Lei Yang, Dmitry Zhdan, Emmett Kilgariff,
Eric B. Lum, Yubo Zhang, Matthew Johnson,
and Henrik Rydgård

NVIDIA

VISUALLY LOSSLESS CONTENT AND MOTION ADAPTIVE SHADING IN GAMES

ACM I3D, May 21-23 2019 - Montréal, Canada

OUTLINE

Background

Content adaptive shading

- Theory & Results

Motion adaptive shading

- Theory & Results

Demo: Wolfenstein II: the New Colossus

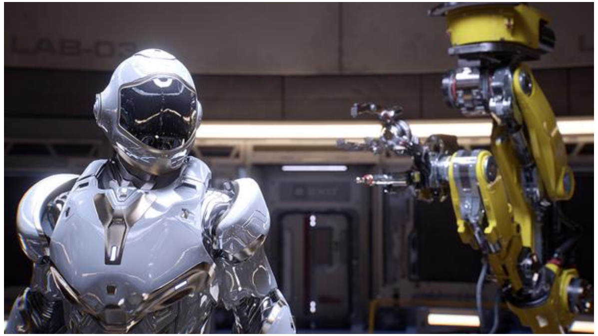
BACKGROUND

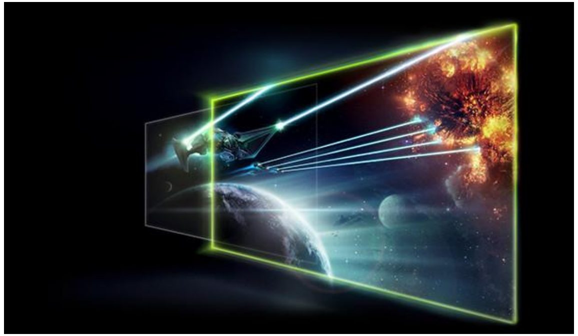
OVERVIEW

Why do we need adaptive shading?

The cost of pixel shading increases dramatically in today’s games

- Enhanced realism and special effects
- Higher resolution and framerate





OVERVIEW

Pixel Shading Inefficiencies

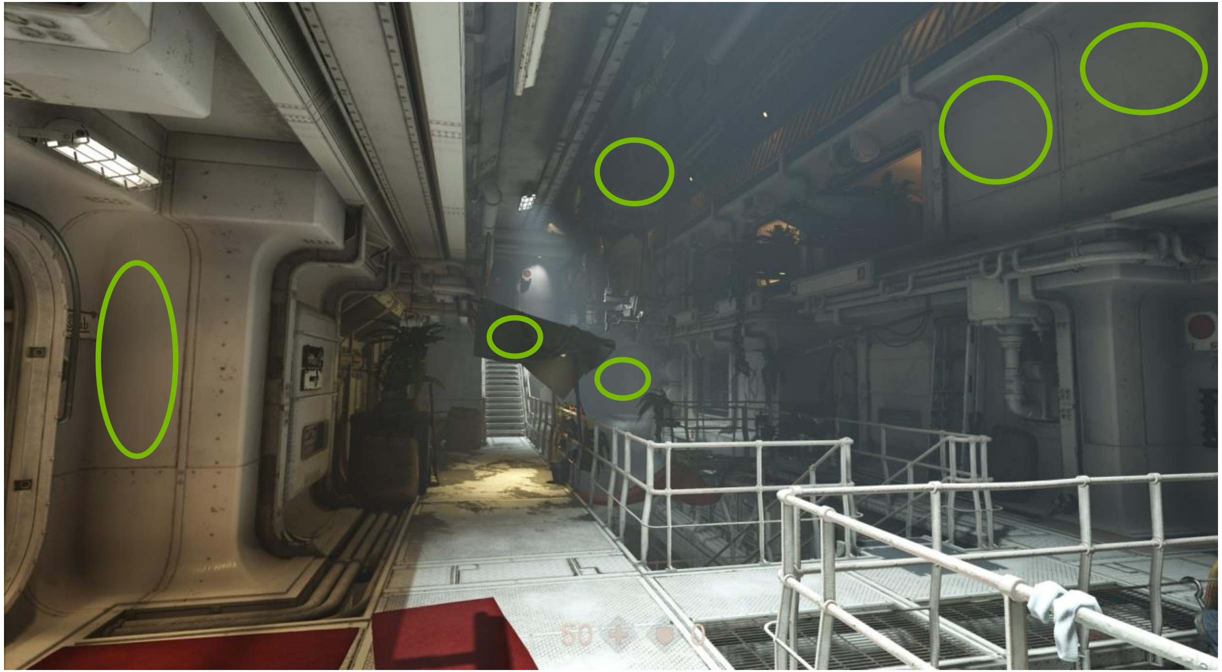
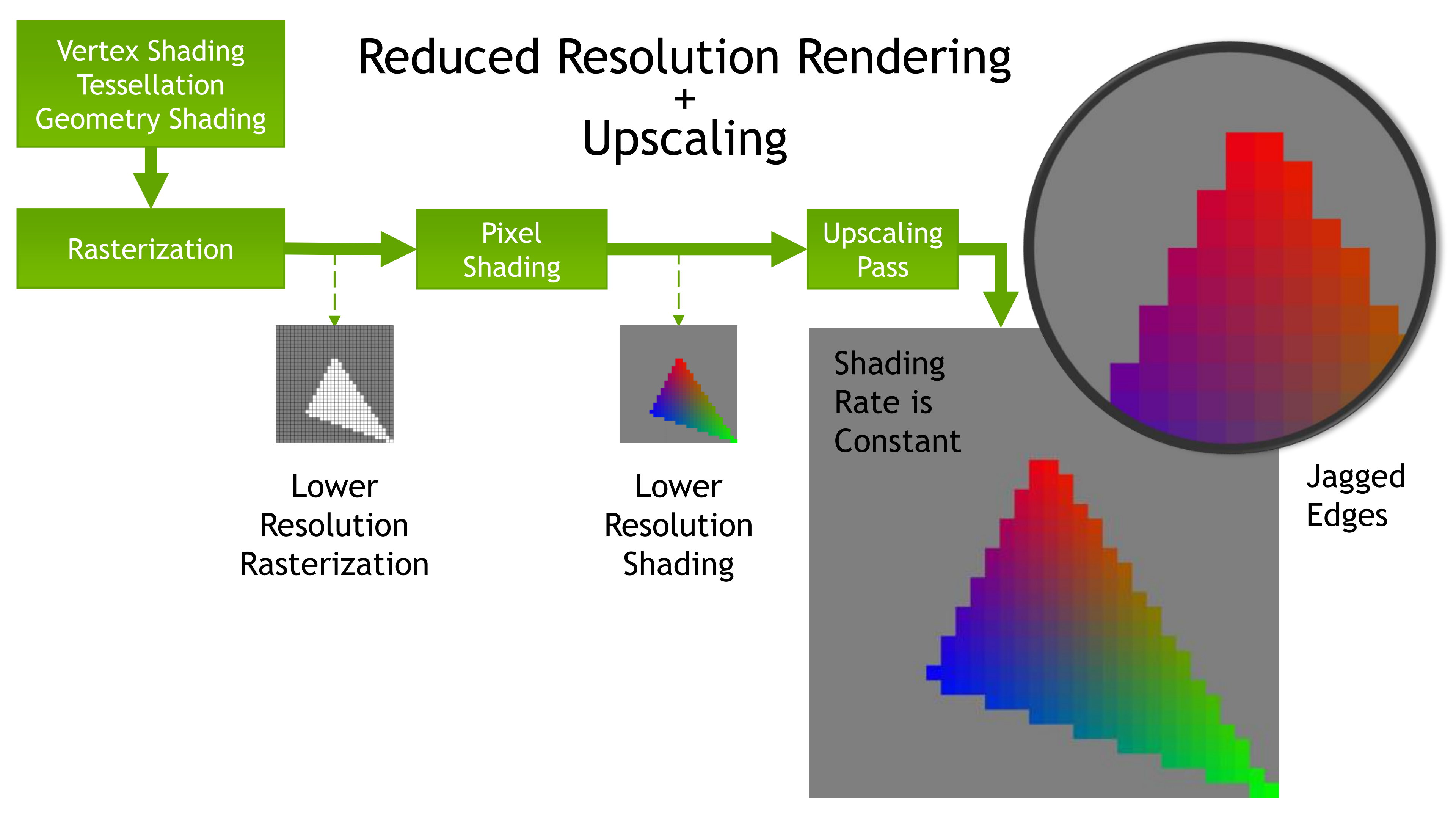


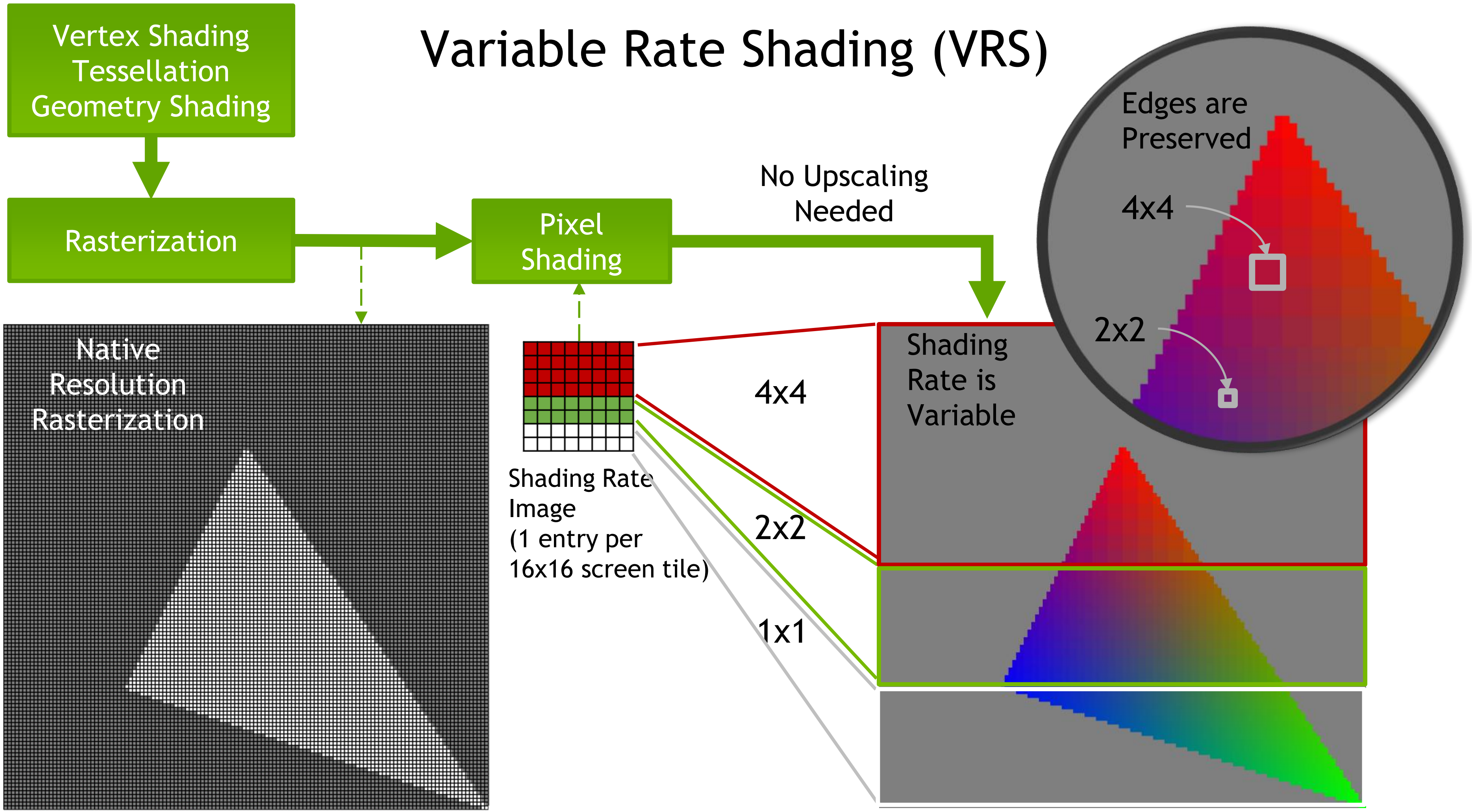
Image from Wolfenstein II: the New Colossus

Shader gets run every single pixel, regardless of content



Reduced Resolution Rendering
+
Upscaling
Vertex Shading
Tessellation
Geometry Shading
Rasterization
Pixel Shading
Upscaling Pass
Lower Resolution Rasterization
Lower Resolution Shading
Shading Rate is Constant
Jagged Edges

Variable Rate Shading (VRS)

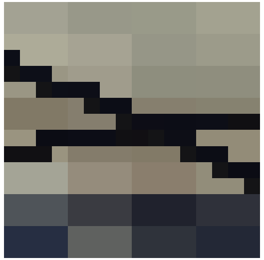


VARIABLE RATE SHADING (VRS)

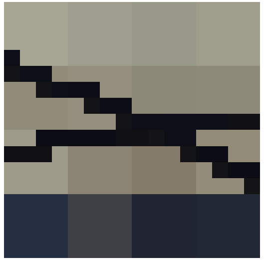
Shading rate options

1x1

2x1



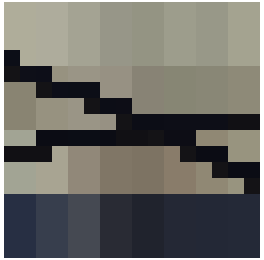
4x2



4x4

1x2

2x2



2x4

MAIN CONTRIBUTIONS

Questions we answered in this paper

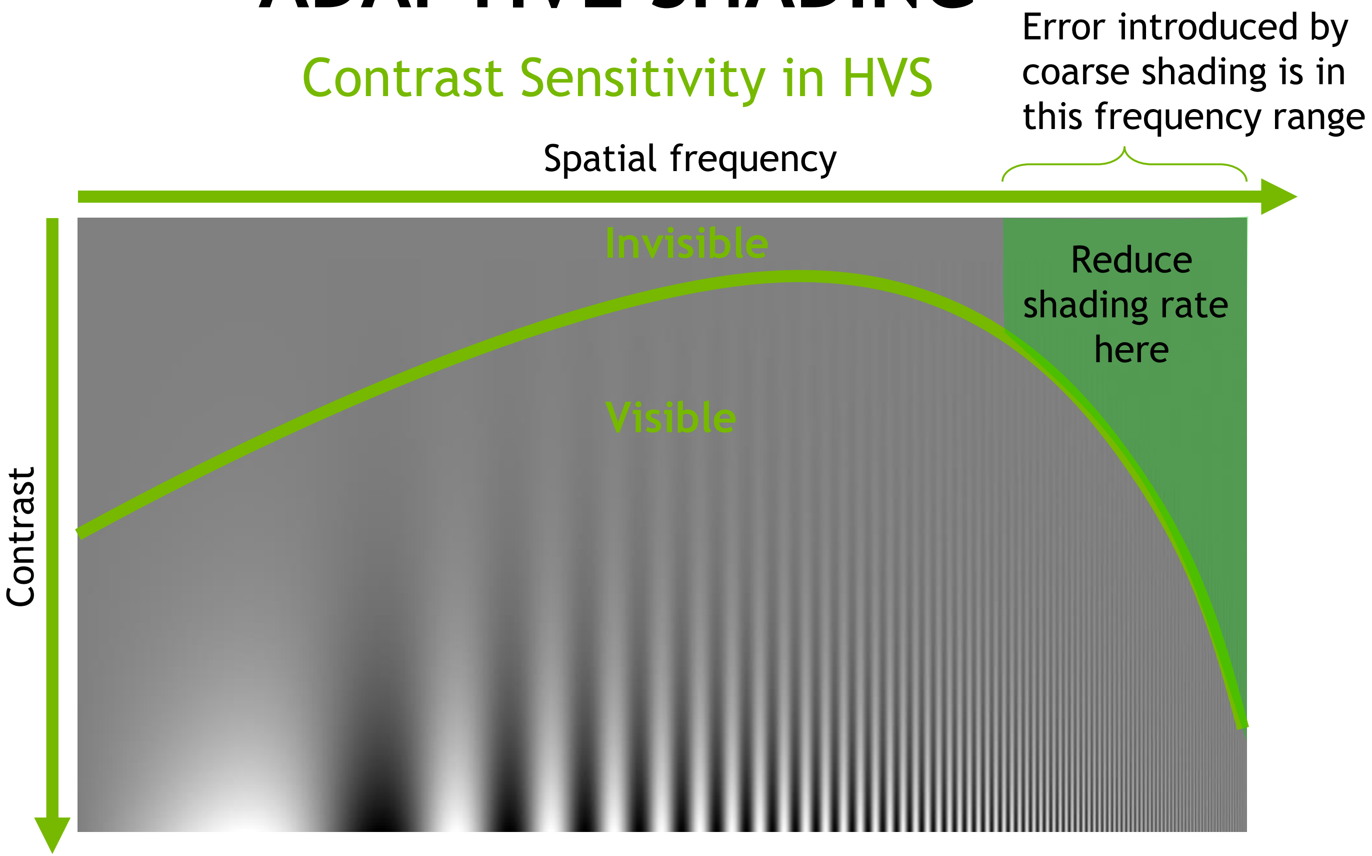
1. **What is the error caused by lowering shading rate?**
2. **How is visual error affected by motion?**
3. **How to implement adaptive shading efficiently in games?**

~~Heuristics~~

CONTENT ADAPTIVE SHADING

ADAPTIVE SHADING

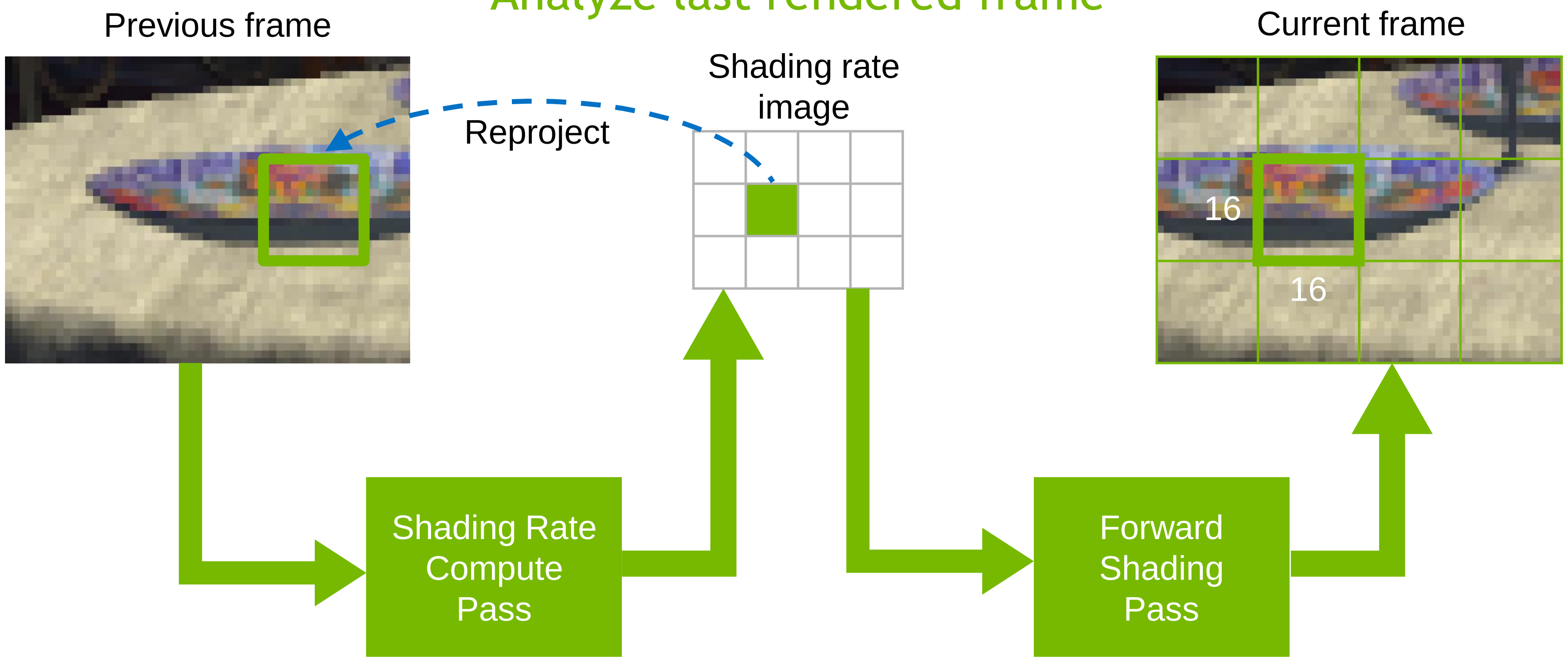
Contrast Sensitivity in HVS



Campbell-Robson CSF chart
Courtesy of Izumi Ohzawa

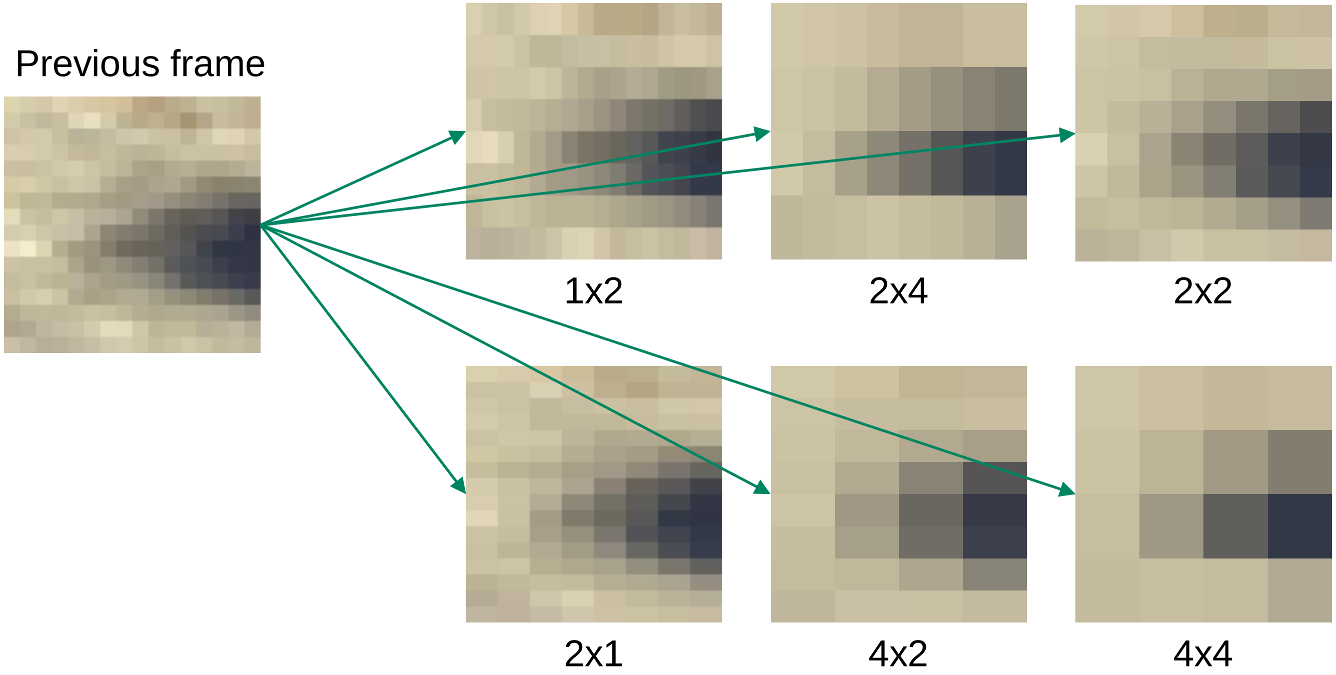
ADAPTIVE SHADING FLOW

Analyze last rendered frame



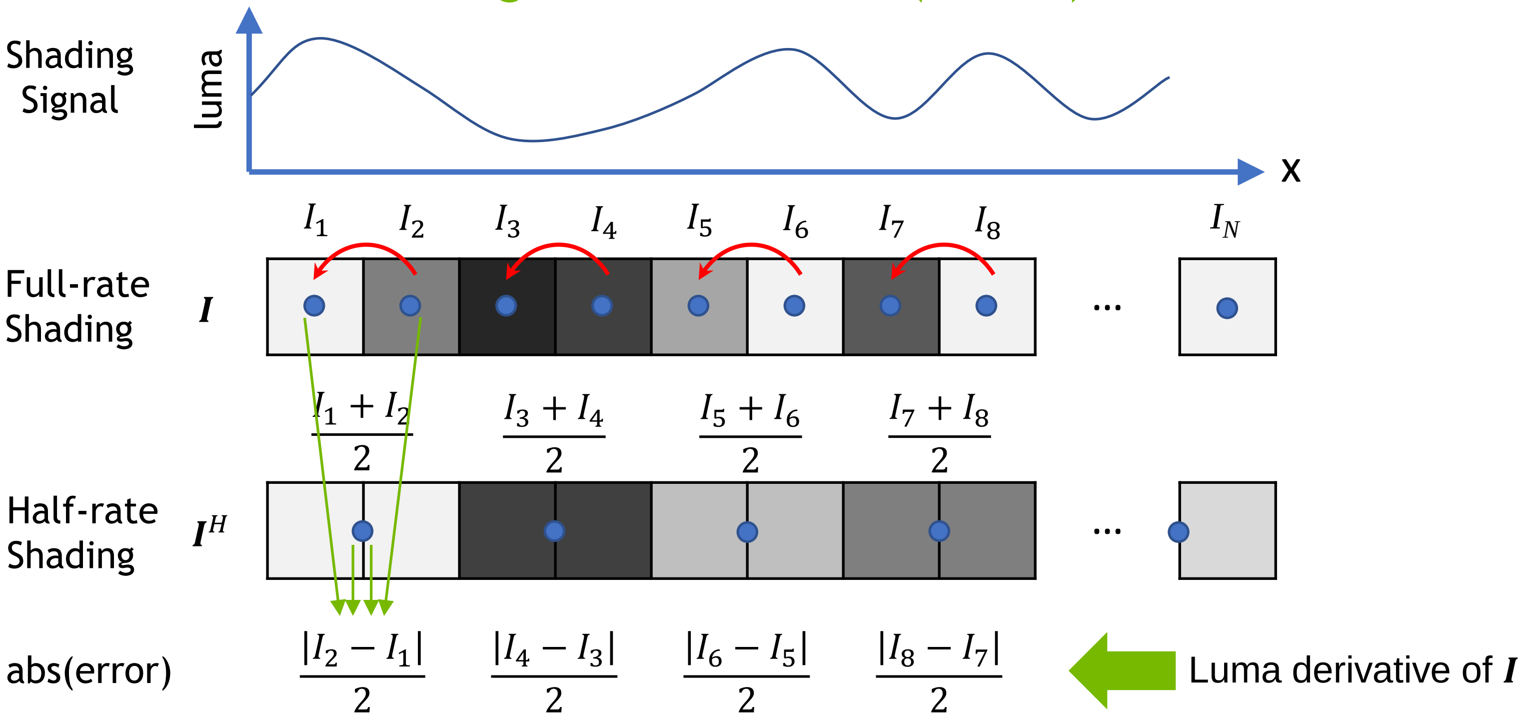
CONTENT ADAPTIVE SHADING

Which shading rate to use? Tell me the error



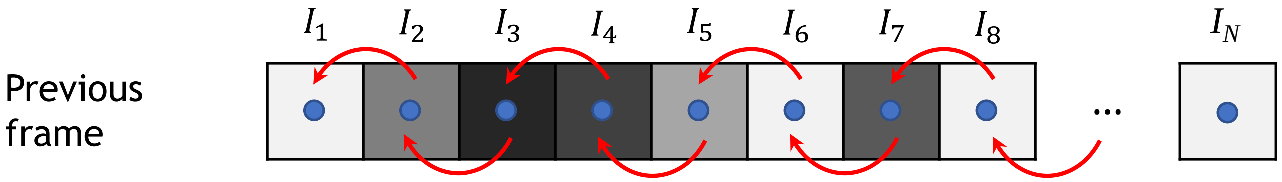
CONTENT ADAPTIVE SHADING

Deriving the error in 1D (X or Y)



CONTENT ADAPTIVE SHADING

Estimate the error per tile



$\mathcal{E}(\boldsymbol{I}, \boldsymbol{I}^H) = \|\boldsymbol{I} - \boldsymbol{I}^H\|_2 \cong \sqrt{\sum_N (I_i - I_{i-1})^2} = \|\boldsymbol{I} * \boldsymbol{D}\|_2$ ($\boldsymbol{D}$ is the differencing kernel $[-0.5, 0.5]$)

Instead of using $(I_2 - I_1,\ I_4 - I_3,\ \ldots)$, we involve all pairs $(I_2 - I_1,\ I_3 - I_2,\ I_4 - I_3,\ \ldots)$

- Stabilized result when image is shifted by 1 pixel
- More robust estimation when using previous frame instead of true signal

CONTENT ADAPTIVE SHADING

Error estimate for quarter-rate shading

Full-rate shading

I

Half-rate shading

I^H

Quarter-rate shading

I^Q

$$\mathcal{E}(I, I^H) = \|I - I^H\|_2$$ (Mean-squared error)

$$\cong \|I * D\|_2$$ (D: differencing kernel)

$$= \|\mathcal{F}(I) \cdot \mathcal{F}(D)\|_2$$ (Convolution and Parseval theorems) (omitting const. $1/\sqrt{N}$)

$$= \|\mathcal{F}(I) \cdot (\mathbf{1} - \mathcal{F}(B_2))\|_2$$ (B_2: box kernel ½[1,1])

$$\mathcal{E}(I, I^Q) = \|\mathcal{F}(I) \cdot (\mathbf{1} - \mathcal{F}(B_4))\|_2$$ (B_4: box kernel ¼[1,1,1,1])

$$= \|\mathcal{F}(I) \cdot (\mathbf{1} - \mathcal{F}(B_2)) \cdot \mathcal{K}\|_2$$ (*K: frequency bin scaling*)

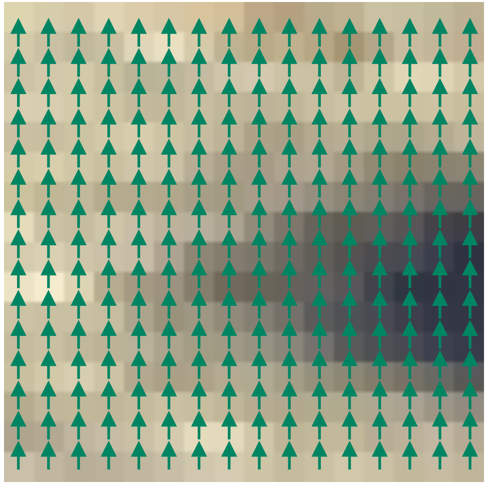
$$\cong k \cdot \|\mathcal{F}(I) \cdot (\mathbf{1} - \mathcal{F}(B_2))\|_2$$ (Pre-integration)

$$= k \cdot \|I * D\|_2$$

CONTENT ADAPTIVE SHADING

Determine shading rate

$$\mathcal{E}(I, I^H) \cong \|I \ast D\|_2$$



Independent decision for X and Y:

$\mathcal{E}(I, I^H) < \text{threshold ?}$

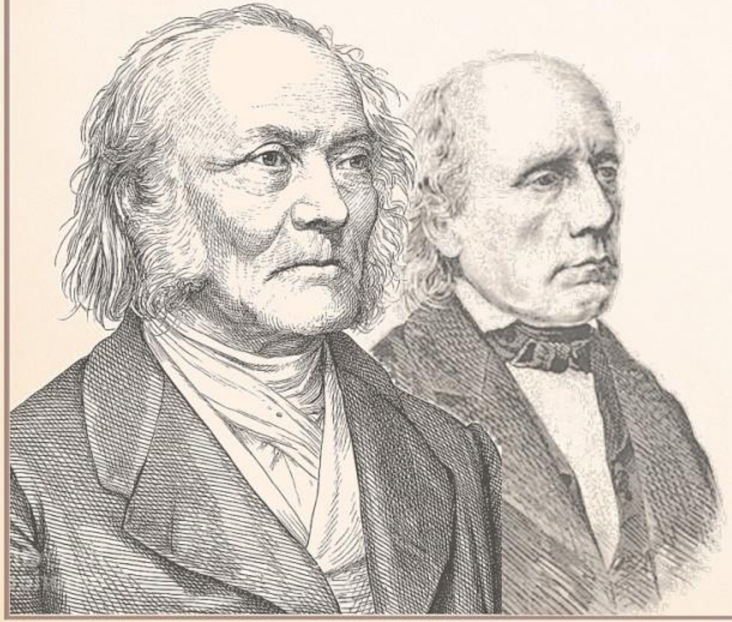
Enable half-rate shading

$\mathcal{E}(I, I^Q) = k \cdot \mathcal{E}(I, I^H) < \text{threshold ?}$

Enable quarter-rate shading

Use a “Just Noticeable Difference” (JND) threshold (Weber-Fechner Law)

WEBER-FECHNER LAW



"The just-noticeable difference between two stimuli is proportional to the magnitude of the stimuli"

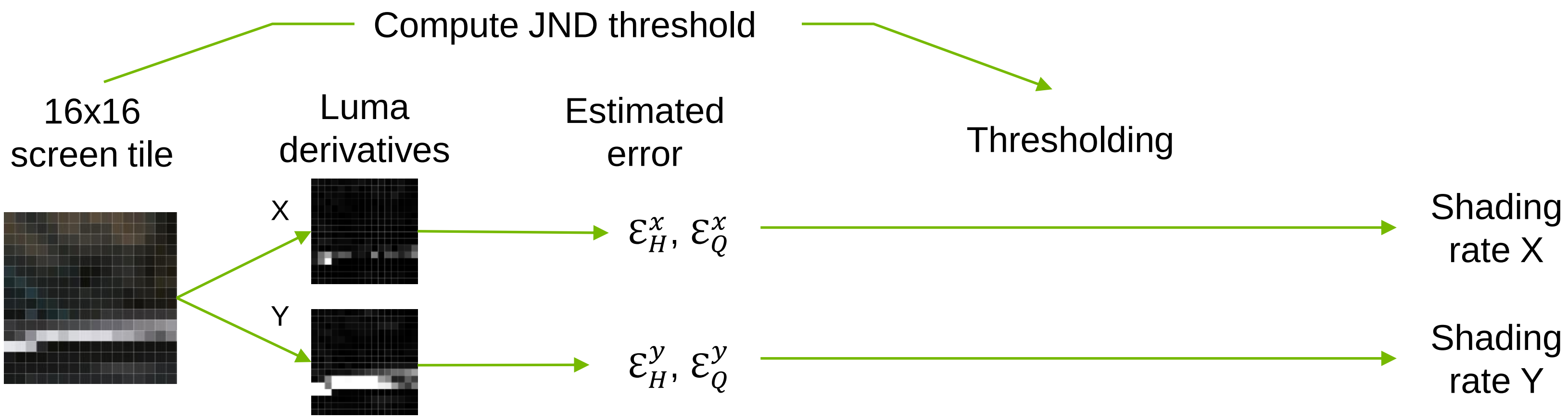
JND_THRESHOLD = SENSITIVITY_THRESHOLD * BASE_LUMINANCE

Image courtesy of Denys Mishunov

CONTENT ADAPTIVE SHADING

Back to 10,000 feet view

For each 16x16 screen tile reprojected from previous frame:



RESULTS

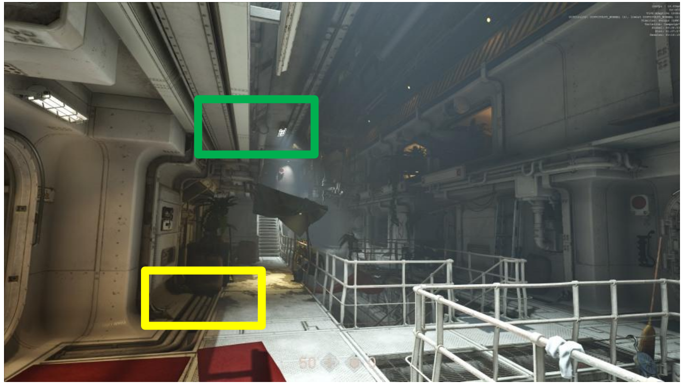
Roswell Scene

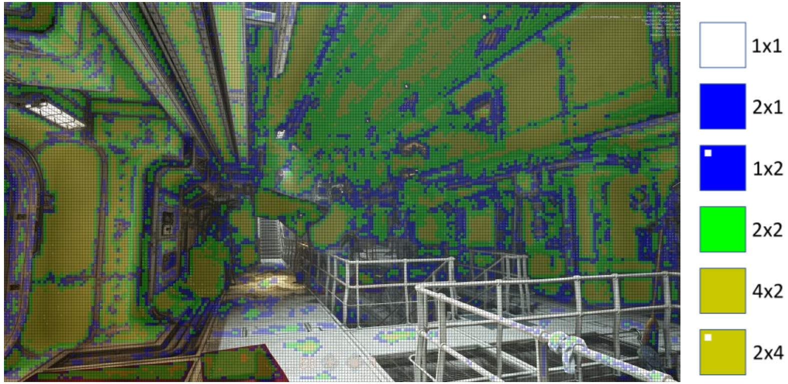
(Average shading rate: 0.55x)

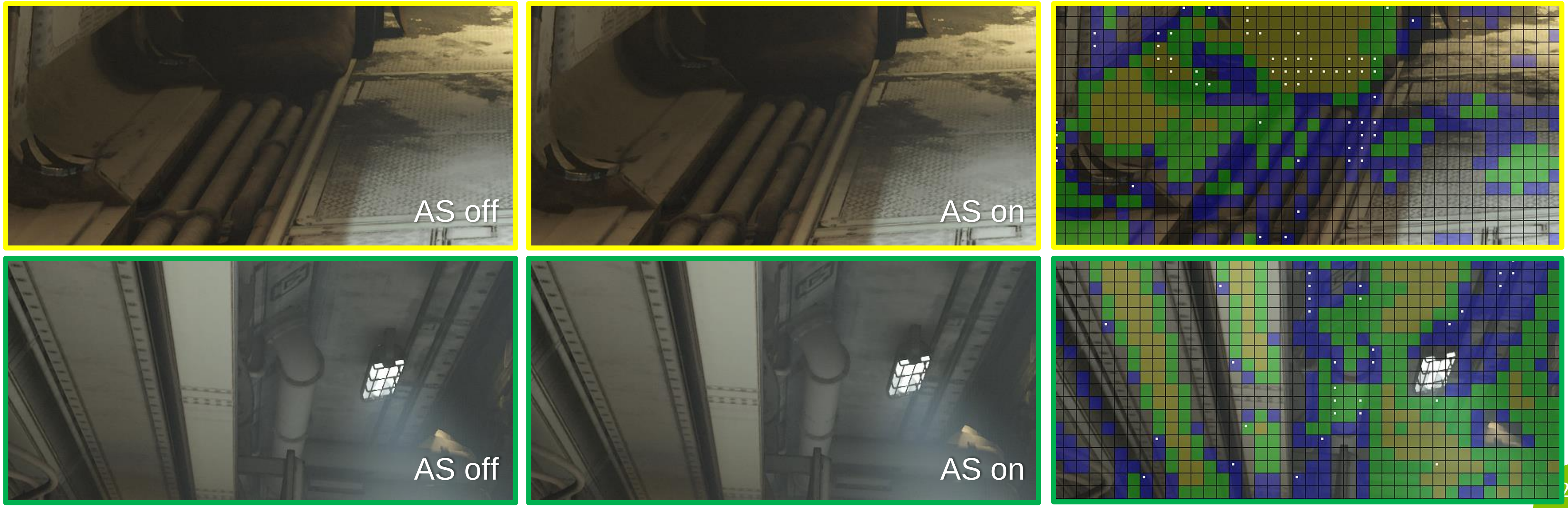
RESULTS

Submarine Scene

(Average shading rate: 0.41x)







MOTION ADAPTIVE SHADING

* Visit testufo.com
to learn more

LCD PERSISTENCE BLUR *

IN REAL WORLD

ON THE SCREEN

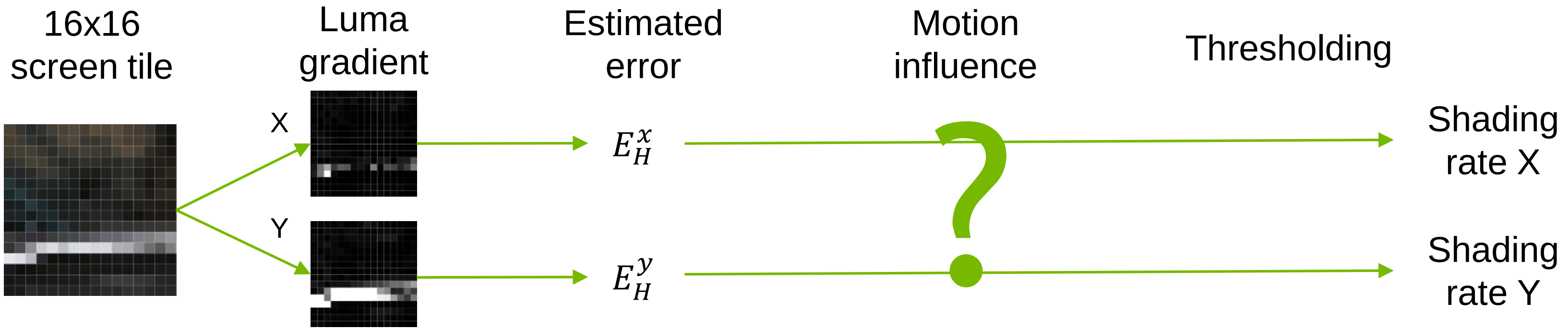
MOTION BLUR AND PERCEIVED ERROR

How motion blur hides VRS error

UFO image courtesy of testufo.com (Blur Busters UFO Motion Tests)

MOTION ADAPTIVE SHADING

Factor motion influence into adaptive shading



MOTION ADAPTIVE SHADING

Motion-based VRS error estimate scaler

Two scalar functions:

- For half-rate: $\mathcal{E}_H(v) = b_H(v) \cdot \mathcal{E}_H$
- For quarter-rate: $\mathcal{E}_Q(v) = b_Q(v) \cdot \mathcal{E}_H$

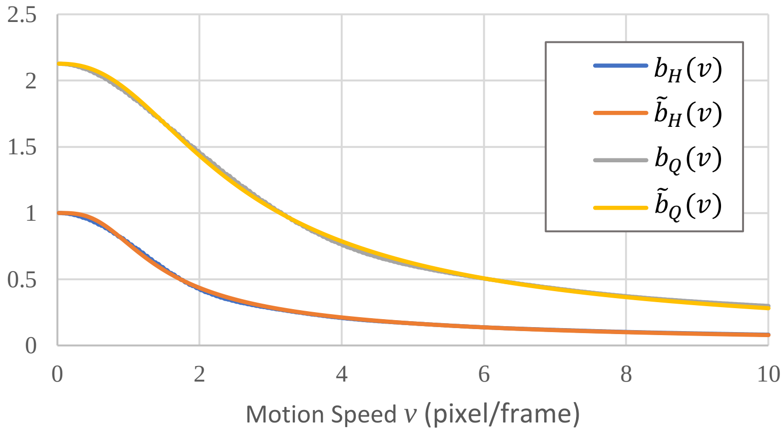
We compute $b_H(v)$ and $b_Q(v)$ numerically

Then fit two closed-form functions:

$$\tilde{b}_H(v) = \left(\frac{1}{1 + (1.05v)^{3.10}}\right)^{0.35},$$

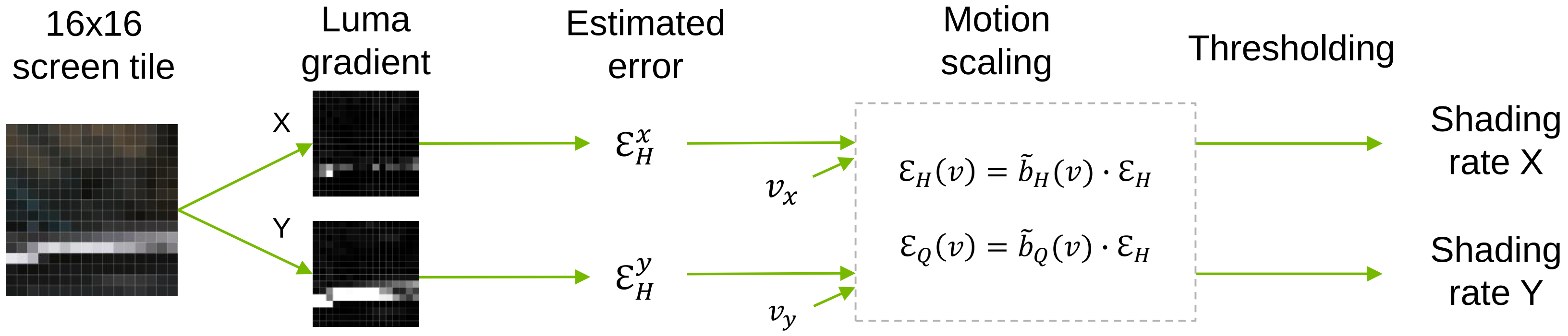
$$\tilde{b}_Q(v) = 2.13\left(\frac{1}{1 + (0.55v)^{2.41}}\right)^{0.49}.$$

(Details in the paper)



MOTION ADAPTIVE SHADING

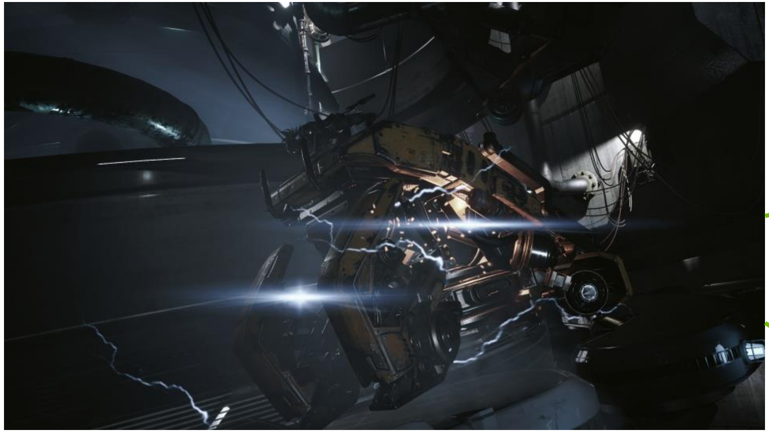
Unified content and motion adaptive shading



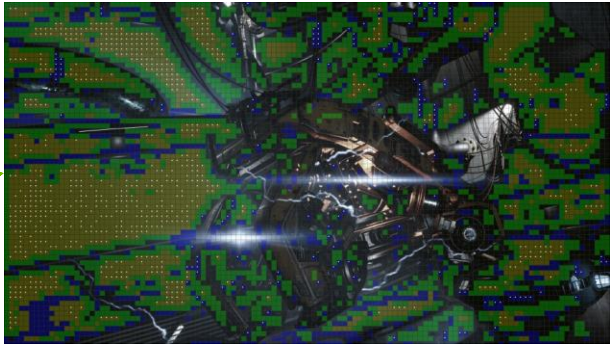
Per-tile motion speed v is the *minimum* motion vector across the tile

MOTION ADAPTIVE SHADING

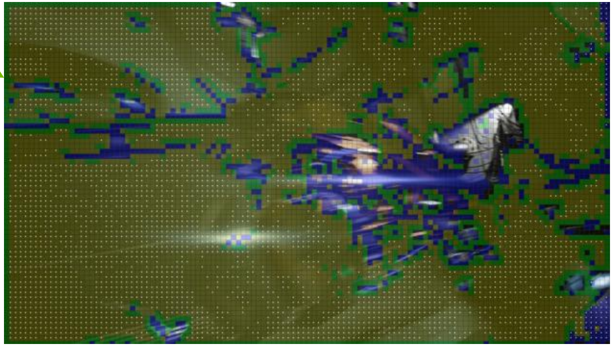
Effect on the shading rate pattern



Content Adaptive



Content + Motion Adaptive



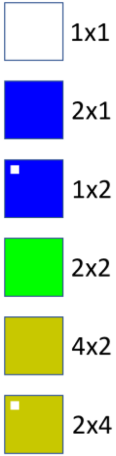


Image captured from “Infiltrator” demo rendered in Unreal Engine 4

ADAPTIVE SHADING PERFORMANCE

Overall cost and benefit

Engine change: add a few compute passes to generate shading rate image

- Simple API calls to enable VRS on forward shading passes
- Overhead: 0.2ms for 4K (2160p) on a RTX2080Ti, completely hidden in async compute

Performance benefit (UE4/id Tech 6) :

- 2x average, up to 5x performance gain in PS-bound forward passes
- 5% - 20% reduction in frame time (~1-3ms saving for 60FPS target)

MACHINEGAMES

WOLFENSTEIN II DEMO


109fps - 9.10ms
100%h, 100%v
Tick-Adaptive 109Hz
50
0

SUMMARY

Adaptive shading: now and future

- Use the new GPU feature: VRS - Minor change in engine, supported by all major APIs
- Model the shading error with frequency space analysis - say goodbye to old heuristics
- 2x average, up to 5x performance gain in forward shading passes
- Visually indistinguishable from full-rate shading

Future work: adaptive shading in non-forward shading passes

- Deferred & post-processing passes using compute pipeline
- Ray tracing

QUESTIONS?

Special thanks:

- Jim Kjellin and MachineGames studio
- Henry Moreton, Dale Kirkland, and others from NVIDIA GPU Arch & DevTech
- Anonymous reviewers