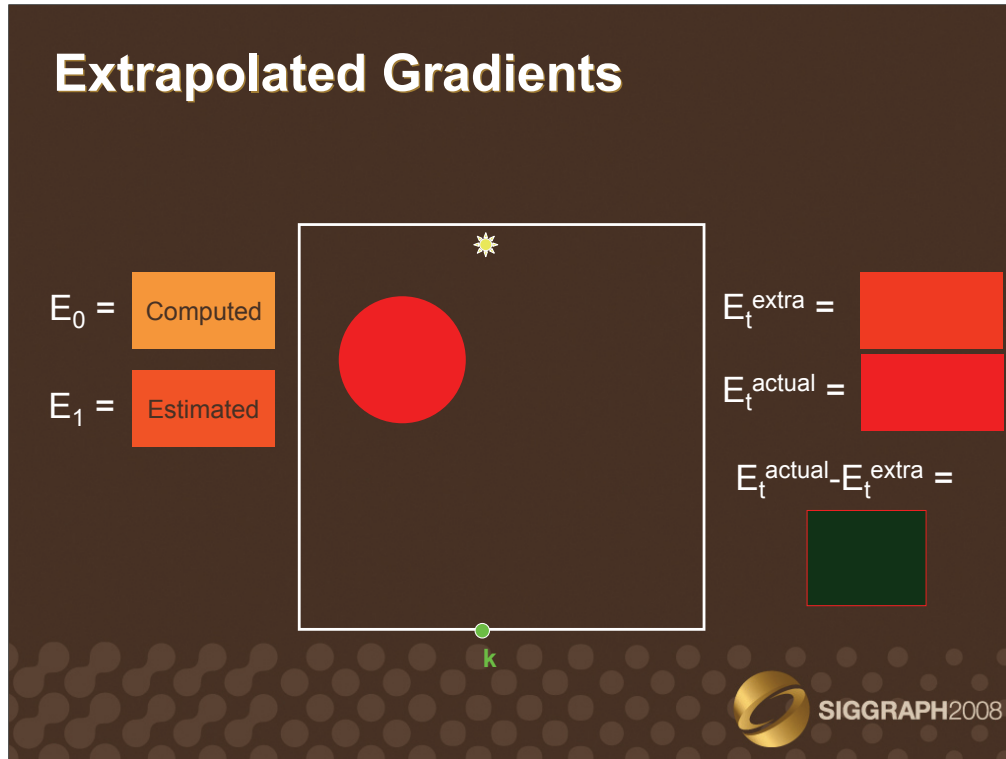
# Contributions

## Temporal (ir)radiance interpolation scheme

Flying Kite - No temporal coherence  
Rendering time: 5109s

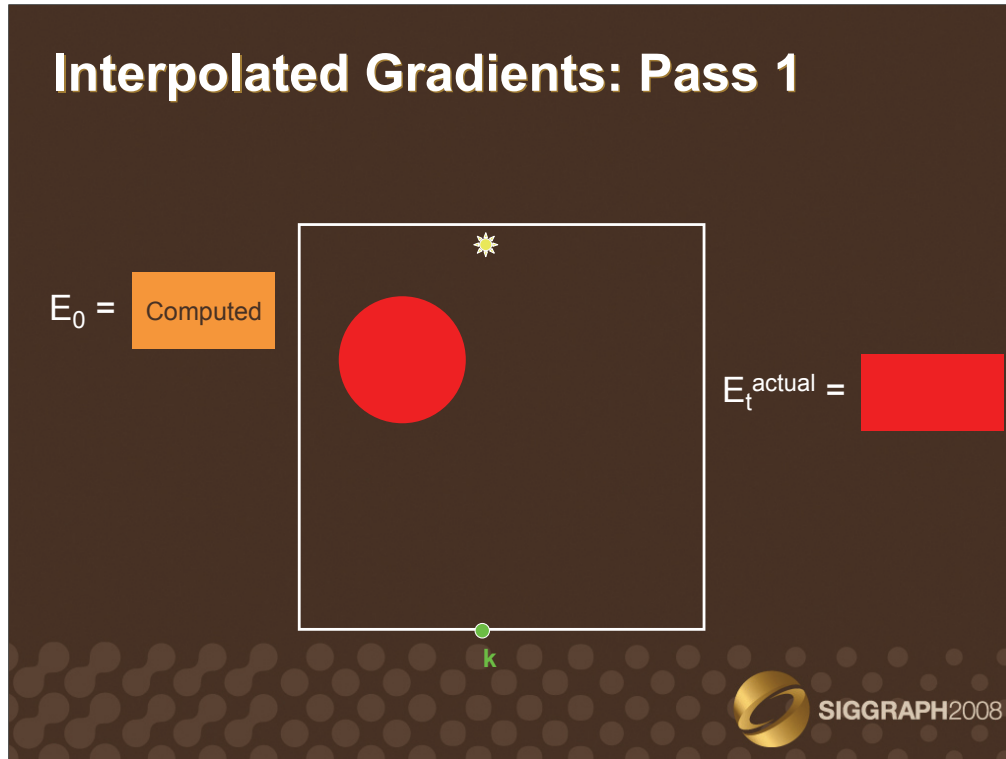
Temporal weighting function  
Fast estimate of future indirect lighting  
Temporal gradients

Flying Kite- Interpolated gradients  
Rendering time: 733.4s Speedup: 6.52x



First, we propose to use the estimate of the future incoming lighting to extrapolate the lighting over the entire lifespan of the record. While this method reduces the gap between the extrapolated and the actual irradiance values, the difference is still not negligible. As a consequence, some popping artifacts will remain visible.

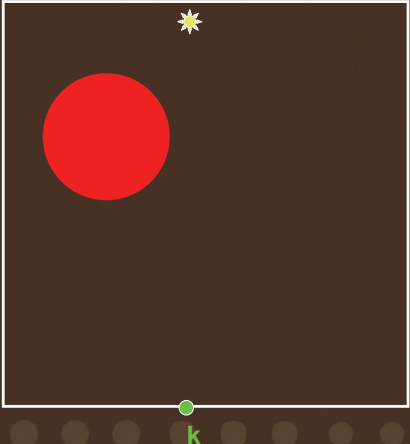
This method has one major advantage: the indirect lighting can be computed and displayed on the fly, as the animation is played. Particularly, such gradients could be used in the context of interactive global illumination computation.



Interpolated gradients completely avoid the popping artifacts by temporally interpolating the irradiance. In a first pass, records are generated as explained previously: each record is used within its lifespan, and new records are computed as the lifespans expire. Let us consider a record  $R_0$  computed at time  $t_0$  and located at point  $k$ . When this record expires after  $n$  frames, it is replaced by a new record  $R_1$  also located at point  $k$ . The temporal gradient of  $R_0$  is then deduced from  $(E_{R_1} - E_{R_0})/n$ .

This gradient approximates the change of lighting within the lifespan of a record by linearly interpolating the irradiance at the beginning and at the end of the lifespan. While completely removing popping artifacts, this linear approximation may smooth out high frequency changes that may happen during the lifespan of the record. Accounting for such changes either require a reduction of  $a^t$  and  $\delta_{t_{\text{max}}}$ , or the definition of a higher order interpolation scheme.

## Interpolated Gradients: Pass 2



The diagram shows a scene with a sun at the top, a red sphere in the center, and a point  $k$  on a surface at the bottom. A white square frame encloses the scene.


$E_0 =$  Computed

$E_t =$  Computed

$E_t^{inter} =$   

$E_t^{actual} =$   

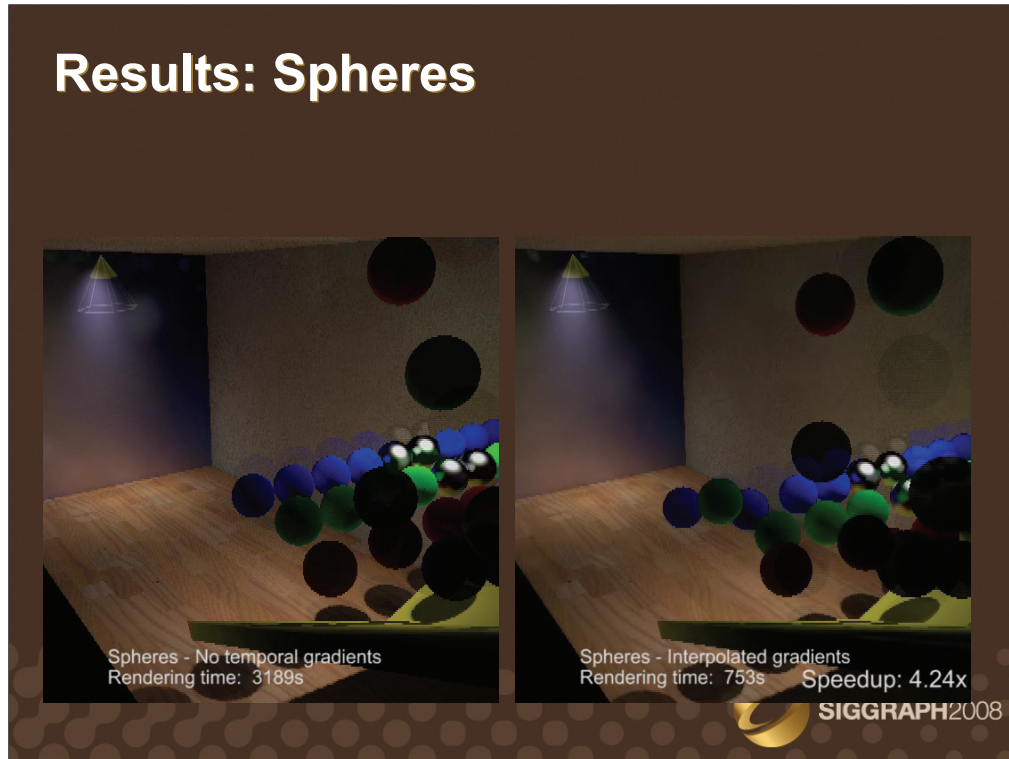
$E_t^{actual} - E_t^{inter} =$   



SIGGRAPH 2008



Videos illustrating the interpolated gradients and the adaptive record lifespan in different scenes can be found in [MyWebSite].



This method can be straightforwardly extended to nondiffuse interreflexions using radiance caching.

## Conclusion

Temporal radiance interpolation scheme

Reuse records across frames

```
graph TD; A[Reuse records across frames] --> B[Quality improvement]; A --> C[Speedup];
```


Quality improvement

Speedup

Dynamic objects, light sources, viewpoint

Easily integrates within (ir)radiance caching-based renderers

GPU Implementation

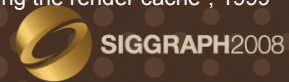


The method described in this part is based on the reuse of irradiance records across frames. While reducing the computational cost of animation rendering, the flickering artifacts are drastically reduced. This method supports any type of dynamic scene components, and can be easily integrated in existing renderers.

Future work will consider the elimination of the maximum lifespan  $\delta_{tmax}$ . Also, we would like to devise methods for higher order temporal interpolation to account for sharp changes of indirect lighting.

## Bibliography

- [Foley05] Foley T., Sugerma J. "KD-tree acceleration structures for a GPU raytracer", 2005
- [Krivanek05a] Krivanek J., Gautron P., Pattanaik S., Bouatouch K. "Radiance Caching for Efficient Global Illumination Computation", 2005
- [Krivanek05b] Krivanek J., Gautron P., Bouatouch K., Pattanaik S. "Improved Radiance Gradient Computation", 2005
- [LC04] Larsen B. D., Christensen N. "Simulating photon mapping for real-time applications", 2004
- [Purcell02] Purcell T. J., Buck I., Mark W. R., Hanrahan P. "Ray tracing on programmable graphics hardware", 2002
- [Purcell03] Purcell T. J., Donner C., Cammarano M., Jensen H. W., Hanrahan P. "Photon mapping on programmable graphics hardware", 2003
- [Walter99] Walter B., Drettakis G., Parker S. "Interactive rendering using the render cache", 1999





## Bibliography

[Ward88] Ward G. J., Rubinstein F. M., Clear R. D. "A ray tracing solution for diffuse interreflection", 1988

[Ward92] Ward G. J., Heckbert P. S. "Irradiance gradients", 1992

[Wil78] Williams L. "Casting curved shadows on curved surfaces", 1978

Additional materials available on [MyWebSite]:  
<http://gautron.pascal.free.fr>

