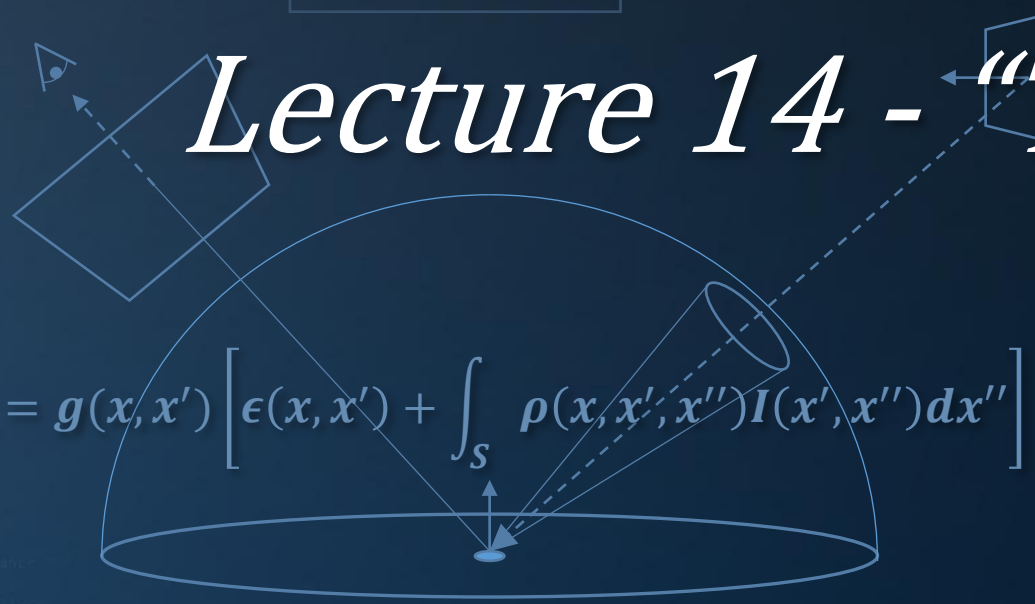
INFOMAGR – Advanced Graphics

Jacco Bikker - November 2021 - February 2022

Lecture 14 - "TAA & ReSTIR"



$$I(x,x') = g(x,x')\left[\epsilon(x,x') + \int_S \rho(x,x',x'')I(x',x'')dx''\right]$$

Welcome!

Today’s Agenda:

- Prerequisits
- Reprojection
- Resampling
- ReSTIR

Introduction

Towards Noise-free Path Tracing

Ingredients:

Convergence:

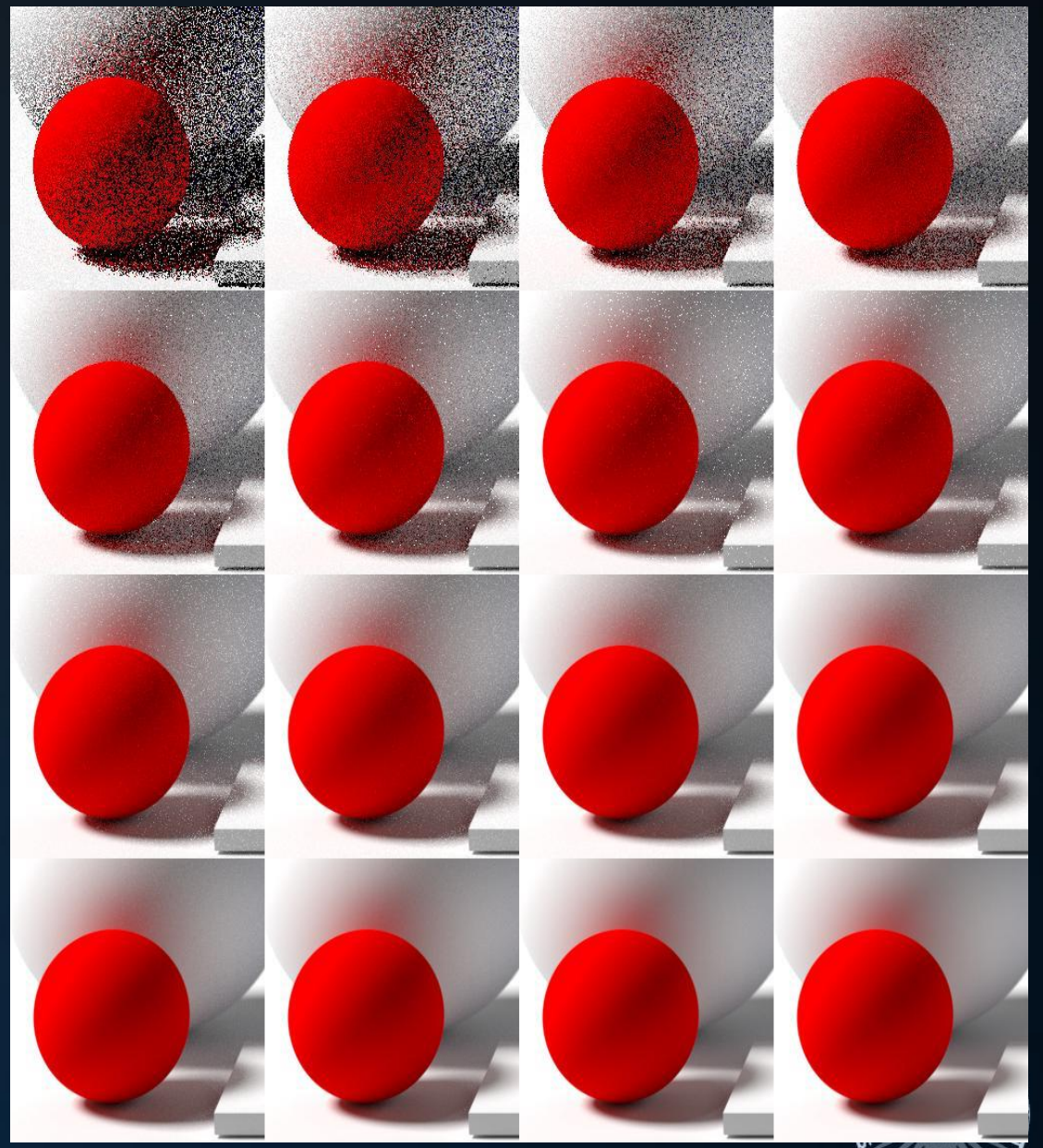
Average many samples, rely on the law of large numbers to approach E.

Importance Sampling:

"Work smarter, not harder", by taking better samples.

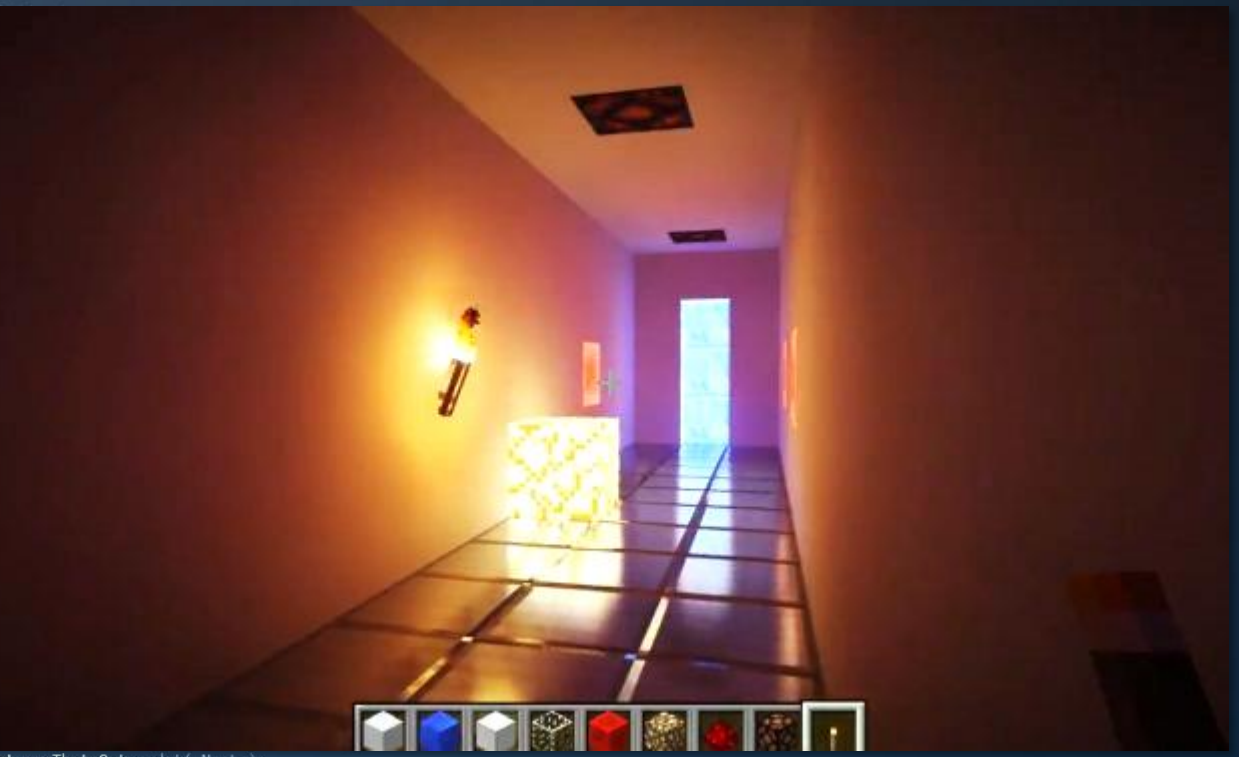
Specialized techniques:

The right tool for the task; e.g. light tracing / photon mapping for caustics.



Introduction

Towards Noise-free Path Tracing



SEUS mod for Minecraft

Quake II RTX

Introduction

Towards Noise-free Path Tracing

Game graphics:

- 60fps
- 1spp
- …
- Profit

But:

- It is good if it looks good
- Developer control lights, geometry, camera (somewhat)

Opportunity:

- Trade some bias for performance.

The Tomorrow Children, PS4

Today’s Agenda:

- Prerequisits
- Reprojection
- Resampling
- ReSTIR

ReSTIR, ingredient 1: Reprojection

Reprojection

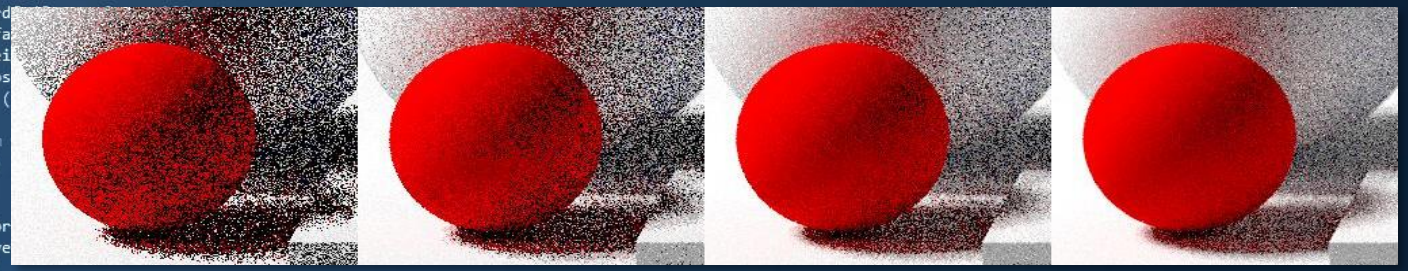
Assumptions

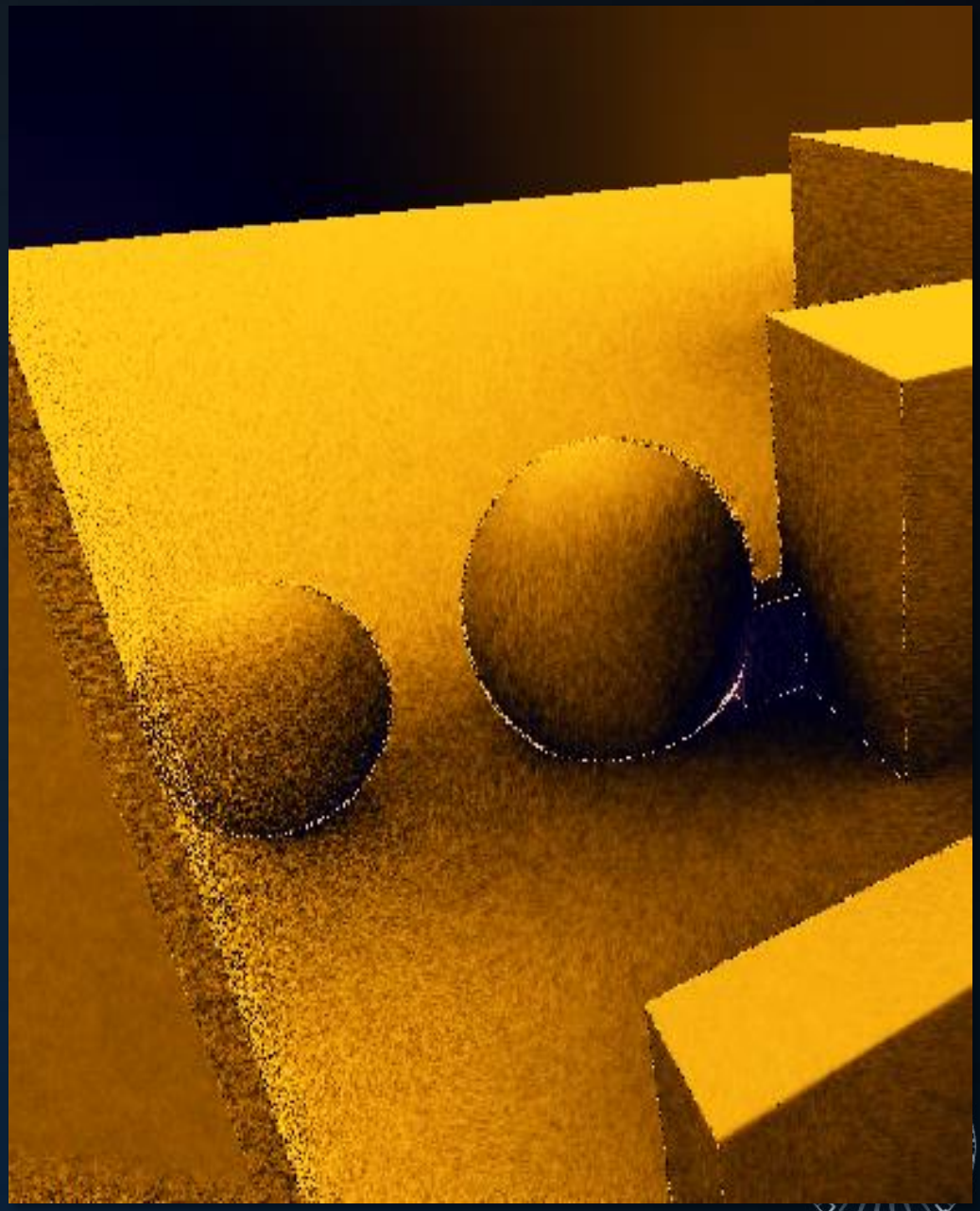
Converging requires a stationary camera.

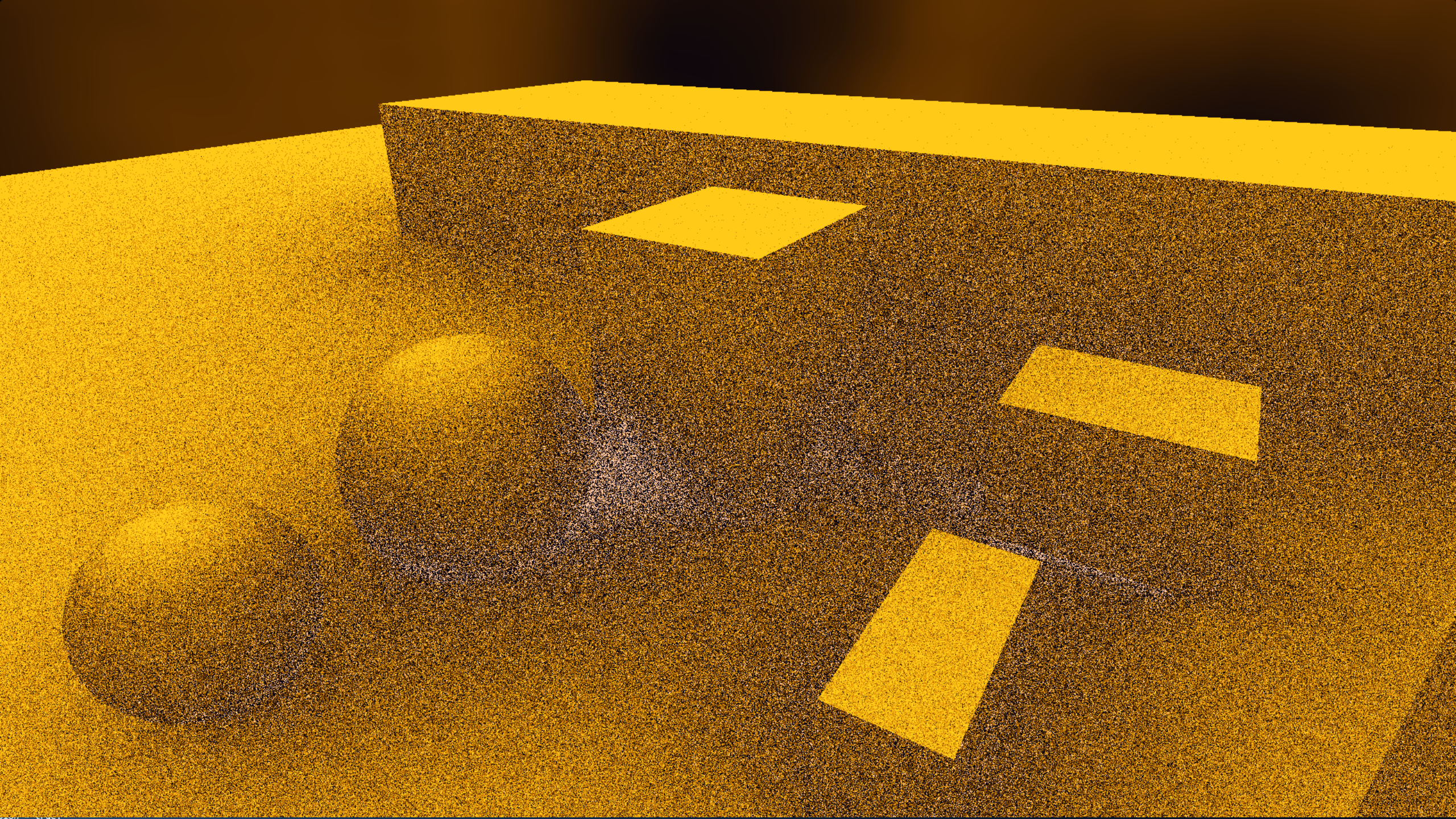
Or: *reprojection.*

“where was pixel x,y in the previous frame?”

https://www.shadertoy.com/view/ldtGWl








Well-converged:
Renderer was able to use reprojected data.
Noisy: surface was
recently disoccluded.
Noisy: surface was
recently disoccluded.

Reprojection

Practical Reprojection

Converging requires a stationary camera. Or: *reprojection.*

MPEG2 approach*:

Using the color of pixel (x,y), search the neighborhood for a similar color.
(problem: requires color of pixel (x,y), which (for path tracing) is noisy in the current and the previous frames).

Alternative, using additional data: using the world space position of the primary intersection point, search the neighborhood for the same position.
(we need to store a worldspace position per pixel)
(we may need to store a worldspace positions per sample)

*: New Fast Binary Pyramid Motion Estimation for MPEG-2 and HDTV Encoding, Song et al., IEEE, 2000.

Reprojection

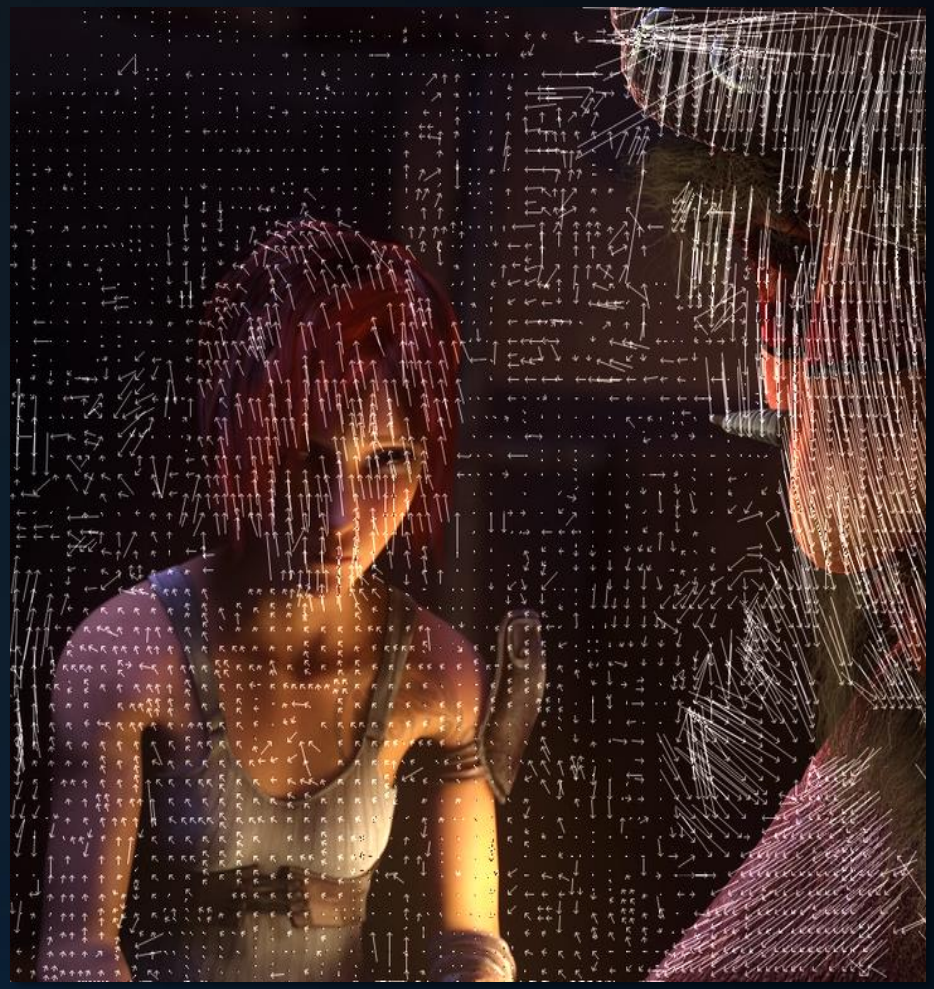
Practical Reprojection

Rasterization approach*:

In the vertex shader, transform the vertex with two matrices:

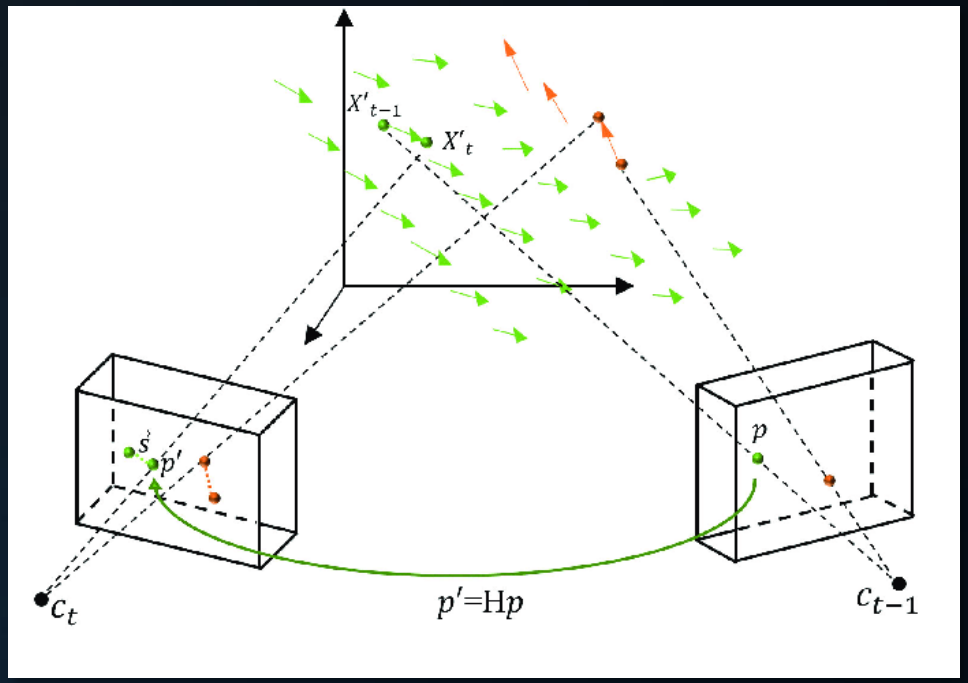
1. The current model/view/projection matrix;
2. The MVP matrix of the previous frame.

The difference is a (linear) *motion vector* per vertex. These are then interpolated over the triangle, yielding a motion vector per pixel.



*: Stupid OpenGL Tricks, Simon Green, NVIDIA, GDC2003 *(sorry, best reference I could find…).*

Reprojection



Practical Reprojection

Poor man's approach:

Assume a fully static scene. Then:

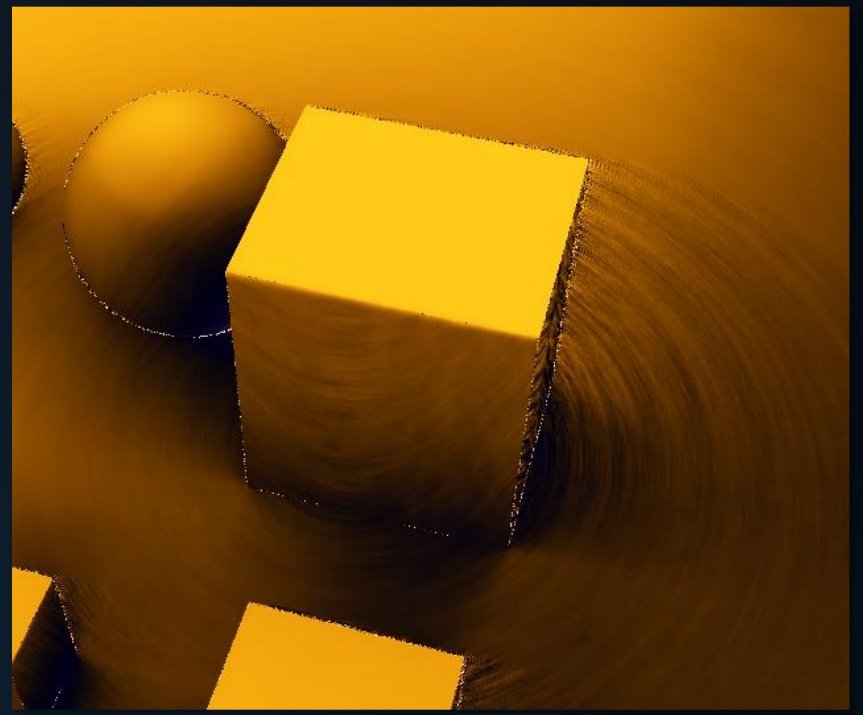
1. Multiply the camera space primary intersection position by the inverse camera matrix;
2. Multiply the result by the camera matrix of the previous frame.

Or:

Use the planes of the view pyramid to determine where pixel (x,y) would have been in the previous frame.

(this method is implemented in Lighthouse 2 and the Voxel World Template).

Reprojection



Challenges

Sample convergence over many frames:

Keep a running average*:

$$f_n(p) = \alpha \cdot s_n(p) + (1 - \alpha) \cdot f_{n-1}\big(\pi(p)\big)$$

where

$f_n(p)$ is frame n's color output at pixel p,
α is the blending factor (~0.1),
$s_n(p)$ is frame n's new sample color at pixel p,
$f_{n-1}\big(\pi(p)\big)$ is the reprojected history color from the previous frame.

*: A Survey of Temporal Antialiasing Techniques, Yang et al., 2020

Reprojection

Challenges

Reprojection may fail in several cases:

1. In the previous frame, pixel (x,y) was off-screen.
2. In the previous frame, pixel (x,y) was occluded.
3. Pixel (x,y) is on a specular surface, which is a view-dependent BRDF.
4. Pixel (x,y) was in the shadow of a moving light in the previous frame, now it isn't.
5. ...

Solutions:

Pragmatic. Case 1: drop the sample, we can't average with it. All other cases: *clip*.*

*: A Survey of Temporal Antialiasing Techniques, Yang et al., 2020

Reprojection

Further Reading

THE CODE CORSAIR

AHOY WORLD!

HOME BLOG CODING ABOUT

https://www.elopezr.com/temporal-aa-and-the-quest-for-the-holy-trail

TEMPORAL AA AND THE QUEST FOR THE HOLY TRAIL

BY ADMIN JANUARY 2, 2022 GRAPHICS

Long gone are the times where Temporal AA was a novel technique, and slowly more articles appear covering motivations, implementations and solutions. I will throw my programming hat into the ring to walk through it, almost like a tutorial, for the future me and for anyone interested. I am using Matt Pettineo's MSAAFilter demo to show the different stages. The contents come mostly from the invaluable work of many talented developers, and a little from my own experience. I will introduce a couple of tricks I have come across that I haven't seen in papers or presentations.

Sources of aliasing

The origin of aliasing in CG images varies wildly. Geometric (edge) aliasing, alpha testing, specular highlights, high frequency normals, parallax mapping, low resolution effects (SSAO, SSR), dithering and noise all conspire to destroy our visuals. Some solutions, like hardware MSAA and screen space edge detection techniques, work for a subset of cases but fail in different ways. Temporal techniques attempt to achieve supersampling by distributing the computations across multiple frames, while addressing all forms of aliasing. This stabilizes the image but also creates some challenging artifacts.

Jitter

The main principle of TAA is to compute multiple sub-pixel samples across frames, then combine those together into a single final pixel. The simplest scheme generates random samples within the pixel, but there are better ways of producing fixed sequences of samples. A short overview of quasi-random sequences can be found here. It is important to select a

SEARCH

POSTS

Temporal AA and the quest for the Holy Trail January 2, 2022

The Rendering of Mafia: Definitive Edition August 23, 2021

A Macro View of Nanite May 30, 2021

The Rendering of Jurassic World: Evolution March 12, 2021

Rendering Line Lights July 10, 2019

The Rendering of Rise of the Tomb Raider December 31, 2018

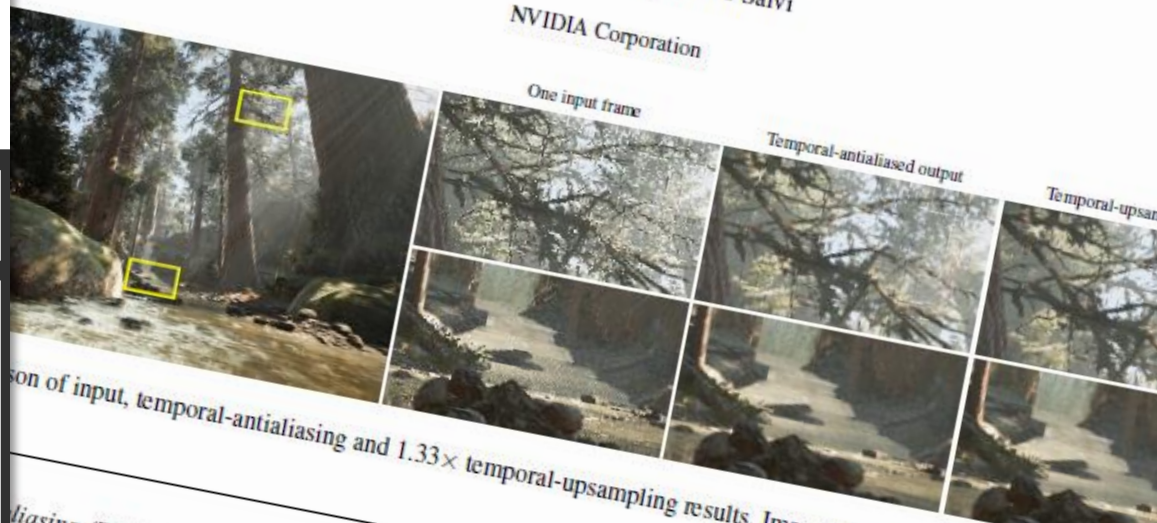
A real life pinhole camera January 14, 2018

The Rendering of Middle Earth:

A Survey of Temporal Antialiasing Techniques

Lei Yang, Shiqiu Liu, Marco Salvi

NVIDIA Corporation



son of input, temporal-antialiasing and 1.33× temporal-upsampling results. Images are rendered by Unreal Engine 4

liasing (TAA), formally defined as temporally-amortized supersampling, is the most widely used antialiasing ay's real-time renderers and game engines. This survey provides a systematic overview of this technique. We istory of TAA, its development path and related work. We then identify the two main sub-components of TAA, tion and history validation, and discuss algorithmic and implementation options. As temporal upsampling is ingly relevant to today's game engines, we propose an extension of our TAA formulation to cover a pling techniques. Despite the popularity of TAA, there are still significant unresolved uality in many scenarios. We provide an in-depth analysis of these challe Finally, we summarize popular algorithms and topics th eas may either benefit from or feedback


https://www.shadertoy.com/view/Wt3XRX

Today’s Agenda:

- Prerequisits
- Reprojection
- Resampling
- ReSTIR

ReSTIR, ingredient 2: Resampling

Resampling

Importance Sampling Lights

Sampling N lights with 1 ray:

- Use a constant discrete pdf: $\rho_i = \frac{1}{N}$, where $i \in [1..N]$.

Or:

- Use importance sampling.

But, what is the importance of each light?
We can use the *potential contribution** of each light:

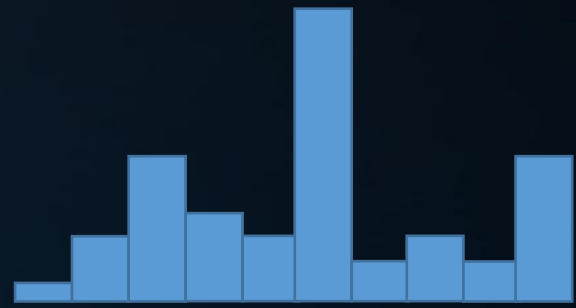
$$L_p = E \frac{A \cos\theta_i \cos\theta_o}{d^2}$$

That is: the emittance of the light, scaled by its solid angle, as seen from a point in the scene.

*: Note: the potential contribution may differ significantly from the actual contribution!

Note:

- We calculate the potential contribution for each light, at each path vertex.
- We create a discrete *cdf* over the lights (CDF: *cumulative distribution function).*
- To pick one light, we need to walk the cdf a second time.



N=10:	1 3 6 4 3 11 2 4 2 6
$\sum L_p$:	42
$\rho_i =$	$\{ 1/42,\ 3/42,\ 6/42, \ldots \}$

Resampling

Resampling

From Multiple to *Many* Lights

Challenge:

Picking a light with a probability proportional to its potential contribution requires evaluation of all lights, i.e.:

- Fetching light properties from memory;
- Calculations: includes $1/r^2$, $\cos\theta$, BRDF evaluation;
- Storing the result in the cdf array.

If we have more than a dozen or so lights, this is not feasible.

Solution: *precalculate potential contribution?*

Resampling

From Multiple to *Many* Lights

Potential contribution is proportional to:

- Solid angle
- Brightness of the light

Sadly, we cannot pre-calculate potential contribution; it depends on the location and orientation of the light source relative to the point we are shading.

We can however precalculate a less refined potential contribution based on:

- Area
- Brightness

Resampling

Many Lights Array

The light array stores pointers to (or indices of) the lights in the scene.
For N lights, light array size M is several times N.

Each light occupies several consecutive slots in the light array, proportional to its (coarse) potential contribution.

Selecting a random slot in the light array now yields (in constant time) a single light source L, with a probability of $\frac{slots\ for\ L}{M}$.

Resampling

Resampled Importance Sampling*

The light array allows us to pick a light source proportional to importance.
However, this importance is not very accurate.

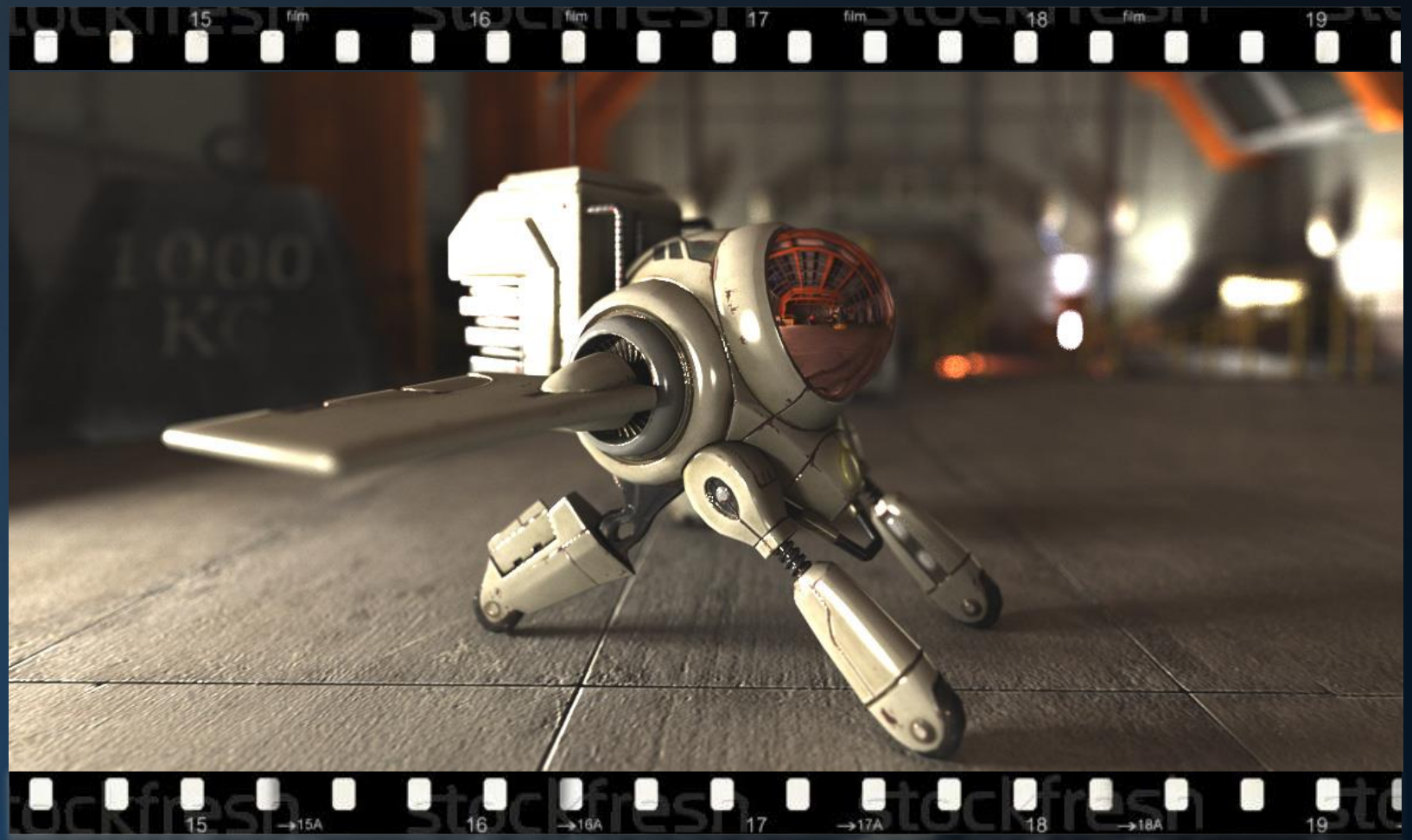
We can improve our choice using *resampled importance sampling*.

1. Pick *N* lights from the light array (where *N* is a small number, e.g. 4);
2. For each of these lights, determine the more accurate potential contribution;
3. Translate the potential contribution to importance (using a small cdf);
4. Choose a light with a probability proportional to this importance.

This scheme allows for unbiased, accurate and constant time selection of a good light source.

*: Importance Resampling for Global Illumination, Talbot et al., 2005.

Resampling



https://www.youtube.com/watch?v=Znr1JJLI5uY

Today’s Agenda:

- Prerequisits
- Reprojection
- Resampling
- ReSTIR

ReSTIR

ReSTIR*

Ingredient 1:
Reprojection

Ingredient 2:
Resampling.

Ingredient 3:
Streaming RIS.

*: Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting, Bitterli et al., 2020.

ReSTIR

ReSTIR* - Direct Light from a Very Large Number of Lights

Algorithm, in short:

Sample millions of lights for a pixel with one ray to an important light.

To pick the important light:

1. Use RIS;
2. Consider the knowledge of the neighbors;
3. Consider the knowledge of the past.

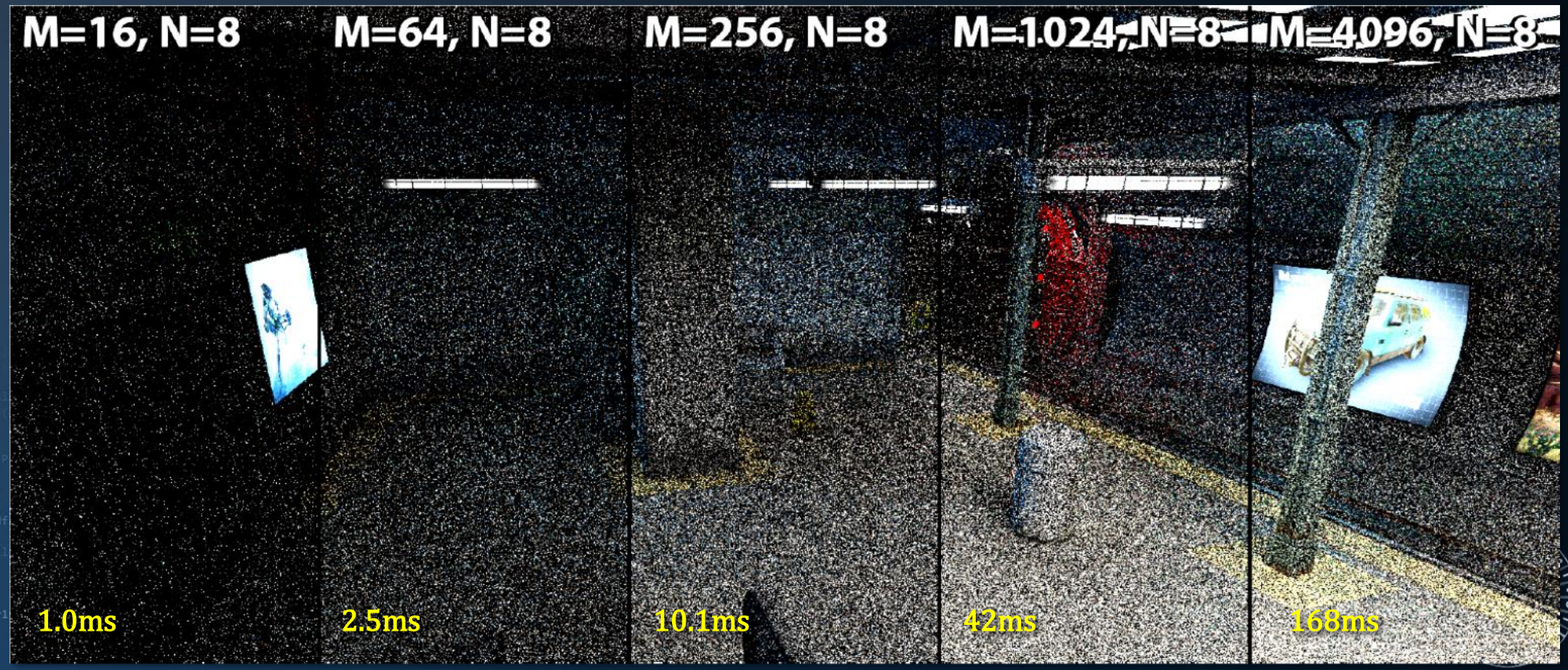
Hence: *spatio-temporal resampling*.

*: Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting, Bitterli et al., 2020.

ReSTIR

ReSTIR* - Direct Light from a Very Large Number of Lights

Applying RIS:



ReSTIR

Weighted Reservoir Sampling

Recall:

Selecting a light with a probability proportional to its potential contribution:

1. Calculate and store the weight of each light
2. Calculate the sum S of the weights
3. Draw a random number in the range $[0..S)$
4. Walk the array of stored weights until the running sum exceeds S.

Problems:

- Storage;
- Visiting the data twice.

ReSTIR

Weighted Reservoir Sampling

Alternative approach:

```
y = 0, w_sum = 0
for each light index i with weight w_i
      w_sum += w_i
      if (rand() < w_i / w_sum) y = i;
```

After processing all lights, y contains the chosen light.

- No storage
- Data is visited only once

There is a third, somewhat hidden benefit:

- After choosing the light, we still have the reservoir state in y and w_{sum}.

ReSTIR

Back to ReSTIR

Combining the ingredients:

1. Pass 1: for each pixel, use reservoir sampling to pick a light.

Result: a reservoir state, consisting of light index y, and w_{sum}.

2. Pass 2: for each pixel, combine reservoirs of neighborhood.

Result: neighborhood combines forces to pick much better lights.

3. And finally: store final reservoirs for the next frame.

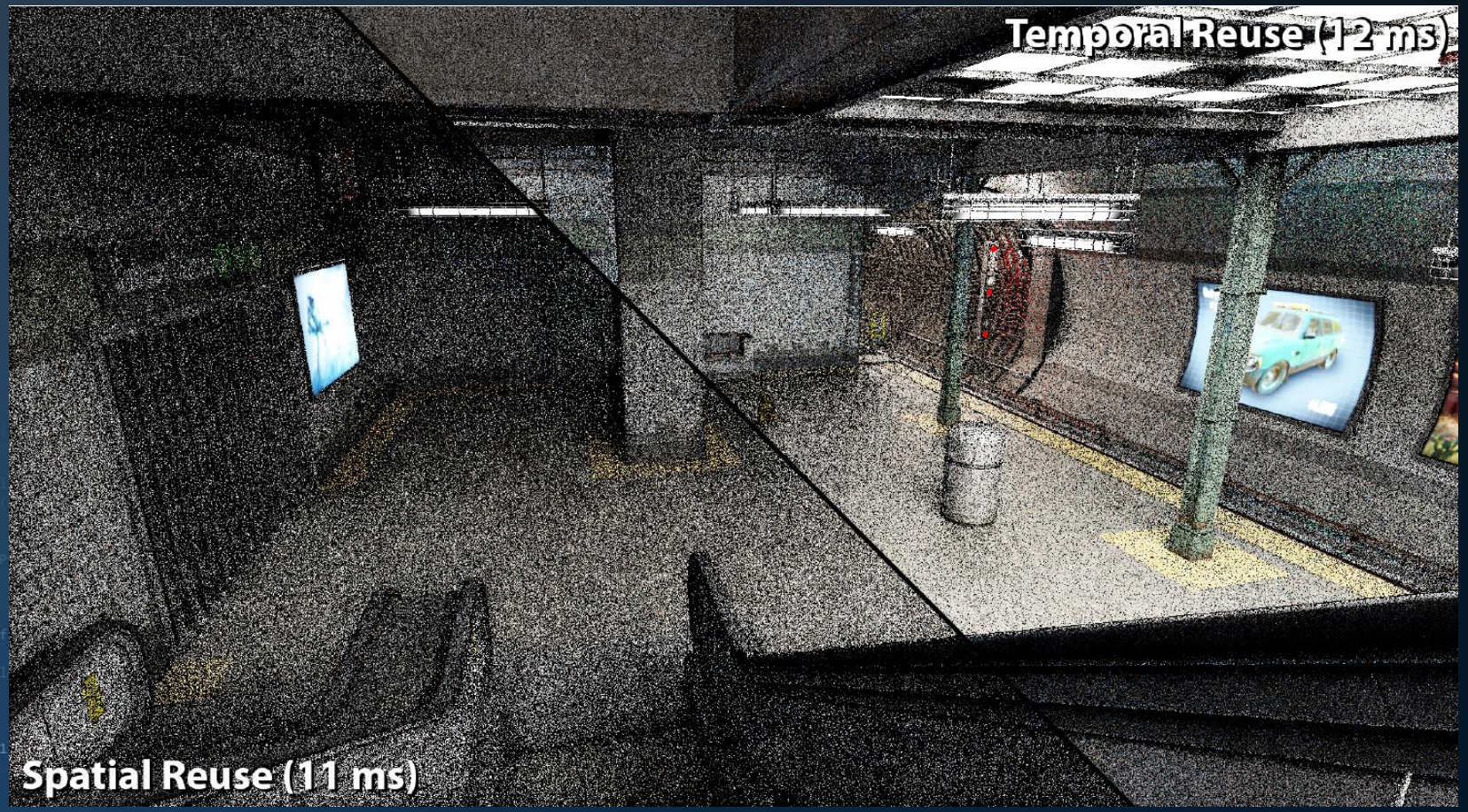
Result: good picks of previous frames become candidates in the current frame.

The light to which we finally send a shadow ray to is effectively taken from a massive pool of candidates.

ReSTIR

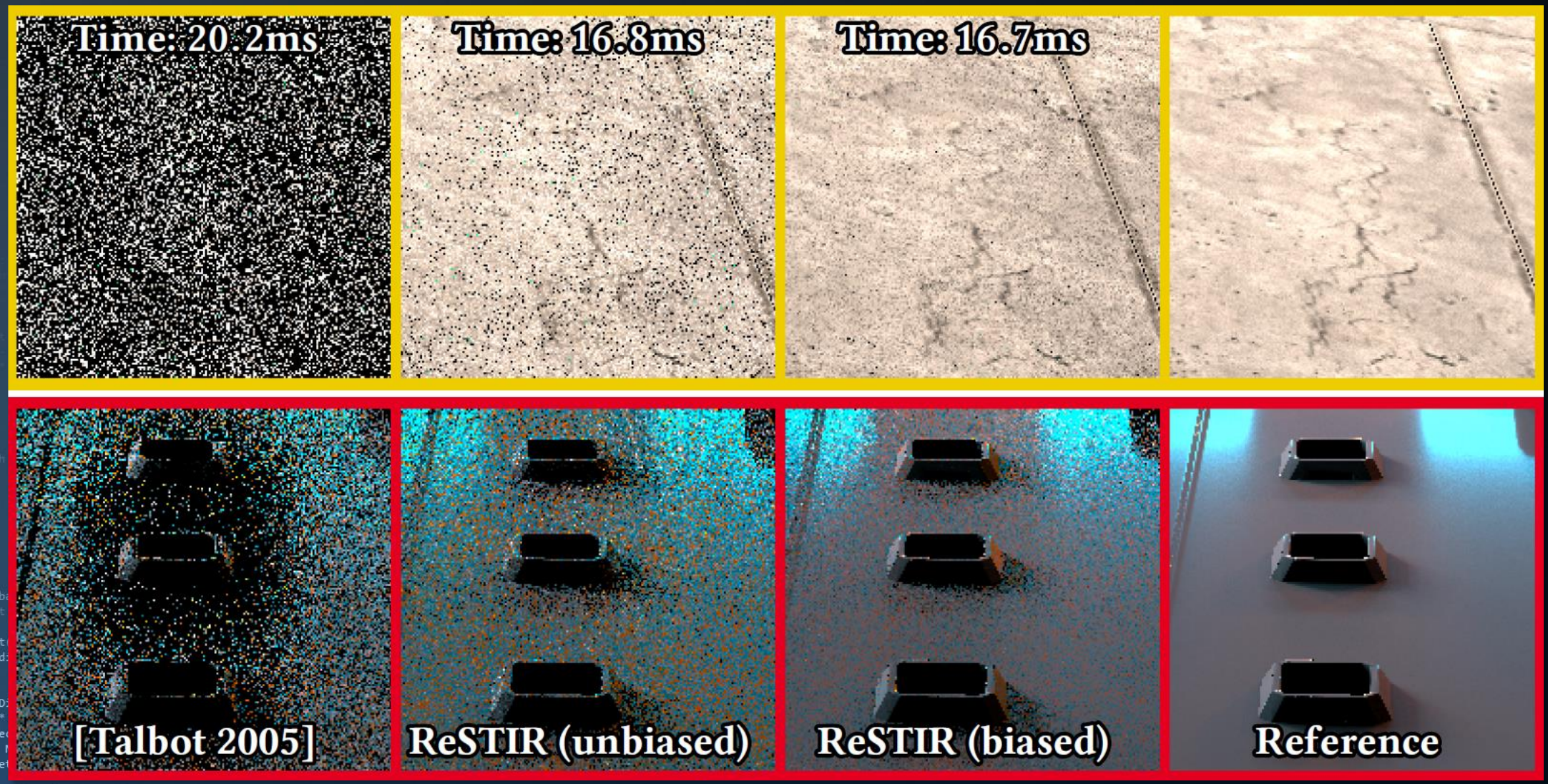
ReSTIR* - Direct Light from a Very Large Number of Lights

Neighbors & the past:



Today’s Agenda:

- Prerequisits
- Reprojection
- Resampling
- ReSTIR



There's still noise… now what?

INFOMAGR – Advanced Graphics

Jacco Bikker - November 2021 - February 2022

END of “ReSTIR”

next lecture: “Filtering & Learning GI”