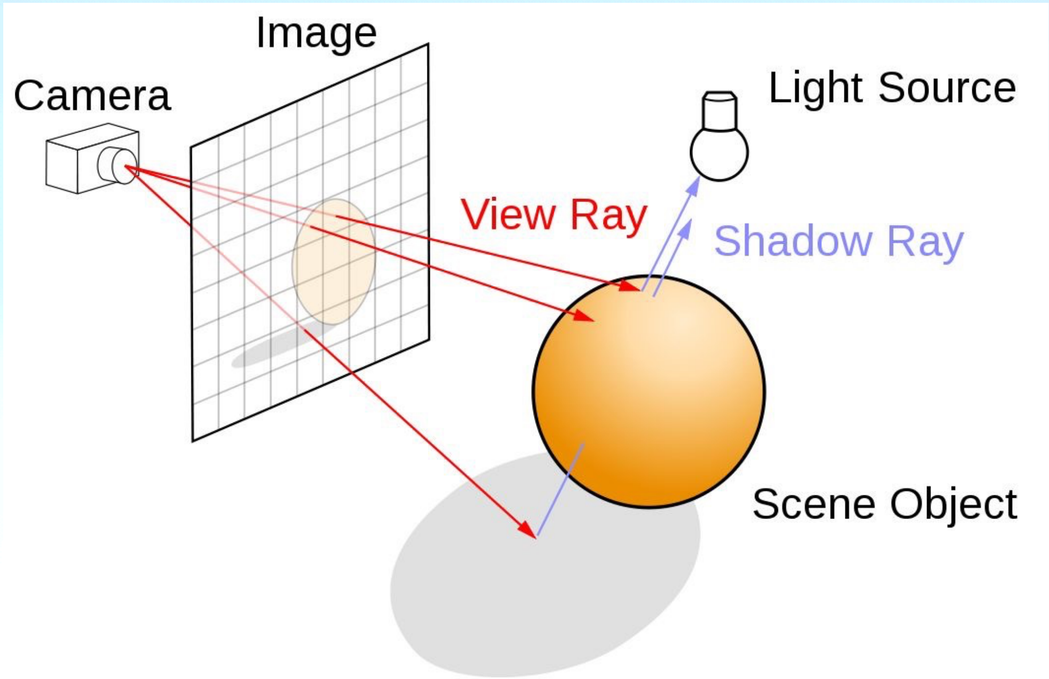
Gradient Estimation for Real-Time Adaptive Temporal Filtering

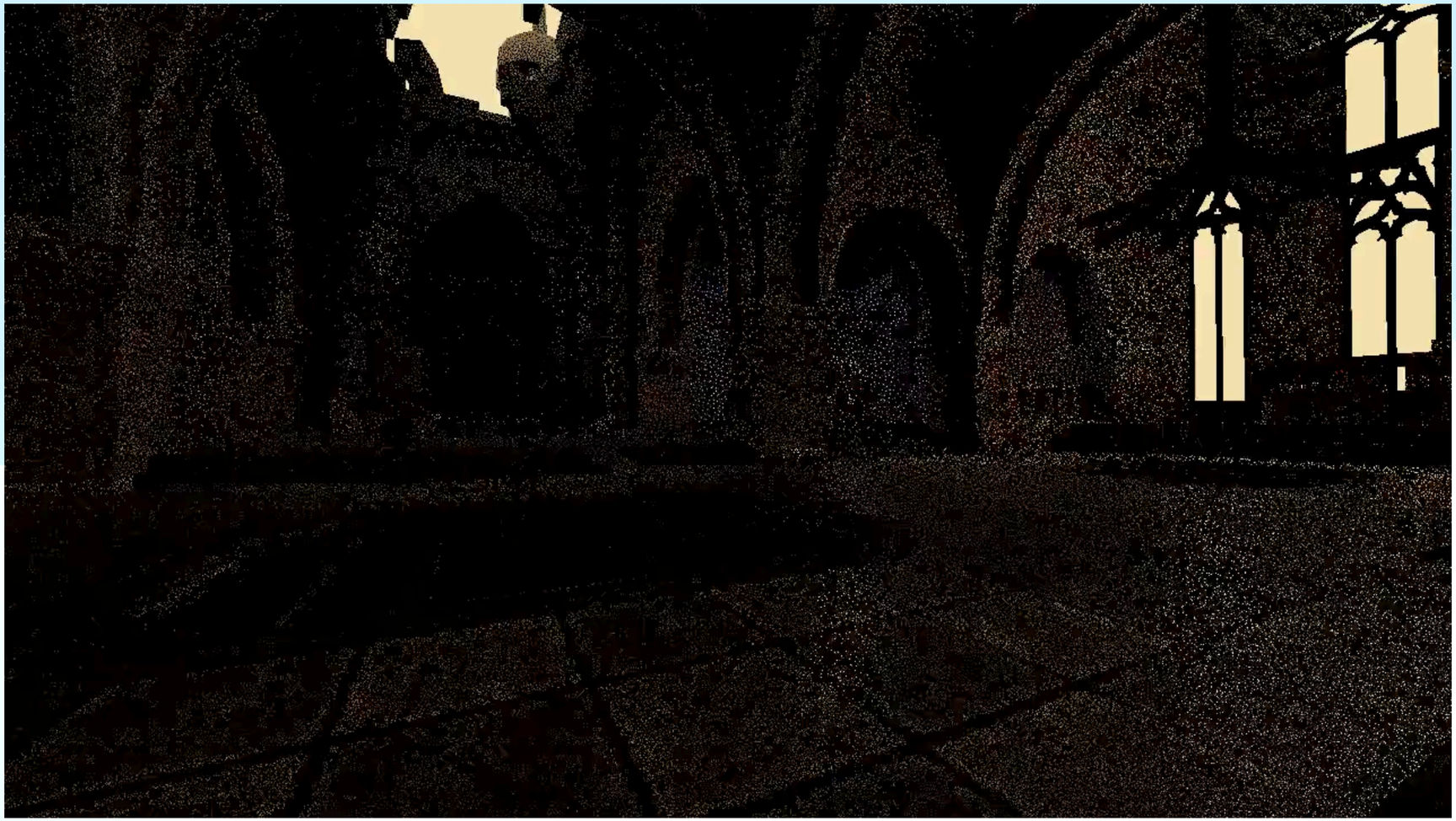
Kai YANG

Real-Time Render

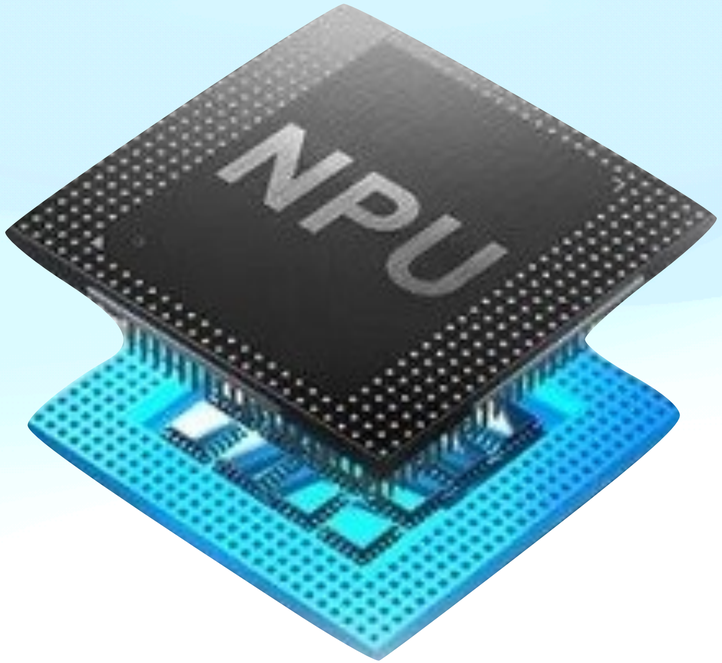
Real-Time Render



light tracing



How ?



Neural Processing Unit


720 p
1080 p through NPU

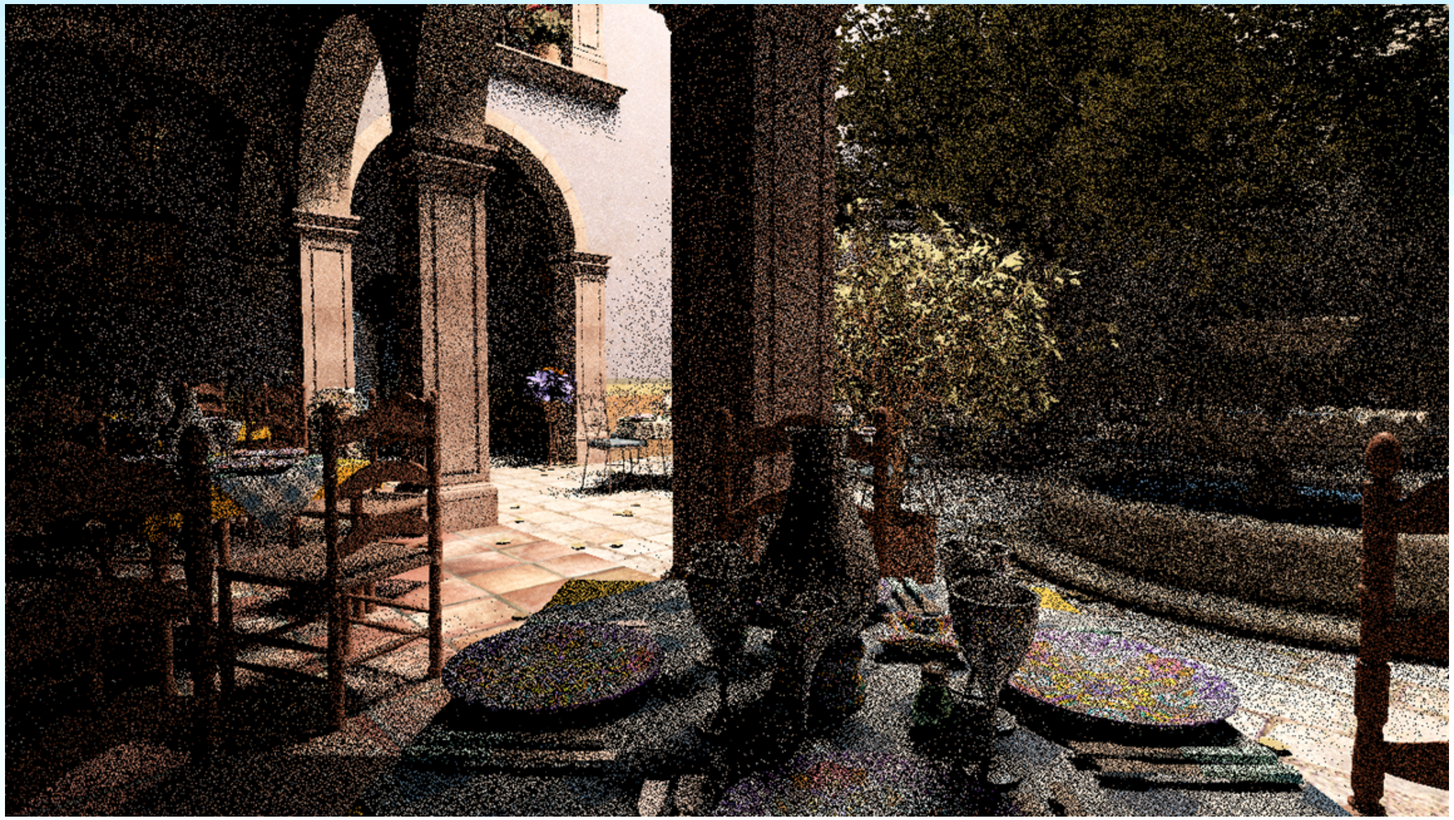
Upsampling

With Adaptive Temporal Filtering

(A-SVGF)

SVGF

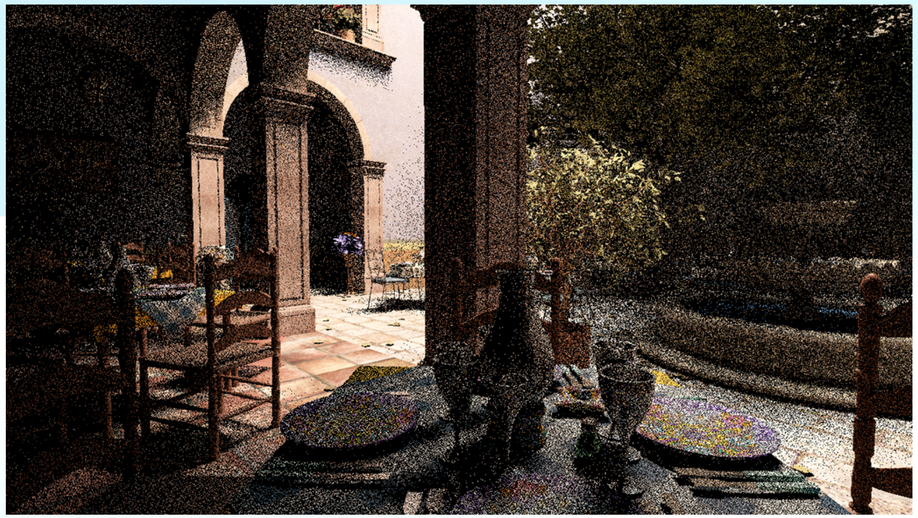
Spatiotemporal Variance-guided Filtering



1SPP

SVGF

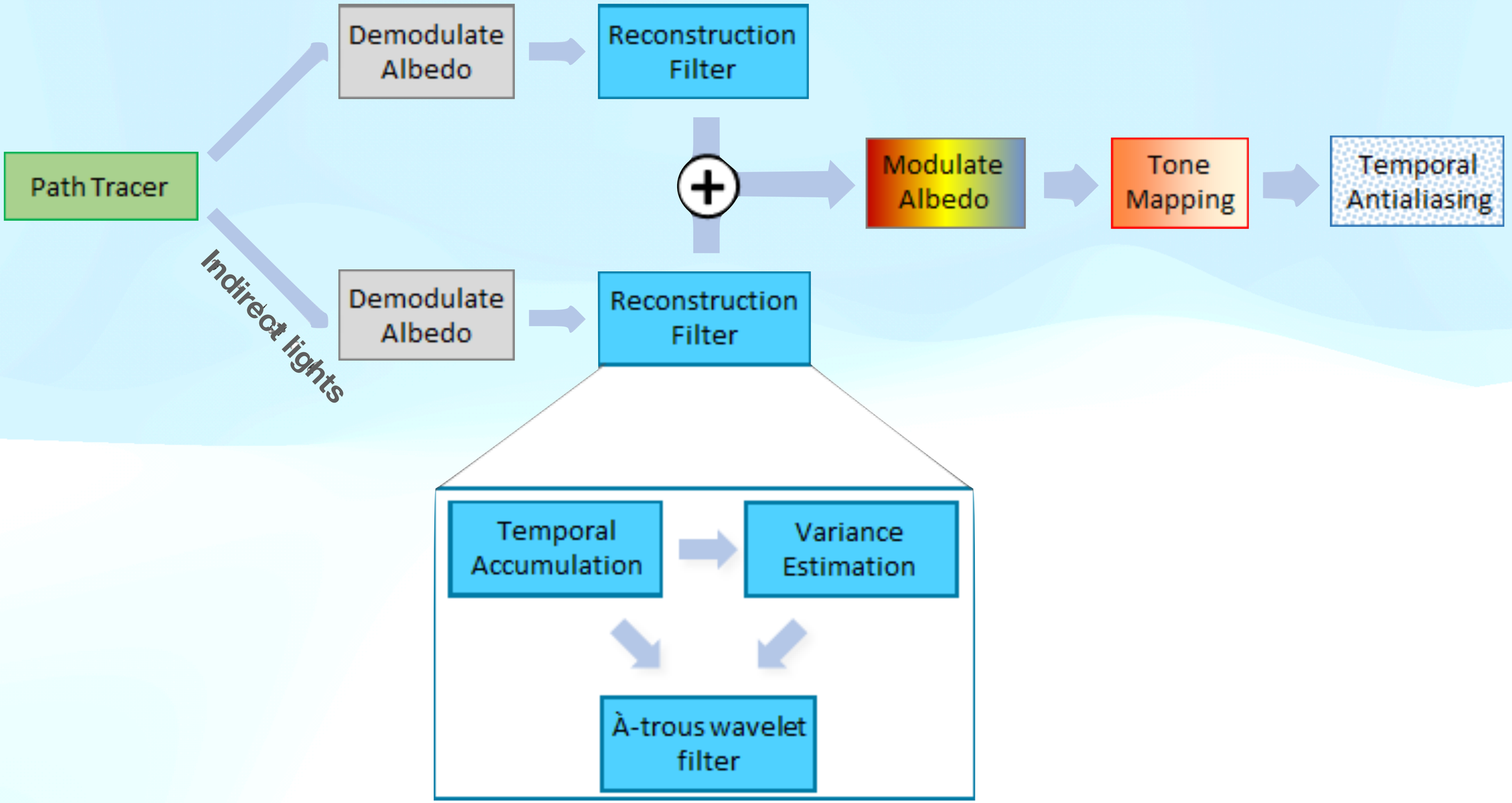
Spatiotemporal Variance-guided Filtering



Filtering

+

G-Buffer (color map, motion map, depth, etc)


Path Tracer
Demodulate Albedo
Reconstruction Filter
Indirect lights
Demodulate Albedo
Reconstruction Filter
Modulate Albedo
Tone Mapping
Temporal Antialiasing
Temporal Accumulation
Variance Estimation
À-trous wavelet filter

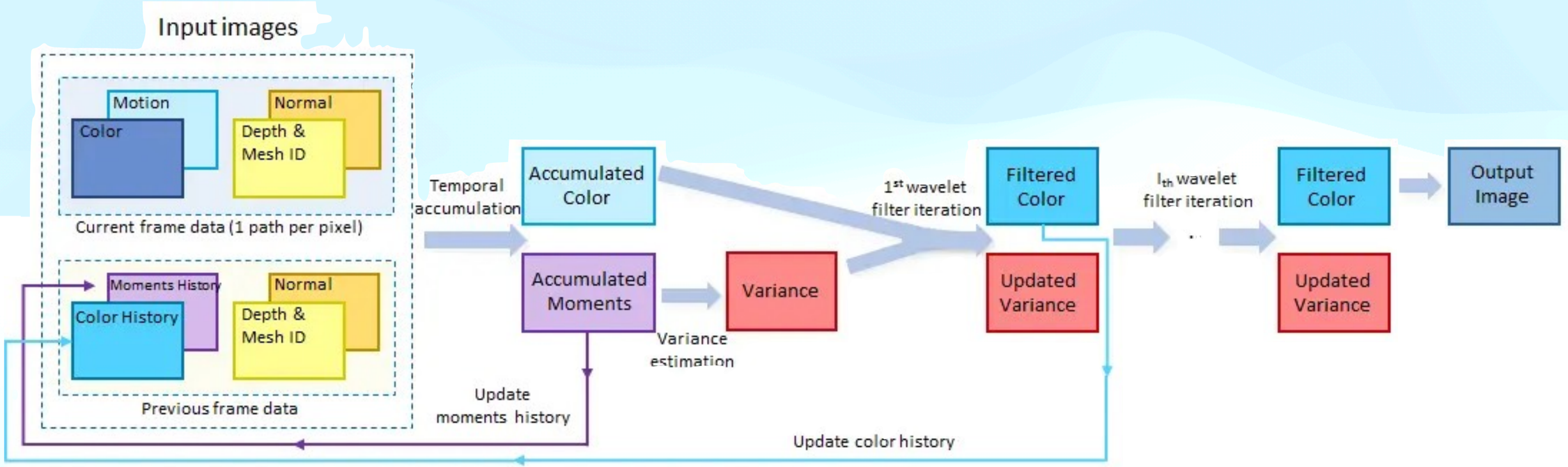
Temporal Accumulation

$$\hat{c}_i(x) = \alpha \cdot c_i(x) + (1 - \alpha) \cdot \hat{c}_{i-1}(\overleftarrow{x})$$

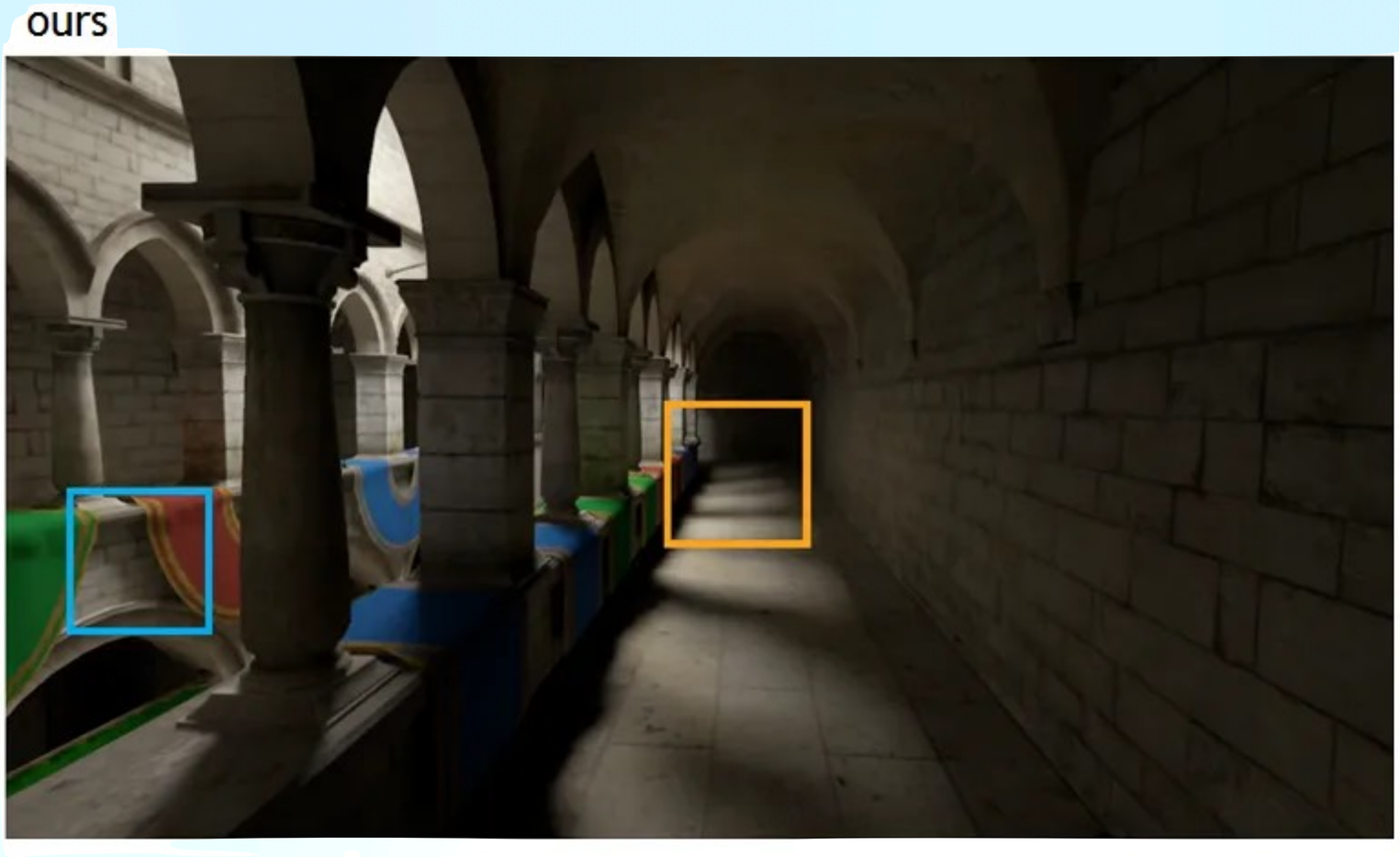
Current frame

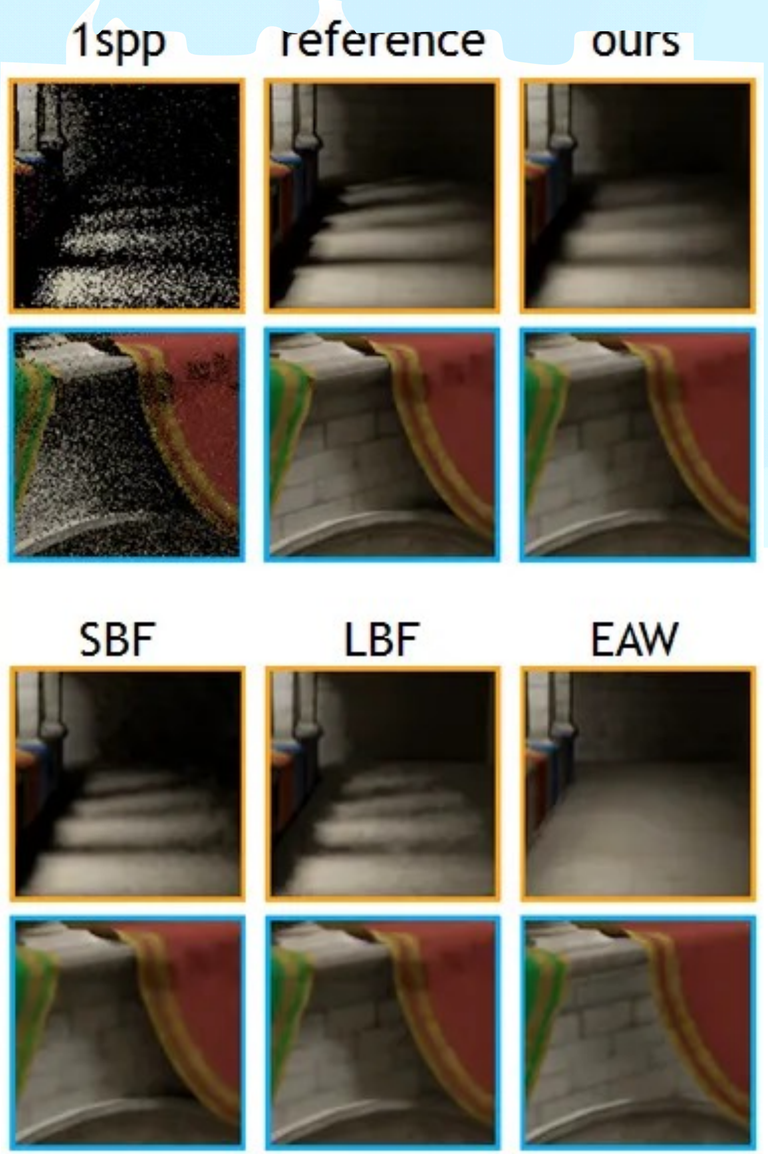
History frame which from motion vector reproduction

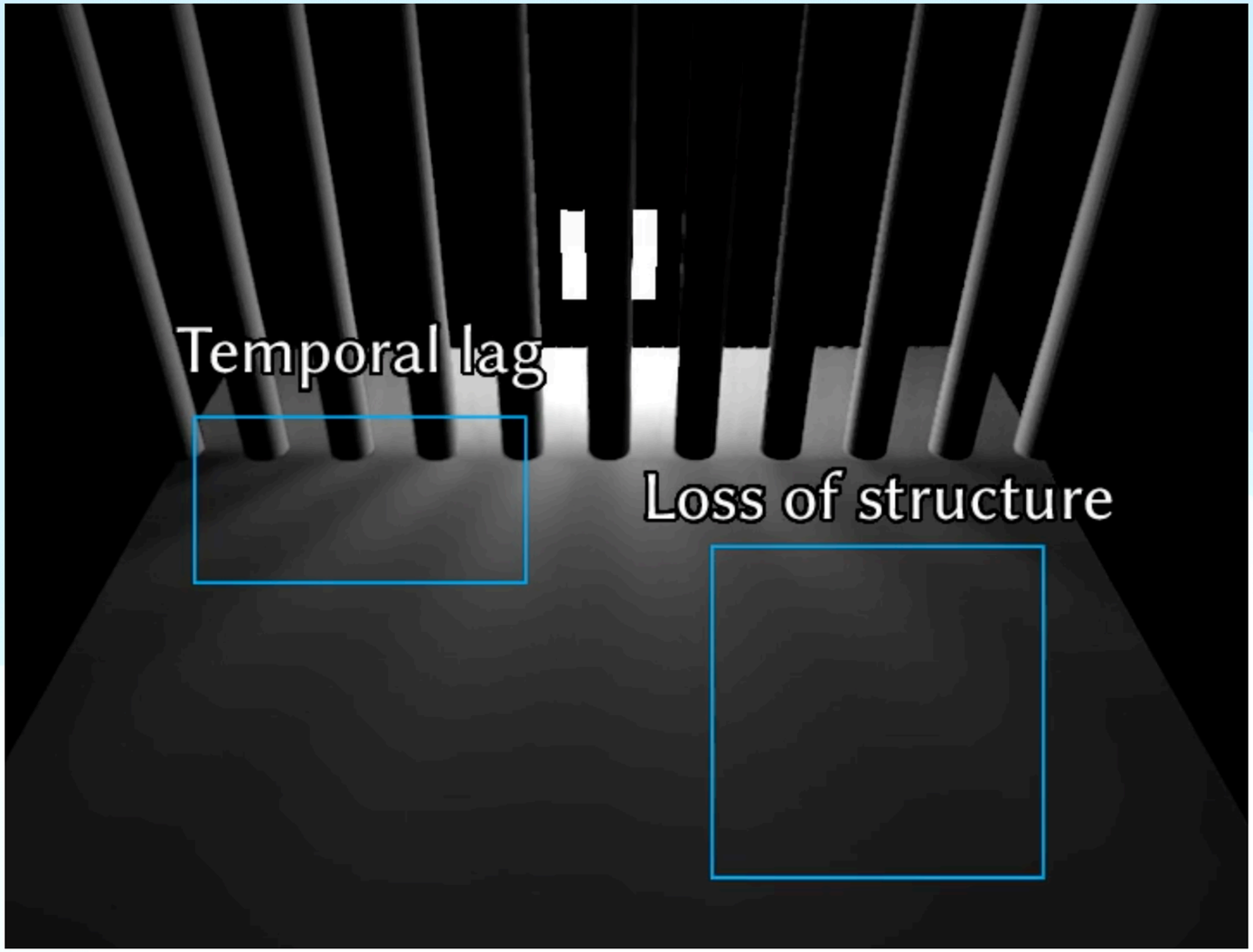
Reconstruct filtering



Result







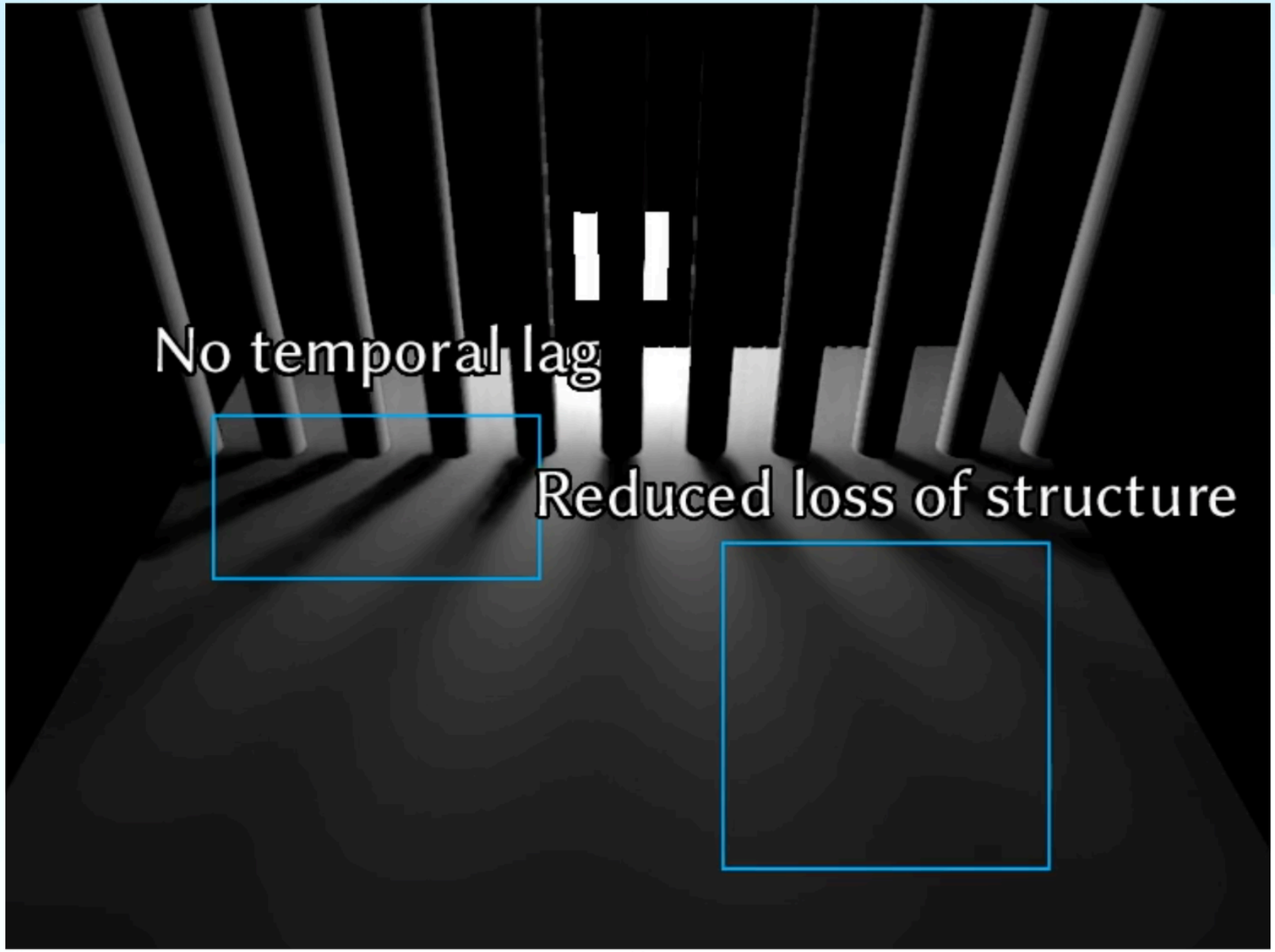
$$\hat{c}_i(x) = \alpha \cdot c_i(x) + (1 - \alpha) \cdot \hat{c}_{i-1}(\overleftarrow{x})$$

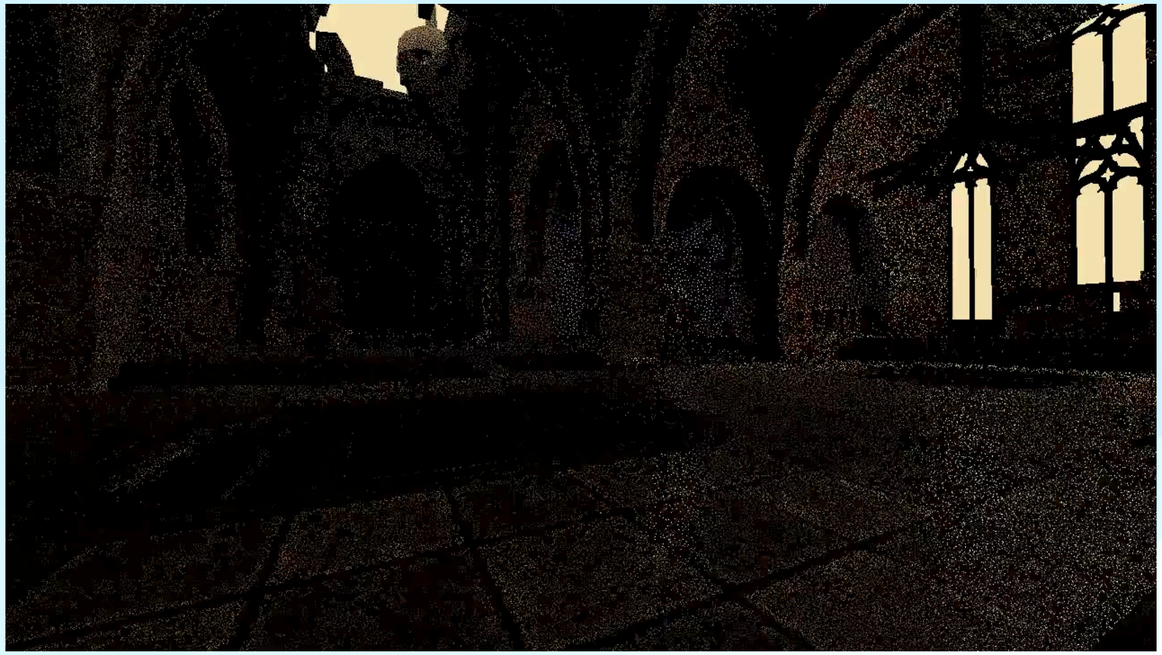
$$\hat{c}_i(x) = \alpha \cdot c_i(x) + (1 - \alpha) \cdot \hat{c}_{i-1}(\overleftarrow{x})$$

$$\alpha_i(p) = (1 - \lambda(p)) \cdot \alpha + \lambda(p)$$

$$\lambda(p) = \min\left(1, \frac{\delta_i(p)}{\hat{\Delta}_i(p)}\right)$$

$$\hat{c}_i(x) = \alpha \cdot c_i(x) + (1 - \alpha) \cdot \hat{c}_{i-1}(\overleftarrow{x})$$





1SPP

a-SVGF

Thanks
for your attention