Supplementary material for: Perceptual error optimization for Monte Carlo rendering

VASSILLEN CHIZHOV, MIA Group, Saarland University, Max-Planck-Institut für Informatik, Germany
ILIYAN GEORGIEV, Autodesk, United Kingdom
KAROL MYSZKOWSKI, Max-Planck-Institut für Informatik, Germany
GURPRIT SINGH, Max-Planck-Institut für Informatik, Germany

In this supplemental document we discuss various details related to our general formulation from the main paper. We start with a description of the extension of our framework to the *a-priori* setting (Section 1). In Section 2 we describe a way in which textures can be accounted for in our *horizontal* approach, so that mispredictions due to multiplicative (and additive) factors are eliminated. In Section 3 we describe ways in which the runtime of iterative energy minimization methods can be improved considerably. Notably, an expression is derived allowing the energy difference due to trial swaps to be evaluated in constant time (no scaling with image size or kernel size). In the remaining sections we analyze how current *a-posteriori* [Heitz and Belcour 2019] (Section 5) and *a-priori* [Georgiev and Fajardo 2016; Heitz et al. 2019] (Section 6) state of the art approaches can be related to our framework. Interpretations are discussed, major sources of error are identified, and the assumptions of the algorithms are made explicit.

1 A-PRIORI OPTIMIZATION

We extend our theory to the *a-priori* setting and discuss the main factors affecting the quality. The quality of *a-priori* approaches is determined mainly by three factors: the energy, the search space, and the optimization strategy. We discuss each of those briefly in the following paragraphs.

Our energy. We extend the *a-posteriori* energy from the main paper in order to handle multiple estimators involving different integrands: $\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_T$, with associated weights $w_1, \ldots, w_T$:

$$E(\boldsymbol{S}) = \sum_{t=1}^{T} w_t \|\boldsymbol{g} * \boldsymbol{Q}_t(\boldsymbol{S}) - \boldsymbol{I}_t\|^2. \tag{1}$$

In the above $\boldsymbol{g}$ would typically be a low-pass kernel (e.g., Gaussian), and $\boldsymbol{I}_t$ is the integral of the function used in the estimator $\boldsymbol{Q}_t$. Through this energy a whole set of functions can be optimized for, in order for the sequence to be more robust to different scenes and estimators, that do not fit any of the considered integrands exactly. We note that the derived optimization in Section 3 below is also applicable to the minimization of the proposed energy.

Search space. The search space plays an important role for the qualities which the optimized sequences exhibit. A more restricted search space provides more robustness and may help avoid overfitting to the considered set of integrands.

For instance, sample sets may be generated randomly within each pixel. Then, their assignment to pixels may be optimized over the space of all possible permutations. This is the setting of *horizontal* methods. If additionally this assignment is done within each dimension separately it allows for an even better fit to the integrands in the energy (but may degrade the general integration properties of the sequence). The scrambling keys' search space in [Heitz et al. 2019] is a special case of the latter applied for the Sobol sequence.

Constraining the search space to points generated from low-discrepancy sequences provides further robustness and guarantees desirable integration properties of the considered sequences. Similarly to [Heitz et al. 2019], we can consider a search space of Sobol scrambling keys in order for the optimized sequence to have a low discrepancy.

Ideally, such integration properties should arise directly from the energy. However, in practice the scene integrand cannot be expected to exactly match the set of considered integrands, thus extra robustness is gained through the restriction. Additionally, optimizing for many dimensions at the same time is costly as noted in [Heitz et al. 2019], thus imposing low-discrepancy properties also helps in that regard.

Finally, by imposing strict search space constraints a severe restriction on the distribution of the error is imposed. This can be alleviated by imposing the restrictions through soft penalty terms in the energy. This can allow for a trade-off between blue noise distribution and integration quality for example.

Progressive rendering. In order to make the sequence applicable to progressive rendering, subsets of samples should be considered in the optimization. Given a sample set S_i for pixel i we can decompose it in sample sets of $1, \ldots, N$ samples: $S_{i,1} \subset \ldots \subset S_{i,N} \equiv S_i$. We denote the respective images of sample sets $\boldsymbol{S}_1, \ldots, \boldsymbol{S}_N$.

Then an energy that also optimizes for the distribution of the error at each sample count is:

$$E(\boldsymbol{S}) = \sum_{t=1}^{T} \sum_{k=1}^{N} w_{t,k} \|\boldsymbol{g} * \boldsymbol{Q}_t(\boldsymbol{S}_k) - \boldsymbol{I}_t\|^2. \tag{2}$$

If $w_{i,k}$ are set to zero for $k < N$ then the original formulation is recovered. The more general formulation imposes additional constraints on the samples, thus the quality at the full sample count may be compromised if we also require a good quality at lower sample counts.

Choosing samples from S_i for $S_{i,1}, \ldots, S_{i,N-1}$ (in each dimension) constitutes a *vertical* search space analogous to the one discussed in the main paper for *a-posteriori* methods. The ranking keys' optimization in [Heitz et al. 2019] is a special case of this search space for the Sobol sequence.

Adaptive sampling can be handled by allowing a varying number of samples per pixel, and a corresponding energy derived from the one above. Note that this poses further restrictions on the achievable distribution.

Optimization strategies. Typically the energies for *a-priori* methods have been optimized through simulated annealing [Georgiev and Fajardo 2016; Heitz et al. 2019]. Metaheuristics can lead to very good minima especially if the runtime is not of great concern, which is the case since the sequences are precomputed. Nevertheless, the computation still needs to be tractable. The energies in previous works are generally not cheap to evaluate. On the other hand, our energies, especially if the optimizations in Section 3 are considered, can be evaluated very efficiently. This is beneficial for keeping the runtime of metaheuristics manageable, allowing for more complex search spaces to be considered.

Implementation details. Implementation decisions for a renderer, such as how samples are consumed, or how those are mapped to the hemisphere and light sources, affect the estimator $\boldsymbol{Q}$. This is important, especially when choosing $\boldsymbol{Q}$ for the described energies to optimize a sequence. It is possible that very small implementation changes make a previously ideal sequence useless for a specific renderer. It is important to keep this in mind when optimizing sequences by using the proposed energies and when those are used in a renderer.

2 TEXTURE DEMODULATION FOR HORIZONTAL OPTIMIZATION

Our iterative energy minimization algorithms (Alg. 1, Alg. 2) directly work with the original energy formulation, unlike error diffusion and dither matrix halftoning which only approximately minimize the energy. This allows textures to be handled more robustly compared to the permutation approach of Heitz and Belcour.

Reducing misprediction errors. Our horizontal approach relies on a dissimilarity metric $d(\cdot,\cdot)$ which approximates terms involving the difference $\boldsymbol{\Delta}$ due to swapping sample sets instead of pixels. This difference can be decreased, so that d is a better approximation, if additional information is factored out in the energy: screen-space varying multiplicative and additive terms. Specifically, if we have a spatially varying multiplicative image $\boldsymbol{\alpha}$, and a spatially varying additive image $\boldsymbol{\beta}$:

$$\boldsymbol{Q} = \boldsymbol{\alpha}\boldsymbol{Q}' + \boldsymbol{\beta} \tag{3}$$

$$\boldsymbol{\Delta}'(\pi) = \boldsymbol{\alpha} \odot \boldsymbol{Q}'(\pi(\boldsymbol{S})) - \boldsymbol{\alpha} \odot \pi(\boldsymbol{Q}'(\boldsymbol{S})) \tag{4}$$

$$\begin{aligned}\boldsymbol{\Delta}(\pi) &= \boldsymbol{Q}(\pi(\boldsymbol{S})) - \pi(\boldsymbol{Q}(\boldsymbol{S})) = \\ &\boldsymbol{\alpha} \odot \boldsymbol{Q}'(\pi(\boldsymbol{S})) + \boldsymbol{\beta} - \pi(\boldsymbol{\alpha} \odot \boldsymbol{Q}'(\boldsymbol{S}) + \boldsymbol{\beta}),\end{aligned} \tag{5}$$

we can make use of this in our formulation:

$$E(\pi) = \|\boldsymbol{g} * \boldsymbol{Q}(\pi(\boldsymbol{S})) - \boldsymbol{h} * \boldsymbol{I}\|_2^2 \tag{6}$$

$$\sqrt{E(\pi)} \le \|\boldsymbol{g} * (\boldsymbol{\alpha} \odot \pi(\boldsymbol{Q}'(\boldsymbol{S})) + \boldsymbol{\beta}) - \boldsymbol{h} * \boldsymbol{I}\|_2 + \|\boldsymbol{g}\|_1 \|\boldsymbol{\Delta}'\|_2. \tag{7}$$

Contrast this to the original formulation where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are not factored out:

$$\sqrt{E(\pi)} \le \|\boldsymbol{g} * \pi\left(\boldsymbol{\alpha} \odot \boldsymbol{Q}'(\boldsymbol{S}) + \boldsymbol{\beta}\right) - \boldsymbol{h} * \boldsymbol{I}\|_2 + \|\boldsymbol{g}\|_1 \|\boldsymbol{\Delta}\|_2. \tag{8}$$

With the new formulation it is sufficient that $\boldsymbol{Q}'(\pi(\boldsymbol{S})) = \pi(\boldsymbol{Q}'(\boldsymbol{S}))$ for $\boldsymbol{\Delta}'$ to be zero, while originally both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ play a role in $\boldsymbol{\Delta}$ becoming zero. Intuitively this means that screen space integrand differences due to additive and multiplicative factors do not result in mispredictions with the new formulation, if the integrand is assumed to be the same (locally) in screen space.

Comparison to demodulation. In the method of Heitz and Belcour the permutation is applied on the albedo demodulated image. This preserves the property that the global minimum of the implicit energy can be found through sorting. Translated to our framework this can be formulated as ($\boldsymbol{B}$ is a blue noise mask optimized for a kernel $\boldsymbol{g}$):

$$E_{HBP}(\pi) = \|\pi(\boldsymbol{Q}'(\boldsymbol{S})) - \boldsymbol{I}' - \boldsymbol{B}\|_2^2 \approx \|\boldsymbol{g} * \pi(\boldsymbol{Q}'(\boldsymbol{S})) - \boldsymbol{g} * \boldsymbol{I}'\|_2^2. \tag{9}$$

We have assumed that $\boldsymbol{\beta}$ is zero, but we can also extend the method to handle an additive term $\boldsymbol{\beta}$ as in our case. The more important distinction is that while the albedo demodulated image $\boldsymbol{Q}'$ is used in the permutation, it is never re-modulated ($\boldsymbol{\alpha} \odot \cdot$ is missing). Thus, this does not allow for proper handling of textures, even if it allows for modest improvements in practice. An example of a fail case consists of an image $\boldsymbol{\alpha}$ that is close to white noise. Then the error distribution will also be close to white noise due to the missing $\boldsymbol{\alpha} \odot \cdot$ factor. More precisely, even if $\pi(\boldsymbol{Q}'(\boldsymbol{S})) - \boldsymbol{I}'$ is distributed as $\boldsymbol{B}$, this does not imply that $\boldsymbol{\alpha} \odot \pi(\boldsymbol{Q}'(\boldsymbol{S})) - \boldsymbol{I}'$ will be distributed similarly. Dropping $\boldsymbol{\alpha} \odot \cdot$ is, however, a reasonable option if one is restricted to sorting as an optimization strategy.

We propose a modification of the original approach (and energy) such that not only the demodulated estimator values are used, but the blue noise mask $\boldsymbol{B}$ is also demodulated (Fig. 7). To better understand how it is derived (and how $\boldsymbol{\beta}$ may be integrated) we study a bound based on the assumption that $\alpha_i \in [0,1]$, and $\boldsymbol{\Delta}' = 0$

$$E(\pi) = \|\boldsymbol{g} * (\boldsymbol{\alpha} \odot \pi(\boldsymbol{Q}'(\boldsymbol{S})) + \boldsymbol{\beta}) - \boldsymbol{g} * \boldsymbol{I}'\|_2^2 \approx \tag{10}$$

$$\|\boldsymbol{\alpha} \odot \pi(\boldsymbol{Q}'(\boldsymbol{S})) + \boldsymbol{\beta} - \boldsymbol{I}' - \boldsymbol{B}\|_2^2 = \tag{11}$$

$$\sum_i \alpha_i^2 \left((\pi(\boldsymbol{Q}'(\boldsymbol{S})))_i + \frac{\beta_i - I'_i - B_i}{\alpha_i} \right)^2 \le \tag{12}$$

$$\left\| \pi(\boldsymbol{Q}'(\boldsymbol{S})) + \frac{\boldsymbol{\beta} - \boldsymbol{I}' - \boldsymbol{B}}{\boldsymbol{\alpha}} \right\|_2^2. \tag{13}$$

The global minimum of the last energy (w.r.t. π) can be found through sorting also, since there is no spatially varying multiplicative factor $\boldsymbol{\alpha}$ in front of the permutation.

Sinusoidal textures. To demonstrate texture handling (multiplicative term $\boldsymbol{\alpha}$), in the top row of Fig. 1, a white-noise texture W is multiplied with a sine-wave input signal: $f(x,y) = 0.5 * (1.0 + \sin(x+y)) * W(x,y)$. The reference is a constant image at 0.5. Heitz and Belcour proposed to handle such textures by applying their method on the albedo-demodulated image. While this strategy may lead to a modest improvement, it ignores the fact that the image is produced by re-modulating the albedo, which can negate that improvement. Instead, our horizontal iterative minimization algorithm can incorporate the albedo explicitly using the discussed energy.

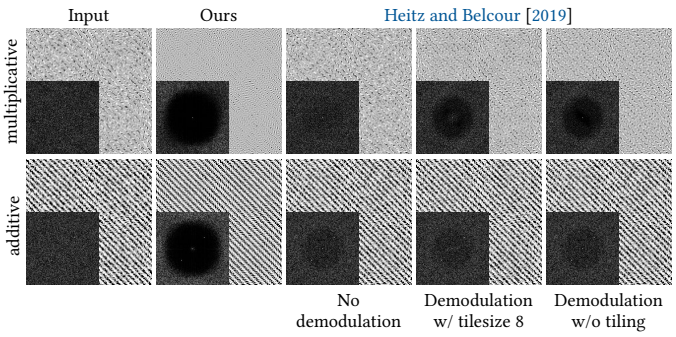


Fig. 1. We demonstrate the importance of the extension presented in Section 2. A high-frequency sinusoidal texture is corrupted by white noise (leftmost column) multiplicatively (**top row**) and additively (**bottom row**). Contrary to Heitz and Belcour's method, our optimization distributes error as a high-quality blue-noise distribution (see the power-spectrum insets). The reference images for the top/bottom image are respectively a flat grey and a sinusoidal image.

The bottom row demonstrates the effect of a non-flat signal on the error distribution (additive term $\boldsymbol{\beta}$). Here W is added to a sine-wave input signal: $f(x,y) = 0.5 * (1.0 + \sin(x+y)) + W(x,y)$. The reference image is $0.5 * (1 + \sin(x+y))$. Our optimization is closer to the reference suggesting that our method can greatly outperform the current state of the art by properly accounting for auxiliary information, especially in regions with high-frequency textures.

Dimensional decomposition. The additive factor $\boldsymbol{\beta}$ can be used to motivate splitting the optimization over several dimensions, since the Liouville–Neumann expansion of the rendering equation is additive [Kajiya 1986]. If some dimensions are smooth (e.g., lower dimensions), then a screen space local integrand similarity assumption can be encoded in $d(\cdot,\cdot)$ and it will approximate $\boldsymbol{\Delta}$ better for smoother dimensions. If the optimization is applied over all dimensions at the same time, this may result in many mispredictions due to the assumption being violated for dimensions in which the integrand is less smooth in screen space (e.g., higher dimensions). We propose splitting the optimization problem starting from lower dimensions and sequentially optimizing higher dimensions while encoding a local smoothness (in screen space) assumption on the integrand in $d(\cdot,\cdot)$ (e.g., swaps limited to a small neighborhood around the pixel). This requires solving several optimization problems, but potentially reduces the amount of mispredictions. Note that it does not require more rendering operations than usual.

3 IMPROVING ITERATIVE-OPTIMIZATION PERFORMANCE

The main cost of iterative minimization methods is computing the energy for each trial swap, more specifically the required convolution and the subsequent norm computation. In the work of Analoui and Allebach an optimization has been proposed to efficiently evaluate such trial swaps, without recomputing a convolution or norm at each step, yielding a speed up of more than 10 times. The optimization relies on the assumption that the kernel $\boldsymbol{g}$ is the same in screen space (the above optimization is not applicable for spatially varying kernels). We extend the described optimization to a more general case, also including spatially varying kernels. We also note some details not mentioned in the original paper.

3.1 Horizontal swaps

We will assume the most general case: instead of just swapping pixels, we consider swapping sample sets from which values are generated through $\boldsymbol{Q}$. It subsumes both swapping pixel values and swapping pixel values in the presence of a multiplicative factor $\boldsymbol{\alpha}$.

Single swap. The main goal is to evaluate the change of the energy δ due to a swap between the sample sets of some pixels a, b. More precisely, if the original sample set image is $\boldsymbol{S}$ then the new sample set image is $\boldsymbol{S'}$ such that $S'_a = S_b, S'_b = S_a$, and $S'_i = S_i$ everywhere else. This corresponds to images: $\boldsymbol{Q} = \boldsymbol{Q}(\boldsymbol{S})$ and $\boldsymbol{Q'} = \boldsymbol{Q}(\boldsymbol{S'})$. The two images differ only in the pixels with indices a and b. Let:

$$\delta_a = Q'_a - Q_a = Q_a(S_b) - Q_a(S_a) \tag{14}$$

$$\delta_b = Q'_b - Q_b = Q_b(S_a) - Q_b(S_b). \tag{15}$$

We will also denote the convolved images $\tilde{\boldsymbol{Q}} = \boldsymbol{g} * \boldsymbol{Q}$ and $\tilde{\boldsymbol{Q}}' = \boldsymbol{g} * \boldsymbol{Q'}$, and also $\boldsymbol{\epsilon} = \tilde{\boldsymbol{Q}} - \boldsymbol{I}$. Specifically:

$$\tilde{Q}_i = \sum_{j \in \mathbb{Z}^2} Q_j g_{i-j},\ \tilde{Q}'_i = \tilde{Q}_i + \delta_a g_{i-a} + \delta_b g_{i-b}. \tag{16}$$

We want to be able to efficiently evaluate $\delta = \|\tilde{\boldsymbol{Q}}' - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}} - \boldsymbol{I}\|^2$, since in the iterative minimization algorithms the candidate with the minimum δ is kept. Using the above expressions for $\tilde{Q}'_i$ we rewrite δ as:

$$\delta = \|\tilde{\boldsymbol{Q}}' - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}} - \boldsymbol{I}\|^2 = \tag{17}$$

$$\sum_{i \in \mathbb{Z}^2} \|\tilde{Q}_i - I_i + \delta_a g_{i-a} + \delta_b g_{i-b}\|^2 - \|\tilde{\boldsymbol{Q}} - \boldsymbol{I}\|^2 = \tag{18}$$

$$2 \sum_{i \in \mathbb{Z}^2} \langle \tilde{Q}_i - I_i, \delta_a g_{i-a} + \delta_b g_{i-b} \rangle + \sum_{i \in \mathbb{Z}^2} \|\delta_a g_{i-a} + \delta_b g_{i-b}\|^2 = \tag{19}$$

$$\begin{aligned} &2\langle \delta_a, \sum_{i \in \mathbb{Z}^2} \epsilon_i g_{i-a} \rangle + 2\langle \delta_b, \sum_{i \in \mathbb{Z}^2} \epsilon_i g_{i-b} \rangle + \\ &\langle \delta_a^2, \sum_{i \in \mathbb{Z}^2} g_{i-a} g_{i-a} \rangle + \langle \delta_b^2, \sum_{i \in \mathbb{Z}^2} g_{i-b} g_{i-b} \rangle + \\ &2\langle \delta_a \delta_b, \sum_{i \in \mathbb{Z}^2} g_{i-a} g_{i-b} \rangle = \end{aligned} \tag{20}$$

$$\begin{aligned} &2\langle \delta_a, C_{\boldsymbol{g},\boldsymbol{\epsilon}}(a) \rangle + 2\langle \delta_b, C_{\boldsymbol{g},\boldsymbol{\epsilon}}(b) \rangle + \\ &\langle (\delta_a^2 + \delta_b^2), C_{\boldsymbol{g},\boldsymbol{g}}(0) \rangle + 2\langle \delta_a \delta_b, C_{\boldsymbol{g},\boldsymbol{g}}(b-a) \rangle, \end{aligned} \tag{21}$$

where $C_{f,h}(x) = \sum_{i \in \mathbb{Z}^2} f(i-x)h(i)$ is the cross-correlation of f and h. We have reduced the computation of δ to the sum of only 4 terms. Assuming that $C_{\boldsymbol{g},\boldsymbol{g}}$ is known (it can be precomputed once for a known kernel) and that $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$ is known (it can be recomputed after a sufficient amount of swaps have been accepted), then evaluating a trial swap takes constant time (it does not scale in the size of the image or the size of the kernel).

Multiple accepted swaps. It may be desirable to avoid recomputing $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$ even upon accepting a trial swap. For that purpose we extend the strategy from [Analoui and Allebach 1992] for computing $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$,

where $\boldsymbol{\epsilon}^n$ is the error image after n swaps have been accepted:

$$\{(\delta_{a^1}, \delta_{b^1}), \ldots, (\delta_{a^n}, \delta_{b^n})\}. \tag{22}$$

This implies: $\tilde{Q}_i^n = \tilde{Q} + \sum_{k=1}^n (\delta_{a^k} g_{i-a^k} + \delta_{b^k} g_{i-b^k})$, and consequently:

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(x) = \tag{23}$$

$$\sum_{i \in \mathbb{Z}^2} \left(\tilde{Q}_i - I_i + \sum_{k=1}^n (\delta_{a^k} g_{i-a^k} + \delta_{b^k} g_{i-b^k}) \right) g_{i-x} = \tag{24}$$

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}}(x) + \sum_{k=1}^n (\delta_{a^k} C_{\boldsymbol{g},\boldsymbol{g}}(x - a^k) + \delta_{b^k} C_{\boldsymbol{g},\boldsymbol{g}}(x - b^k)). \tag{25}$$

This allows avoiding the recomputation of $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$ after every accepted swap, and instead, the delta on the $n+1$-st swap with trial differences δ_a, δ_b is:

$$\delta^{n+1} = \|\boldsymbol{Q}^{n+1} - \boldsymbol{I}\|^2 - \|\boldsymbol{Q}^n - \boldsymbol{I}\|^2 = \tag{26}$$

$$\begin{aligned} &2\langle \delta_a, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(a) \rangle + 2\langle \delta_b, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(b) \rangle + \\ &\langle (\delta_a^2 + \delta_b^2), C_{\boldsymbol{g},\boldsymbol{g}}(0) \rangle + 2\langle \delta_a \delta_b, C_{\boldsymbol{g},\boldsymbol{g}}(b - a) \rangle, \end{aligned} \tag{27}$$

where $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$ is computed from $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$ and $C_{\boldsymbol{g},\boldsymbol{g}}$ as derived in Eq. (17). This computation scales only in the number of accepted swaps since the last recomputation of $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$. We also note that $C_{\boldsymbol{g},\boldsymbol{g}}(x - y)$ evaluates to zero if $x - y$ is outside of the support of $C_{\boldsymbol{g},\boldsymbol{g}}$. Additional optimizations have been devised due to this fact [Analoui and Allebach 1992].

3.2 Vertical swaps

In the *vertical* setting swaps happen only within the pixel itself, that is: $\delta_a = Q_a(S'_a) - Q_a(S_a)$. Consequently, $\tilde{Q}'_i = \tilde{Q}_i + \delta_a g_{i-a}$. Computing the difference in the energies for the $n+1$-st swap:

$$\delta^{n+1} = \|\tilde{\boldsymbol{Q}}^{n+1} - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}}^n - \boldsymbol{I}\|^2 = \tag{28}$$

$$\sum_{i \in \mathbb{Z}^2} \|\tilde{Q}_i^n - I_i + \delta_a g_{i-a}\|^2 - \|\tilde{\boldsymbol{Q}}^n - \boldsymbol{I}\|^2 = \tag{29}$$

$$2 \sum_{i \in \mathbb{Z}^2} \langle \tilde{Q}_i^n - I_i, \delta_a g_{i-a} \rangle + \sum_{i \in \mathbb{Z}^2} \|\delta_a g_{i-a}\|^2 = \tag{30}$$

$$2\langle \delta_a, \sum_{i \in \mathbb{Z}^2} \epsilon_i^n g_{i-a} \rangle + \langle \delta_a^2, \sum_{i \in \mathbb{Z}^2} g_{i-a} g_{i-a} \rangle = \tag{31}$$

$$2\langle \delta_a, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(a) \rangle + \langle \delta_a^2, C_{\boldsymbol{g},\boldsymbol{g}}(0) \rangle. \tag{32}$$

The corresponding expression for $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$ is:

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(x) = C_{\boldsymbol{g},\boldsymbol{\epsilon}}(x) + \sum_{k=1}^n \delta_{a^k} C_{\boldsymbol{g},\boldsymbol{g}}(x - a^k). \tag{33}$$

3.3 Multiple simultaneous updates

If the search space is ignored and the formulation is analyzed in an abstract setting it becomes obvious that the *vertical* approach corresponds to an update of a single pixel, while the *horizontal* approach corresponds to an update of two pixels at the same time. This can be generalized further. Let N different pixels be updated per trial, and let there be n trials that have been accepted since $C_{\boldsymbol{g},\boldsymbol{\epsilon}}$ has been updated. Let the pixels to be updated in the current trial be: $a_1^{n+1}, \ldots, a_N^{n+1}$, and the accepted update at step k be at pixels: $a_1^k, \ldots, a_N^k$. Let $\boldsymbol{Q}^0 = \boldsymbol{Q}$ be the original image. We define the sequence of images: $\boldsymbol{Q}^k : Q_i^k = Q_i^{k-1}, i \notin \{a_1^k, \ldots, a_N^k\}$ and otherwise let $Q_{a_i^k}^k$ be given. Let $\delta_i^k = Q_{a_i^k}^k - Q_{a_i^k}^{k-1}$. Using the above notation we arrive at an expression for $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$:

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(x) = C_{\boldsymbol{g},\boldsymbol{\epsilon}}(x) + \sum_{k=1}^n \sum_{i=1}^N \delta_i^k C_{\boldsymbol{g},\boldsymbol{g}}(x - a_i^k). \tag{34}$$

The change in the energy due to the $n+1$-st update is:

$$\delta^{n+1} = \|\tilde{\boldsymbol{Q}}^{n+1} - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}}^n - \boldsymbol{I}\|^2 = \tag{35}$$

$$\sum_{i \in \mathbb{Z}^2} \|\tilde{Q}_i^n - I_i + \sum_{j=1}^N \delta_j^{n+1} g_{i-a_j^{n+1}}\|^2 - \|\tilde{\boldsymbol{Q}}^n - \boldsymbol{I}\|^2 = \tag{36}$$

$$2 \sum_{i \in \mathbb{Z}^2} \langle \tilde{Q}_i^n - I_i, \sum_{j=1}^N \delta_j^{n+1} g_{i-a_j^{n+1}} \rangle + \sum_{i \in \mathbb{Z}^2} \|\sum_{j=1}^N \delta_j^{n+1} g_{i-a_j^{n+1}}\|^2 = \tag{37}$$

$$\begin{aligned} &2 \sum_{j=1}^N \langle \delta_j^{n+1}, \sum_{i \in \mathbb{Z}^2} \epsilon_i^n g_{i-a_j^{n+1}} \rangle + \\ &\sum_{j=1}^N \sum_{k=1}^N \langle \delta_j^{n+1} \delta_k^{n+1}, \sum_{i \in \mathbb{Z}^2} g_{i-a_j^{n+1}} g_{i-a_k^{n+1}} \rangle = \end{aligned} \tag{38}$$

$$\begin{aligned} &2 \sum_{j=1}^N \langle \delta_j^{n+1}, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(a_j^{n+1}) \rangle + \\ &\sum_{j=1}^N \sum_{k=1}^N \langle \delta_j^{n+1} \delta_k^{n+1}, C_{\boldsymbol{g},\boldsymbol{g}}(a_j^{n+1} - a_k^{n+1}) \rangle. \end{aligned} \tag{39}$$

3.4 Implementation details

Leaky energy. Similar to the original paper [Analoui and Allebach 1992], in our extension δ was computed for a "leaky energy" which extended the support of the image by convolution. That is reflected in the fact that the sums are over $\mathbb{Z}^2$. To rectify this, the sum needs to be limited to the support of $\boldsymbol{I}$. This would require clamped sums of the cross-correlation to be evaluated, which can also be precomputed but requires extra memory. The same holds for the cross-correlation with $\boldsymbol{\epsilon}$, where clamped terms are required near the image boundary.

Reflecting boundary conditions. Another desirable property may be a convolution such that it acts on the image extended to be reflected at the boundaries - this avoids artifacts near the borders. This can be achieved by including the relevant terms including pixels for which the kernel is partially outside of the support of $\boldsymbol{I}$. Care must be taken when expressing $\tilde{Q}_i$, however, since it may include the same updated pixel numerous times (especially if it is near the border). The same ideas apply for a toroidally extended convolution.

Further optimizations. Various other strategies have been proposed in the literature for improving the runtime of iterative error minimization approaches for halftoning.

In our algorithms we usually use a randomized initial state, however, it is possibly to initialize the algorithms with the result of a dither matrix halftoning algorithm or error diffusion algorithm which would result in faster convergence [Analoui and Allebach 1992].

Another strategy involves partitioning the image in blocks. Instead of updating the pixels in raster or serpentine order, the blocks are updated simultaneously by keeping only the best update per block in each iteration. This has been reported to run 10+ times faster [Lieberman and Allebach 1997]. In the same paper [Lieberman and Allebach 1997], approximating the kernel with box functions has been proposed, yielding a speed up of 6 times. Similarly, if the kernel is separable or can be approximated by a separable kernel, the convolution can also be made considerably faster. A speed-up of an additional 30 times has been reported in [Koge et al. 2014] through the usage of a GPU.

Finally, several heuristics related to the order in which pixels are iterated over have been proposed in [Bhatt et al. 2006].

3.5 Spatially varying kernels

We propose an optimization for spatially varying kernels also. Let kernel $\boldsymbol{g}_i$ be associated with pixel i. Let pixel a be updated to a new value Q'_a, while everywhere else the images match: $Q'_i = Q_i$, and $\delta_a = Q'_a - Q_a$. We denote $\tilde{Q}_i = \langle \boldsymbol{g}_i, \boldsymbol{Q} \rangle$, $\tilde{Q}'_i = \langle \boldsymbol{g}_i, \boldsymbol{Q}' \rangle = \tilde{Q}_i + g_{i,a}\delta_a$. Our goal is to evaluate the change in the energy due to the update:

$$\delta = \|\tilde{\boldsymbol{Q}}' - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}} - \boldsymbol{I}\|^2 = \tag{40}$$

$$\sum_{i\in\mathbb{Z}^2} \|\tilde{Q}_i - I_i + g_{i,a}\delta_a\|^2 - \|\tilde{\boldsymbol{Q}} - \boldsymbol{I}\|^2 = \tag{41}$$

$$2\sum_{i\in\mathbb{Z}^2} \langle \epsilon_i, g_{i,a}\delta_a \rangle + \sum_{i\in\mathbb{Z}^2} \|g_{i,a}\delta_a\|^2 = \tag{42}$$

$$2\langle \delta_a, \sum_{i\in\mathbb{Z}^2} \epsilon_i g_{i,a} \rangle + \langle \delta_a^2, \sum_{i\in\mathbb{Z}^2} g_{i,a}g_{i,a} \rangle. \tag{43}$$

In the above $C_{\boldsymbol{g},\boldsymbol{g}}(a) = \sum_{i\in\mathbb{Z}^2} g_{i,a}g_{i,a}$ may be precomputed for every a, which yields a function with support $\mathrm{supp}(C_{\boldsymbol{g},\boldsymbol{g}}) = \bigcup_i \mathrm{supp}(\boldsymbol{g}_i)$, and $C_{\boldsymbol{g},\boldsymbol{\epsilon}}(a) = \sum_{i\in\mathbb{Z}^2} \epsilon_i g_{i,a}$ can also be recomputed after enough updates have been accepted.

Multiple accepted updates. Let a set of accepted updates results in the differences: $\{\delta_{a^1}, \ldots, \delta_{a^n}\}$. And let $\boldsymbol{\epsilon}^n$ be the error image after the updates. We derive an expression for the efficient evaluation of $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$:

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(x) = \sum_{i\in\mathbb{Z}^2} \epsilon_i^n g_{i,x} = C_{\boldsymbol{g},\boldsymbol{\epsilon}}(x) + \sum_{k=1}^{n} \delta_{a^k} \sum_{i\in\mathbb{Z}^2} g_{i,a^k}g_{i,x}. \tag{44}$$

An efficient computation of $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$ can then be achieved if the function $C_{\boldsymbol{g},\boldsymbol{g}}(x,y) = \sum_{i\in\mathbb{Z}^2} g_{i,x}g_{i,y}$ is precomputed. Then, at step $n+1$ the change in energy is:

$$\delta^{n+1} = \|\tilde{\boldsymbol{Q}}^{n+1} - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}}^{n} - \boldsymbol{I}\|^2 = \tag{45}$$

$$2\langle \delta_{a^{n+1}}, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(a^{n+1}) \rangle + \langle \delta_{a^{n+1}}^2, C_{\boldsymbol{g},\boldsymbol{g}}(a^{n+1}) \rangle. \tag{46}$$

Multiple simultaneous updates. We derive an expression where an update consists of changing N pixels simultaneously, and we assume that n such updates have been accepted previously. We denote the differences of the pixels in update k: $\{\delta_1^k, \ldots, \delta_N^k\}$. The expression for the change in the energy is given as:

$$\delta^{n+1} = \|\tilde{\boldsymbol{Q}}^{n+1} - \boldsymbol{I}\|^2 - \|\tilde{\boldsymbol{Q}}^{n} - \boldsymbol{I}\|^2 = \tag{47}$$

$$\sum_{i\in\mathbb{Z}^2} \|\tilde{Q}_i^n - I_i + \sum_{j=1}^{N} \delta_j^{n+1} g_{i,a_j^{n+1}}\|^2 - \|\tilde{\boldsymbol{Q}}^{n} - \boldsymbol{I}\|^2 = \tag{48}$$

$$2\sum_{j=1}^{N} \langle \delta_j^{n+1}, C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(a_j^{n+1}) \rangle + \sum_{i=1}^{N}\sum_{j=1}^{N} \langle \delta_i^{n+1}\delta_j^{n+1}, C_{\boldsymbol{g},\boldsymbol{g}}(a_i^{n+1}, a_j^{n+1}) \rangle. \tag{49}$$

Where $C_{\boldsymbol{g},\boldsymbol{g}}(x,y) = \sum_{i\in\mathbb{Z}^2} g_{i,x}g_{i,y}$ is assumed to be precomputed, and $C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}$ can be computed as:

$$C_{\boldsymbol{g},\boldsymbol{\epsilon}^n}(x) = C_{\boldsymbol{g},\boldsymbol{\epsilon}}(x) + \sum_{k=1}^{n}\sum_{j=1}^{N} \delta_{a_j^k} C_{\boldsymbol{g},\boldsymbol{g}}(a_j^k, x). \tag{50}$$

4 RELATIONSHIP TO PREVIOUS WORK

We show that the recent publications [Georgiev and Fajardo 2016; Heitz et al. 2019; Heitz and Belcour 2019] on blue noise error distribution for path tracing, can be seen as special cases in our framework. This allows for a novel analysis and interpretation of the results in the aforementioned works. We also state the necessary assumptions and approximations necessary to get from our general formulation to the algorithms presented in the papers.

Classification. The proposed techniques can be divided into *a-priori* [Georgiev and Fajardo 2016; Heitz et al. 2019] and *a-posteriori* [Heitz and Belcour 2019]. The main difference is that for *a-priori* techniques broad assumptions are made on the integrand without relying on information from renderings of the current scene. The cited *a-priori* approaches describe ways for constructing offline optimized point sets/sequences. We denote the method in [Georgiev and Fajardo 2016] as BNDS (blue-noise dithered sampling), the method in [Heitz et al. 2019] as HBS (Heitz-Belcour Sobol), and the histogram and permutation method in [Heitz and Belcour 2019] as HBH and HBP respectively (Heitz-Belcour histogram/permutation).

Energy. HBH/HBP both rely on a blue noise dither matrix optimized while using a Gaussian kernel (through void-and-cluster [Ulichney 1993]). This kernel corresponds to the kernel in our framework $\boldsymbol{g}$. The optimization of this dither matrix happens offline unlike in our iterative energy minimization algorithms. This imposes multiple restrictions while allowing for a lower runtime. On the other hand, the dither matrices in HBS and BNDS are optimized with respect to empirically motivated energies that cannot be related directly to what is used as energy in HBH and HBP. In the case of BNDS the energy does not even introduce an implicit integrand, and instead it is devised to represent a whole class of integrands. We propose to substitute those empirically motivated energies with a modified version of our energy. This allows an intuitive interpretation and relating *a-posteriori* approaches to *a-priori* approaches.

Search space. Another notable difference constitute the search spaces on which the different approaches operate. HBH selects a subset from a set of precomputed samples in each pixel, HBP permutes the assignment of sample sets to pixels, BNDS directly modifies the set of samples in each pixel, and HBS considers a search space made up of scrambling and ranking keys for a Sobol sequence.

Working on the space of scrambling and ranking keys guarantees the preservation of the desirable integration qualities of the Sobol sequence used, and it should be clear that other methods can also be restricted to such a space. Clearly, a search space restriction diminishes the achievable blue noise quality. On the other hand, it makes sequences more robust to integrands for which those were not optimized.

5 A-POSTERIORI APPROACHES

In this section we analyze the permutation based approach (HBP) and the histogram sampling approach (HBH) proposed in [Heitz and Belcour 2019]. The two methods can be classified as dither matrix halftoning methods in our framework, that operate on a *horizontal* and *vertical* search space respectively. We make the approximations and assumptions necessary to get from our general formulation to HBP/HBH explicit.

We also note that *a-posteriori* methods lead to solutions that adapt to the current render by exploiting known information (e.g. previously rendered data, auxiliary buffers). They can generally produce better results than *a-priori* methods.

Both HBP and HBH rely on a blue noise dither matrix $\boldsymbol{B}$. Let $\boldsymbol{B}$ be the optimized blue noise dither matrix resulting from the minimization of $E(\boldsymbol{B}) = \|\boldsymbol{g} * \boldsymbol{B}\|_2^2$ over a suitable search space. The kernel $\boldsymbol{g}$ is the one used to generate the blue noise images for HBP/HBH. That is, the Gaussian kernel in the void-and-cluster method [Ulichney 1993]. Our analysis does not rely on the kernel being a Gaussian, or on the void-and-cluster optimization, this is simply the setting of the HBP/HBH method. In the more general setting any kernel is admissible.

5.1 Sorting step for the permutation approach

The permutation approach [Heitz and Belcour 2019] consists of two main parts: sorting (optimization), and retargeting (correcting for mispredictions). The sorting step in HBP can be interpreted as minimizing the energy:

$$E_{HBP}(\pi) = \|\pi(\boldsymbol{Q}) - f_2(\boldsymbol{B})\|_2^2, \forall f_2 : a < b \implies f_2(a) < f_2(b). \quad (51)$$

A global minimum of the above energy is achieved for a permutation π that matches the order statistics of $\boldsymbol{Q}$ and $\boldsymbol{B}$. Thus our goal would be to get from the minimization of:

$$E(\pi) = \|\boldsymbol{g} * (\boldsymbol{Q}(\pi(\boldsymbol{S})) - \boldsymbol{I})\|_2^2 = \|\boldsymbol{g} * \boldsymbol{\epsilon}(\pi(\boldsymbol{S}))\|_2^2, \quad (52)$$

to the minimization of Eq. (51) over a suitable search space (in practice it is limited to permutations within tiles).

We successively bound the error, while introducing the assumptions implicit to the HBP method. The bounds are not tight, however, the different error terms that we consider illustrate the major sources of error due to the approximation of the more general energy (Eq. (52)) with a simpler one (Eq. (51)). The substitution of the kernel convolution $\boldsymbol{g} * \cdot$ by a difference with a blue noise mask $\boldsymbol{B}$ restricts the many possible blue noise error distributions towards which $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ can go with a single one: $\boldsymbol{B}$. A global minimizer of the new simplified energy can thus be found by just sorting.

The closer the distributions of $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ and $\alpha\boldsymbol{B}, \alpha > 0$ are locally, the lower this restriction error can be made. Notably, for a close to linear relationship between the samples and the integrand, and sufficiently many pixels, $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ and $\alpha\boldsymbol{B}$ can be matched closely in practice. A different way to reduce the approximation error is to introduce a sufficient amount of different blue noise images and pick the one that minimizes the error. We start with the original energy (Eq. (52)) and bound it through terms that capture the main assumptions on which the model relies:

$$\begin{aligned} &\|\boldsymbol{g} * \boldsymbol{\epsilon}(\pi(\boldsymbol{S}))\|_2 = \\ &\min_{f_2} \|\boldsymbol{g} * (\boldsymbol{\epsilon}(\pi(\boldsymbol{S})) - f_2(\boldsymbol{B}) + f_2(\boldsymbol{B}))\|_2 \leq \\ &\min_{\alpha>0, f_2} \|\boldsymbol{g}\|_1 \|\boldsymbol{\epsilon}(\pi(\boldsymbol{S})) - f_2(\boldsymbol{B})\|_2 \\ &+ \|\boldsymbol{g} * (f_2(\boldsymbol{B}) - \alpha\boldsymbol{B} + \alpha\boldsymbol{B})\|_2 \leq \\ &\min_{\alpha>0, f_2} \|\boldsymbol{g}\|_1 \|\boldsymbol{\epsilon}(\pi(\boldsymbol{S})) - f_2(\boldsymbol{B})\|_2 \\ &+ \|\boldsymbol{g}\|_1 \|(f_2(\boldsymbol{B}) - \alpha\boldsymbol{B})\|_2 + \alpha\|\boldsymbol{g} * \boldsymbol{B}\|_2. \end{aligned} \quad (53)$$

In the above, f_2 is taken over the space of all strictly monotonically increasing functions, and $\alpha > 0$ is a real value used to provide an amplitude matching between $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ and $\boldsymbol{B}$ (this allows for the second term to go to zero as the pointwise error goes to zero).

5.1.1 Third error term. We note that $\boldsymbol{B}$ is precomputed offline in order to approximately minimize $E(\boldsymbol{B}) = \|\boldsymbol{g} * \boldsymbol{B}\|_2$. Thus, the third term reflects the quality of the blue noise achieved with respect to $\boldsymbol{g}$ in the offline minimization. This error can be made small without a performance penalty since the optimization is performed offline. We factor out a multiplicative scaling factor $\alpha > 0$ in the blue noise quality term, to allow for the second term to go to zero. With this change, we can consider $\boldsymbol{B}$ to be normalized in the range $[-1, 1]$ and we can encode the scaling in α.

5.1.2 Second error term. The second term reflects the error introduced by substituting a large search space (many local minima) with a small search space. It introduces the first implicit assumption of HBP by relating the first and third error terms (by using f_2 and α respectively) through the second error term. The assumption is that there exists a permutation for which $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ can be made close to $\alpha\boldsymbol{B}$, which would make the second term small. This holds in practice if the pixel-wise error is zero on average (unbiased estimator within each pixel), and we have a sufficiently large resolution/tiles: which results in a higher probability that pixels from $\boldsymbol{\epsilon}(\pi(\boldsymbol{S}))$ can match $\boldsymbol{B}$ well. Then the term $\|\boldsymbol{g}\|_1 \|f_2(\boldsymbol{B}) - \alpha\boldsymbol{B}\|_2$ can be made small. We note that this is a generalization of the third optimality condition in [Heitz and Belcour 2019] (*correlation-preserving integrand*) since an integrand linear in the samples can also better match $\boldsymbol{B}$ provided enough pixels. For a linear integrand the optimal f_2 is also a linear function (ideal correlation between samples and integrand). The main difference between a linear integrand and a nonlinear/discontinuous one, is the amount of sample sets/pixels necessary to match $f_2(\boldsymbol{B})$ well, given an initial white noise samples' distribution. So in practice there are 4 factors directly affecting the magnitude of the second term: the number of considered blue noise images, the size of the tiles, the correlation between samples and integrand (accounted for by f_2), the bias/consistency of the estimators.

We note that the number of considered pixels depends on the tile size in HBP, and the practical significance of this has been demonstrated through a canonical experiment in the main paper.

5.1.3 First error term. Before we proceed we need to further bound the first error term by substituting $\boldsymbol{Q}(\pi(\boldsymbol{S}))$ by $\pi(\boldsymbol{Q}(\boldsymbol{S}))$. As discussed in the main paper, this is achieved by introducing a difference term $\boldsymbol{\Delta}(\pi) = \boldsymbol{Q}(\pi(\boldsymbol{S})) - \pi(\boldsymbol{Q}(\boldsymbol{S}))$, and then $\sqrt{E_{HBP}}$ is recovered. The error there can be made arbitrarily small through f_2 (it is accounted for in the second term). Thus we only need to study the remaining error due to $\boldsymbol{\Delta}$. In the case of HBP, $\boldsymbol{\Delta}$ is approximated by non-overlapping characteristic functions in each tile ($d(x, y) = \infty$, for x, y in different tiles). This means that the approximation error is zero within each tile if the integrands are the same within the tile and permutations act only within the tile, since $\boldsymbol{\Delta}(\pi) = \boldsymbol{0}$. On the other hand, if this assumption is violated, mispredictions occur, usually resulting in white noise.

5.1.4 $\boldsymbol{\Delta}$ term. HBP partitions screen space into a several tiles $\mathcal{R}_1$, ..., $\mathcal{R}_K$, and permutations are only over the pixel values in a tile. Having the partition induced by the tiling we can bound the first term:

$$\|\boldsymbol{\epsilon}(\pi(\boldsymbol{S})) - f_2(\boldsymbol{B})\|_2 \leq \sum_{k=1}^{K} \|\boldsymbol{\epsilon}_k(\pi_k(\boldsymbol{S}_k)) - f_2(\boldsymbol{B})\|_2. \tag{54}$$

Since additionally the permutations are optimized for the pixel values instead of the sample sets (which saves re-rendering operations), then there is an assumption that within each tile $\mathcal{R}_k$ the following holds (we denote $\boldsymbol{A}_k = \boldsymbol{A}|_{\mathcal{R}_k}$) :

$$\boldsymbol{Q}_k(\pi_k(\boldsymbol{S}_k)) = \pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)). \tag{55}$$

Consequently it follows that $I_i = I_j, \forall i, j \in \mathcal{R}_k$.

This assumption can be identified with the 4-th optimality condition proposed in [Heitz and Belcour 2019]: *screen-space coherence*. As discussed, the search space restriction to the tiles corresponds to an approximation of the $\boldsymbol{\Delta}$ term in our framework by characteristic functions: $d_k(x, y) = \infty, x \in \mathcal{R}_k, y \notin \mathcal{R}_k$ and $d_k(x, y) = 0, x, y \in \mathcal{R}_k$. To account for the actual error when the assumption is violated we introduce an additional error term per tile: $\boldsymbol{\Delta}_k = \boldsymbol{Q}_k(\pi_k(\boldsymbol{S}_k)) - \pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k))$, then we have the bound:

$$\begin{gathered}\|\boldsymbol{\epsilon}_k(\pi_k(\boldsymbol{S}_k)) - f_2(\boldsymbol{B}_k)\|_2 = \\ \|\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)) - \boldsymbol{I}_k - f_2(\boldsymbol{B}_k) + \boldsymbol{\Delta}_k\|_2 \leq \\ \|\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)) - \boldsymbol{I}_k - f_2(\boldsymbol{B}_k)\|_2 + \|\boldsymbol{\Delta}_k\|_2.\end{gathered} \tag{56}$$

This means that even if all of the previous error terms are made small, including $\|\pi_k(\boldsymbol{\epsilon}_k(\boldsymbol{S}_k)) - f_2(\boldsymbol{B}_k)\|_2$, the error may still be large due to $\|\boldsymbol{\Delta}\|_2$. We refer to a large error due to the delta term as *misprediction* - that is, a mismatch between the predicted error distribution from the minimization of $\|\pi_k(\boldsymbol{\epsilon}_k(\boldsymbol{S}_k)) - f_2(\boldsymbol{B}_k)\|_2$ and the actual error distribution resulting from the above permutation applied to $\boldsymbol{\epsilon}_k(\pi_k(\boldsymbol{S}_k))$. The best way to identify mispredictions is to compare the predicted image $\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k))$ and the image rendered with the same permutation for the sample sets $\boldsymbol{Q}_k(\pi_k(\boldsymbol{S}_k))$. A misprediction occurring means that the assumption made to approximate $\boldsymbol{\Delta}$ was incorrect ($\boldsymbol{\Delta}_k \neq \boldsymbol{0}$ for some tile $\mathcal{R}_k$), equivalently the optimality condition of *screen-space coherence* is not satisfied.

Avoiding mispredictions. In practice mispredictions often occur for larger tile sizes, since it is hard to guarantee that the integrand remains similar over each tile. On the other hand, larger tiles allow for a better blue noise as long as $\boldsymbol{\Delta}_k = 0$ in each tile, thus larger tiles are desirable. The method fails even more often near edges, since even for small tile sizes it allows swapping pixels over an edge. A straightforward improvement involves partitioning the domain by respecting edges. More involved methods may take into account normals, depth, textures, etc.

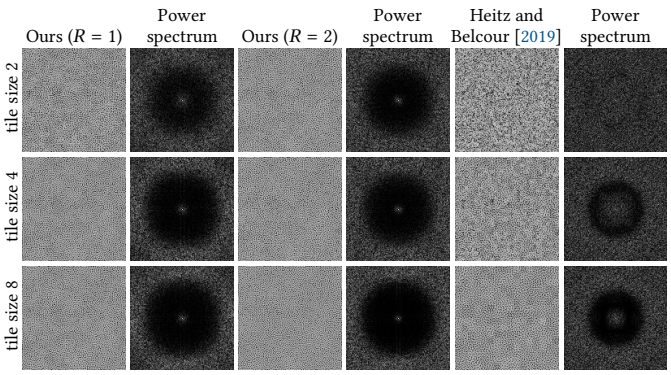


Fig. 2. Here we showcase the effect of tile size on the quality of blue noise. We also demonstrate the effect of a larger search neighborhood R in our optimization Alg. 2. For our case, we consider disk neighborhoods so that they are contained within Heitz and Belcour's tiles in terms of size, but they can also overlap due to our formulation. From left-to-right, the input white noise texture is optimized using our relocation algorithm. The last two columns are from Heitz and Belcour's [2019] method. The corresponding power spectra of these optimized images (128 × 128) are also shown.

5.1.5 E_{HBP} error term. The final step involves the minimization of the energy in Eq. (56). Since different tiles do not affect each other the minimization can be performed per tile (we adopt the assumption from HBP $\boldsymbol{\Delta}_k = \boldsymbol{0}$):

$$\begin{gathered}\pi_k^* \in \arg\min_{\pi_k} \|\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)) - \boldsymbol{I}_k - f_2(\boldsymbol{B}_k)\|_2 = \\ \arg\min_{\pi_k} \|\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)) - f_2(\boldsymbol{B}_k)\|_2^2.\end{gathered} \tag{57}$$

We have dropped the term $\boldsymbol{I}_k$ since it does not affect the set of minimizers ($\boldsymbol{I}_k$ is assumed constant in each tile). As discussed in Eq. (51), a global minimum is given by matching the order statistics of $\boldsymbol{Q}_k$ to the order statistics of $f_2(\boldsymbol{B})$) (we note that the order statistics of $\boldsymbol{B}_k$ do not change from the application of f_2 since it is a strictly increasing function). This is equivalent to performing the sorting pass described in [Heitz and Belcour 2019]. A minor optimization would be to pre-sort $\boldsymbol{B}$ and instead store the sorted indices.

Tiling effect. In Fig. 2 we compare the effect of the tile size. In our approach, the "tiles" can be defined per pixel, can have arbitrary shapes, and are overlapping, the last being crucial for achieving a good blue noise distribution. We consider white-noise with mean 0.5 (which is an ideal scenario for Heitz and Belcour's method) and compare various tile sizes. For a fair comparison, our tile radius r corresponds similar tile-size in the permutation [2019] approach. The power-spectrum profiles confirm the better performance of our method. Retargeting [2019] cannot improve the quality of the permutation approach either, since no misprediction can occur ($\boldsymbol{\Delta} = 0$). The adverse effect of tiling is exacerbated in practice since, for

images which are not smooth enough in screen space, tiles of smaller sizes need to be considered.

Custom surrogate. The $\boldsymbol{I}_k$ term does not need to be assumed constant in fact. If it is assumed constant, that is equivalent to picking a tile-constant surrogate, however, a custom surrogate may be provided instead. Then one would simply minimize the energy:

$$\|\pi_k(\boldsymbol{Q}_k(\boldsymbol{S}_k)) - (\boldsymbol{B}_k + \boldsymbol{I}_k)\|_2^2. \tag{58}$$

The energy has a different minimizer than the original HBP energy, but the global minimum can be found efficiently through sorting once again.

5.2 HBP retargeting

The retargeting pass in HBP achieves two things. It introduces new possible target solutions through new blue noise images, and it corrects for mispredictions. The first is not so much a result of the retargeting, as it is of varying the blue noise image every frame. Ideally several blue noise images would be considered in a single frame, and the best image would be chosen per tile (in that case one must make sure that there are no discontinuities between the blue noise images' tiles) in order to minimize the second term in Eq. (53). Instead, in HBP this is amortized over several frames.

The more important effect of retargeting is correcting for mispredictions, by transferring the recomputed correspondence between sample set and pixel value (achieved through rerendering) to the next frame. This allows reducing the error due to the approximation of $\boldsymbol{\Delta}$ (when the piecewise-tile constancy assumption on the integrand is violated). Note however, that this is inappropriate if there is a large temporal discontinuity between the two frames.

Implementation details. Retargeting requires a permutation that transforms the blue noise image in the current frame into the blue noise image of the next frame [Heitz and Belcour 2019]. This permutation is applied on the optimized seeds to transfer the learned correspondence between sample sets and pixel values to the next frame. Implicitly, this transforming permutation also relies on a screen space integrand similarity assumption, since there is no guarantee that the corresponding values from the swap will match, possibly incurring a misprediction once again (it can be modeled by an additional $\boldsymbol{\Delta}$ term). In HBP [Heitz and Belcour 2019] the maximum radius of travel of each pixel in the permutation is set to 6 pixels. This has a direct effect on the approximation of $\boldsymbol{\Delta}$, as the travel distance of a pixel is allowed to extend beyond the original tile bounds. In the worst case scenario a pixel may allowed to travel a distance of $\sqrt{t_x^2 + t_y^2} + 6$ pixels, where t_x, t_y are the dimensions of the tiles. An additional error is introduced since the retargeting pass does not produce the exact blue noise image used in the next frame, but some image that is close to it [Heitz and Belcour 2019]. This seems to be done purely from memory considerations since it allows one blue noise image to be reused by translating it toroidally each frame to produce the blue noise image for the next frame.

Relationship to our horizontal approach. Our horizontal approach does not require a retargeting pass. It can directly continue with the optimized sample sets and pixel values from last frame. There is also no additional travel distance for a matching permutation as in retargeting, which further minimizes the probability of misprediction. Thus, it inherently and automatically produces all of the advantages of retargeting while retaining none of its disadvantages.

5.3 Histogram sampling approach

The histogram sampling approach from Heitz and Belcour can be interpreted as both a dithering and a sampling method. We study the dithering aspect to better understand the quality of blue noise achievable by the method.

Algorithm analysis. The sampling of an estimate in each pixel by using the corresponding mask value to the pixel can be interpreted as performing a mapping of the mask's range and then quantizing to the closest estimate. In HBH each estimate is equally likely to be sampled (if a random mask is used), which implies a transformation that maps equal parts of the range to each estimate. Let $Q_{k,1}, \ldots, Q_{k,N}$ be the greyscale estimates in pixel k sorted in ascending order. Let the range of the blue noise mask be in [0,1]. Then the range is split into N equal subintervals: $[0, \frac{1}{N}), \ldots, [\frac{N-1}{N}, 1]$ which respectively map to $[Q_1, \frac{Q_1+Q_2}{2}), \ldots, [\frac{Q_{i-1}+Q_i}{2}, \frac{Q_i+Q_{i+1}}{2}), \ldots, [\frac{Q_{N-1}+Q_N}{2}, Q_N]$. If the quantization rounds to the closest estimate, then the above mapping guarantees the desired behavior. We note that since the estimates in each pixel can have different values, the mapping for each pixel may be different. We will denote the above mapping through $\boldsymbol{f}$. Then the mapping plus quantization problem in a pixel k may be formulated as:

$$\min_{i \in \{1,\ldots,N\}} |Q_{k,i} - f_k(B_k)|. \tag{59}$$

Note that the minimization in each pixel is independent, and it aims to minimize the distance between the estimates and the remapped value from the blue noise mask. If the set of estimates are assumed to be the same across pixels, and are also assumed to be spaced regularly, then f is only a linear remapping, which effectively transfers the spectral properties of $\boldsymbol{B}$ onto the optimized image. Notably, the former is the *screen-space coherence* assumption from HBP, while the latter is the *correlation-preserving integrand* assumption. Thus we have seen that for optimal results the HBH method relies on exactly the same assumption as the HBP method (while our *vertical* iterative minimization approach lifts both assumptions).

Disadvantages. One of the key points is that the error distribution and not the signal itself ought to ideally be shaped as $\boldsymbol{B}$. This is actually the case even in the above energy. From the way $\boldsymbol{f}$ was chosen it follows that the surrogate is equivalent to $\boldsymbol{f}(0.5)$ which can be identified as the image made of the median of the sorted estimates within each pixel. This is the case since if the target surrogate of $\boldsymbol{B}$ (during the offline optimization) was assumed to be 0.5, then after the mapping it is $\boldsymbol{f}(0.5)$. Generally, this is a very bad surrogate in the context of rendering, and it generally increases the error compared to the averaged image, making the method impractical.

Another notable disadvantage is that all estimates are considered with an equal weight. This means that outliers are as likely to be picked as estimates closer to the surrogate. This results in fireflies appearing even when those were not present in the averaged imaged.

Compared to classical halftoning, where only the *closest* lower and upper quantization levels are considered, HBH does not minimize the magnitude of the error to the surrogate.

Finally, the two assumptions of: *screen-space coherence* and *correlation-preserving integrand*, generally do not hold in practice. Estimates cannot be assumed to match between pixels (especially if samples are taken at random), and they cannot be assumed to be uniformly distributed, which implies that $\boldsymbol{f}$ is not linear. This greatly impacts the quality of the result, especially if it is compared to *adaptive* approaches such as our *vertical* error diffusion approach and our iterative minimization techniques (see the experiments in the main paper).

Generalization. The method can be generalized to take a custom surrogate instead of the one constructed by the median of the estimates within each pixel. This is achieved by splitting the per pixel set of estimates into two parts: (greyscale) estimates greater than the value of the (greyscale) surrogate in the current pixel, and estimates lower than it. Then the mapping f_k for the current pixel k maps values in $[0, 0.5)$ to the lower set, and values in $[0.5, 1]$ to the higher set, such that $f_k(0.5) = I_k$. The original method is recovered if the surrogate is chosen to be the implicit one for the original histogram sampling method and if the appropriate corresponding mapping $\boldsymbol{f}$ is kept.

The approach can be extended further by setting different probabilities for the different estimates. The original histogram sampling method correspond to setting the same probability for sampling every estimate, equivalently: equal sized sub-intervals from $[0, 1]$ map to each estimate. Classical dither matrix halftoning can be interpreted as setting an equal probability for the closest to the surrogate upper and lower estimates, while every other estimate gets a zero probability. Equivalently: equal sub-intervals from $[0, 1]$ map to the two aforementioned estimates while no part of the interval maps to the remaining estimates. Generally a custom probability can be assigned to each estimate: $p_1, ..., p_N$, by having the intervals $[0, p_1), ..., [\sum_{k=1}^{N-1} p_k, 1]$ map to $Q_1, ..., Q_N$ (after quantization). We note that an unbiased image can be recovered only if there is a map to every estimate.

6 A-PRIORI APPROACHES

We discuss current state of the art *a-priori* approaches [Georgiev and Fajardo 2016; Heitz et al. 2019] and their relation to our framework, as well as insights regarding those.

6.1 HBS

In Heitz et al.'s work, a scrambling energy and a ranking energy have been proposed (note that those energies are maximized and not minimized):

$$E_s = \sum_{i,j} \exp\left(-\frac{\|i-j\|_2^2}{2\sigma^2}\right) \|E_i - E_j\|_2^2 \tag{60}$$

$$E_r = \sum_{i,j} \exp\left(-\frac{\|i-j\|_2^2}{2\sigma^2}\right) (\|E_i^1 - E_j^1\|_2^2 + \|E_i^2 - E_j^2\|_2^2) \tag{61}$$

$$E_i = (e_{1,i}, \ldots, e_{T,i}) \tag{62}$$

$$e_{t,i}(S_i) = \frac{1}{|S_i|} \sum_{k=1}^{|S_i|} f_t(p_{i,k}) - \int_{[0,1]^D} f_t(x)\, dx \tag{63}$$

$$S_i = \{p_{i,1}, \ldots, p_{i,M_i}\}. \tag{64}$$

The upper indices in E_i^1, E_i^2 indicate that the two energies are evaluated with different subsets of the sample set S_i in the pixel i. The f_t are taken from an arbitrary set of functions (in the original paper those are random Heaviside functions). The described form of the energies has been partially motivated by the energy in [Georgiev and Fajardo 2016]. This does not allow for a straightforward interpretation or a direct relation to the (implicit) energy used for *a-posteriori* approaches in [Heitz and Belcour 2019].

Scrambling energy. We modify E_s in order to relate it to the energy in our framework and to provide a meaningful interpretation:

$$E_s' = \sum_{t=1}^{T} w_t \|\boldsymbol{g} * \boldsymbol{Q}_t(\boldsymbol{S}) - \boldsymbol{I}_t\|_2^2, \tag{65}$$

$$Q_{t,i}(\boldsymbol{S}) = \frac{1}{|S_i|} \sum_{k=1}^{|S_i|} f_t(p_{i,k}),\ I_{t,i} = \int_{[0,1]^D} f_t(x)\, dx. \tag{66}$$

We have relaxed the Gaussian kernel to an arbitrary kernel $\boldsymbol{g}$ and absorbed it into the norm. More importantly we have removed the heuristic dependence of error terms on their neighbors, and instead the coupling happens through the kernel itself. Finally, we have introduced weights $w_1, \ldots, w_T$ that allow assigning different importance to different integrands. Thus, this is a weighted average of our original energy applied to several different integrands, matching our *a-priori* approach (Eq. (1)). Through this formulation a direct relationship to the *a-posteriori* methods can be established, and it can be motivated in the context of both the human visual system and denoising. Particularly, the scrambling energy E_s' is over the space of scrambling keys, which allow permuting the assignment of sample sets. This is in fact the *horizontal* setting from our formulation in the main paper. The space can be extended further if the scrambling keys in each dimension are different (as in HBS). The same can be done in *a-posteriori* methods, if the optimization is performed in each dimension as discussed in Section 2.

Ranking energy. The ranking keys in HBS describe the order in which samples are consumed. This is useful for constructing progressive *a-priori* methods. Notably, the order in which samples will be introduced can be optimized. Having a sequence of sample sets in each pixel: $S_{i,1} \subset \ldots \subset S_{i,M} \equiv S_i$ and respectively the images formed by those: $\boldsymbol{S}_1, \ldots, \boldsymbol{S}_M$, the progressive energy may be constructed as:

$$E'_r = \sum_{k=1}^{M} w_k \|\boldsymbol{g} * \boldsymbol{Q}(\boldsymbol{S}_k) - \boldsymbol{I}\|_2^2. \tag{67}$$

The quality at a specific sample count corresponding to $\boldsymbol{S}_k$ is controlled through the weight w_k. The original energy maximizing the quality of the full set is retrieved for $(w_1, \dots, w_{M-1}, w_M) = (0, \dots, 0, 1)$. Since the sample sets $S_i, \dots, S_{i,M}$ are optimized by choosing samples from S_i this can be seen as a *vertical* method. Finally, the ranking keys can also be defined per dimension, which can be related to *a-posteriori* methods through the suggested dimensional decomposition in Section 2.

6.2 Blue-noise dithered sampling energy

In Georgiev and Fajardo's work, in order to get an optimized (multi-channel) blue noise mask, the following energy has been proposed:

$$E(p_1, \dots, p_N) = \sum_{i \neq j} \exp\left(-\frac{\|i-j\|^2}{\sigma^2}\right) \exp\left(-\frac{\|p_i - p_j\|^{d/2}}{\sigma_s^2}\right), \tag{68}$$

which bears some similarity to the weights of a bilateral filter. In the above i, j are pixel coordinates, and p_i, p_j are d-dimensional vectors associated with i, j. Let the image formed by those vectors be $\boldsymbol{S}$. The energy aims to make samples p_i, p_j distant ($\|p_i - p_j\|$ must be large) if they are associated with pixels which are close ($\|i - j\|$ is small).

Relation to our framework. Even though the energy is heuristically motivated, we can very roughly relate it to our framework. The above energy implicitly assumes classes of integrands $\boldsymbol{Q}_1$, ..., $\boldsymbol{Q}_T$, such that close samples p_i, p_j are mapped to close values $Q_{i,t}(p_i)$, $Q_{j,t}(p_j)$, and distant samples are mapped to distant values. Notably, the form of the energy does not change over screen-space, so the same can be implied about the integrands. One such class is the class of bi-Lipschitz functions. The bound can be used to relate a modified version of the original energy, to an energy of the form:

$$E_{\boldsymbol{Q}_t} = \sum_{i \neq j} \exp\left(-\frac{\|i-j\|^2}{\sigma^2}\right) \exp\left(-\frac{C\|Q_{i,t}(p_i) - Q_{j,t}(p_j)\|^{d/2}}{\sigma_s^2}\right). \tag{69}$$

Thus, the original energy can indeed be interpreted as reasonable for a whole class of sufficiently smooth integrands, instead of an energy that works very well with one specific integrand.

A similar thing can be achieved in our framework, if the weighted energy is considered:

$$E'(\boldsymbol{S}) = \sum_{t=1}^{T} w_t \|\boldsymbol{g} * \boldsymbol{Q}_t(\boldsymbol{S}) - \boldsymbol{I}_t\|^2. \tag{70}$$

The kernel $\boldsymbol{g}$ can be a Gaussian with standard deviation σ, as in the original energy, or it can be relaxed to an arbitrary desired kernel. $\boldsymbol{Q}_1, \dots, \boldsymbol{Q}_T$ are representative integrands that satisfy the discussed smoothness requirements, and w_t are associated weights assigning different importance to the integrands. Finally, the reference images are given by the integrals $\boldsymbol{I}_t = \int_{[0,1]^d} \boldsymbol{Q}_t(x)\, dx$.

It should be clear that this is a weighted average constructed from the standard energy in our framework applied to a set of integrands. There are a number of benefits of such an explicit formulation. Most importantly, it allows for *a-priori* methods to be studied in the same framework as *a-posteriori* approaches. Additionally, explicit control is provided over the set of integrands and the kernel in a manner that allows for a straightforward interpretation.

Perceptual quality trade-off. While the energy of Georgiev and Fajardo is able to account for many different integrands, this is achieved at the cost of the perceptual quality of the produced patterns. We illustrate this in Fig. 4 by considering a constraint where 25% of all pixels have an error of +1 and 25% of all pixels have an error of -1.

For the experiment an initial white noise image is permuted using a brute force optimization with our energy from the main paper and the energy of Georgiev and Fajardo. One can see that the pattern resulting from our energy always decays faster under convolution. This can be explained by the fact that the bilateral filter-like energy forces nearby pixels to be as different as possible. This doesn't necessarily lead to the best results under convolution illustrated by Figure 5, but it is necessary in the setting of *a-priori* methods since not much information is assumed regarding the integrand.

We consider a more realistic example in Fig. 6 where the underlying signal is a sine function with vertically increasing frequency. We first degrade the signal with uniform white noise. To optimize the error distribution, we use our Kronecker kernel energy extension (eq. 11 from the main paper where $h = \delta$) that is given by:

$$E(\boldsymbol{Q}) = \|\boldsymbol{g} * \mathcal{T}(\boldsymbol{Q}) - \mathcal{T}(\boldsymbol{I})\|_2^2, \tag{71}$$

where $\mathcal{T}$ simply clamps values to $[0, 1]$. The result with our energy function matches better the original signal. This is perfectly in line with all of our results on realistic scenes presented in the main paper and the supplemental HTML.

6.3 Blue-noise dithered sampling algorithm

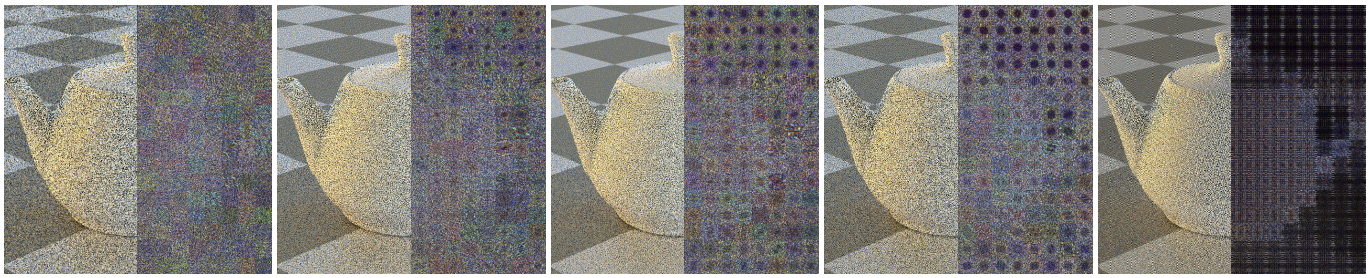
The second contribution of Georgiev and Fajardo's work is a sampler which relies on an image optimized with Eq. (68) and uses it to achieve a blue noise distribution of the rendering error. We summarize the algorithm and discuss some details related to it.

Algorithm. Let $\boldsymbol{B}$ be an image (with d-channels) optimized by minimizing Eq. (68) over a suitable search space. Let $\mathcal{P} = \{p_1, \dots, p_N\}$ be a sequence of d-dimensional points. Within each pixel i the sample set S_i is constructed, such that

$$p_j \in \mathcal{P} \implies p_{i,j} \in S_i : p_{i,j} = (p_j + B_i) \bmod 1. \tag{72}$$

The sequence $\mathcal{P}$ can be constructed by using various samplers (e.g., random, low-discrepancy, blue-noise, etc.). The construction of the new points for pixel i can be interpreted either as toroidally shifting the sequence $\mathcal{P}$ by B_i or equivalently as toroidally shifting the sequence $\{B_i, \dots, B_i\}$ by $\mathcal{P}$.

The sequences constructed within each pixel are used to estimate the integral in the usual manner. Since a finite number of dimensions d are optimized the suggestion is to distribute the constructed sequences over smoother dimensions, while other dimensions may use a standard sampler.



Random	Georgiev and Fajardo [2016]	Heitz et al. [2019]	Ours	Sobol
MSE: 0.118636	0.0921076	0.0787028	0.117336	0.178861
pMSE: 0.0170958	0.011277	0.00869183	0.0119757	0.0126065

Fig. 3. A comparison illustrating that even a sampling sequence formed by a stack of blue noise images (Ours) yields a good distribution (note the tiled error spectra). The integration error is higher however, degrading the quality. This is the case because the assumed integrand is far from linear in each dimension (see *Extension* in Section 6.3). The images use 4 samples per pixel, and the degradation of the spectral properties with the number of samples is clear for [Georgiev and Fajardo 2016] and even [Heitz et al. 2019], while it is not so much the case for Ours. This demonstrates that different methods offer a different trade-off between integration error and distribution for arbitrary integrands. Constraining the search space to using toroidal shifts or scrambling and ranking keys restricts the achievable blue noise distribution.

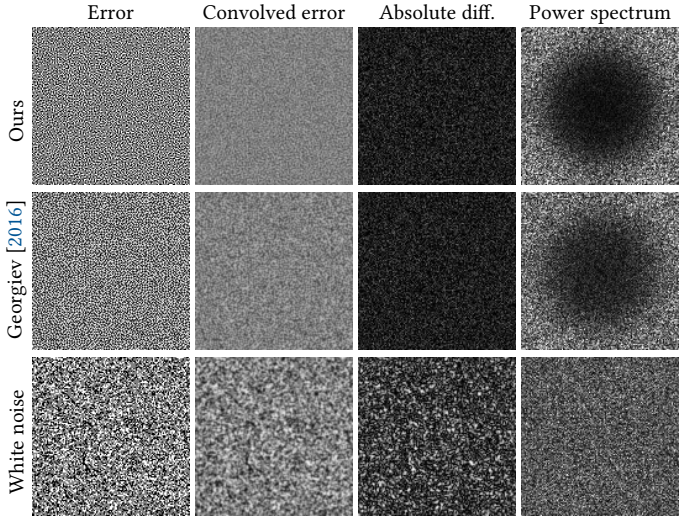


Fig. 4. We show an example demonstrating how our energy (top row) forms clusters where required so that the convolved error (second column) produces the best cancellation effect. The first column shows error images. Ours would converge to a grey (reference) image faster compared to the one using the energy in Eq. (68). The convolved images in the second column show the same behavior. The third column shows the absolute difference between the convolved error and the reference grey image (darker is better). The fourth column shows the error power spectra, with ours showing much better blue-noise characteristics than others.

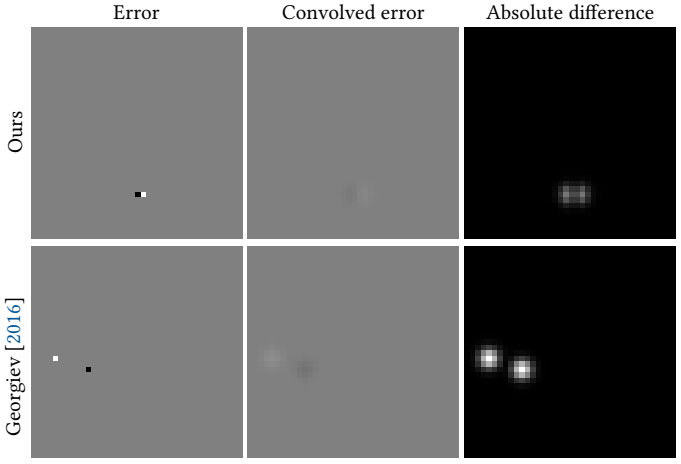


Fig. 5. We consider an example with 2 error pixels (+1 and -1). The first column shows the error images, the second column shows this error convolved with a gaussian kernel, and the third column shows the difference between the convolved error and the *reference* (constant) greyscale image. In the top row, our energy clusters these pixels such that they can cancel out each other's contribution under convolution. Georgiev and Fajardo's energy in the bottom row pushes these pixels farther away. The corresponding absolute difference (convolved error – constant grey image) images in the third column demonstrate that our energy makes the error converge faster to the constant greyscale image (darker is better).

Effect of the toroidal shift. Let us consider a linear one-dimensional integrand $f(q) = \alpha q + \beta$ that does not vary in screen space, and a sequence $\mathcal{P}$ with a single point p. Furthermore, if we assume $p = 0$, then the error is given by:

$$\boldsymbol{Q}(\boldsymbol{B}) - \boldsymbol{I} = \alpha\boldsymbol{B} + \boldsymbol{\beta} - \boldsymbol{I}. \tag{73}$$

Since $\boldsymbol{Q}$ does not vary in screen space, then $\boldsymbol{I}$ also does not. Then the power spectrum of the error (excluding the DC) matches the power spectrum of $\boldsymbol{B}$ up to the multiplicative factor α^2. Then, under the assumption that the integrand is linear, does not vary in screen space, and there is no toroidal shift, the power spectral properties of $\boldsymbol{B}$ are transferred ideally to the error.

On the other hand, if p is chosen to be non-zero, then the spectral characteristics of the image $((\boldsymbol{B} + p) \bmod 1)$ will be transferred instead. We have empirically verified that even with a very good quality blue noise image $\boldsymbol{B}$ the toroidal shift degrades its quality due to the introduced discontinuities. Thus, even in the ideal case

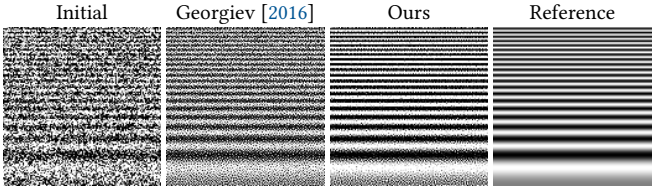


Fig. 6. A more realistic test with kernel $\boldsymbol{g}$ using $\sigma = 1/\sqrt{2}$. The signal is a sine function that increases in frequency along the vertical axis. Our method handles tone mapping and preserves well both the lower and higher frequencies present in the signal.

of a constant in screen space linear 1-D integrand, toroidal shifts degrade the quality.

Effect of using multiple samples. Let us consider the same integrand $f(q) = \alpha q + \beta$, which we have identified as being ideal for transferring the spectral characteristics of $\boldsymbol{B}$ to the error. And let us assume that we are given several samples: $\mathcal{P} = \{p_1, ..., p_N\}$, and we have constructed the sample set image $\boldsymbol{S}$ through toroidal shifts with $\boldsymbol{B}$. Then the error is:

$$Q_i(S_i) - I_i = \frac{\alpha}{N} \sum_{k=1}^{N} p_{k,i} + \beta - I_i. \tag{74}$$

The power spectrum of the error thus matches the power spectrum of the image $A_i = \sum_{k=1}^{N} p_{k,i}$ (excluding the DC) up to a multiplicative factor. For a random point sequence $\mathcal{P}$ the more points are considered, the closer to white noise $\boldsymbol{A}$ becomes. This is further exacerbated by the discussed discontinuities introduced by the toroidal shifts.

Extension. We have argued that both toroidal shifts and increasing the number of samples has a negative effect on transferring the spectral properties of $\boldsymbol{B}$ even in an ideal scenario. Naturally the question arises whether this can be improved. Our proposal is the direct optimization of point sets without the application of a toroidal shift.

For the discussed example this entails constructing a sequence of N images $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_N$ such that $\boldsymbol{A}_k = \sum_{j=1}^{k} \boldsymbol{B}_j$ is a blue noise image. Then the error has the (blue noise) spectral characteristics of $\boldsymbol{A}_k$ at each sample count (Fig. 3):

$$Q_i(B_{1,i}, \ldots, B_{k,i}) - I_i = \frac{\alpha}{k} \sum_{j=1}^{k} B_{j,i} + \beta - I_i. \tag{75}$$

7 ADDITIONAL RESULTS

In Fig. 8 we showcase tiled error spectra and SCIELAB images for several methods for the Wooden Staircase scene. SCIELAB acts very similarly to the pointwise squared error, as confirmed by its preference for methods that minimize the pointwise error. Thus it does not make for a very good metric for quantifying the quality of the distribution of the noise unlike VDP or the tiled error spectra.

REFERENCES

Mostafa Analoui and Jan P. Allebach. 1992. Model-based halftoning using direct binary search. In *Human Vision, Visual Processing, and Digital Display III*, Bernice E. Rogowitz (Ed.), Vol. 1666. International Society for Optics and Photonics, SPIE, 96 – 108. https://doi.org/10.1117/12.135959

Sagar Bhatt, John Sabino, John Harlim, Joel Lepak, Robert Ronkese, and Chai Wah Wu. 2006. Comparative study of search strategies for the direct binary search image halftoning algorithm. In *NIP22 (International Conference on Digital Printing Technologies)*. 244–247. International Conference on Digital Printing Technologies ; Conference date: 17-09-2006 Through 22-09-2006.

Iliyan Georgiev and Marcos Fajardo. 2016. Blue-Noise Dithered Sampling. In *ACM SIGGRAPH 2016 Talks* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 35, 1 pages. https://doi.org/10.1145/2897839.2927430

Eric Heitz and Laurent Belcour. 2019. Distributing Monte Carlo Errors as a Blue Noise in Screen Space by Permuting Pixel Seeds Between Frames. *Computer Graphics Forum* 38, 4 (2019), 149–158. https://doi.org/10.1111/cgf.13778

Eric Heitz, Laurent Belcour, V. Ostromoukhov, David Coeurjolly, and Jean-Claude Iehl. 2019. A Low-Discrepancy Sampler That Distributes Monte Carlo Errors as a Blue Noise in Screen Space. In *ACM SIGGRAPH 2019 Talks* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 68, 2 pages. https://doi.org/10.1145/3306307.3328191

James T. Kajiya. 1986. The Rendering Equation. *SIGGRAPH Comput. Graph.* 20, 4 (aug 1986), 143–150. https://doi.org/10.1145/15886.15902

Hiroaki Koge, Yasuaki Ito, and Koji Nakano. 2014. A GPU Implementation of Clipping-Free Halftoning Using the Direct Binary Search. In *Algorithms and Architectures for Parallel Processing*, Xian-he Sun, Wenyu Qu, Ivan Stojmenovic, Wanlei Zhou, Zhiyang Li, Hua Guo, Geyong Min, Tingting Yang, Yulei Wu, and Lei Liu (Eds.). Springer International Publishing, Cham, 57–70. https://doi.org/10.1007/978-3-319-11197-1_5

D.J. Lieberman and J.P. Allebach. 1997. Efficient model based halftoning using direct binary search. In *Proceedings of International Conference on Image Processing*, Vol. 1. 775–778 vol.1. https://doi.org/10.1109/ICIP.1997.648077

Robert A. Ulichney. 1993. Void-and-cluster method for dither array generation. In *Human Vision, Visual Processing, and Digital Display IV*, Jan P. Allebach and Bernice E. Rogowitz (Eds.), Vol. 1913. International Society for Optics and Photonics, SPIE, 332 – 343. https://doi.org/10.1117/12.152707

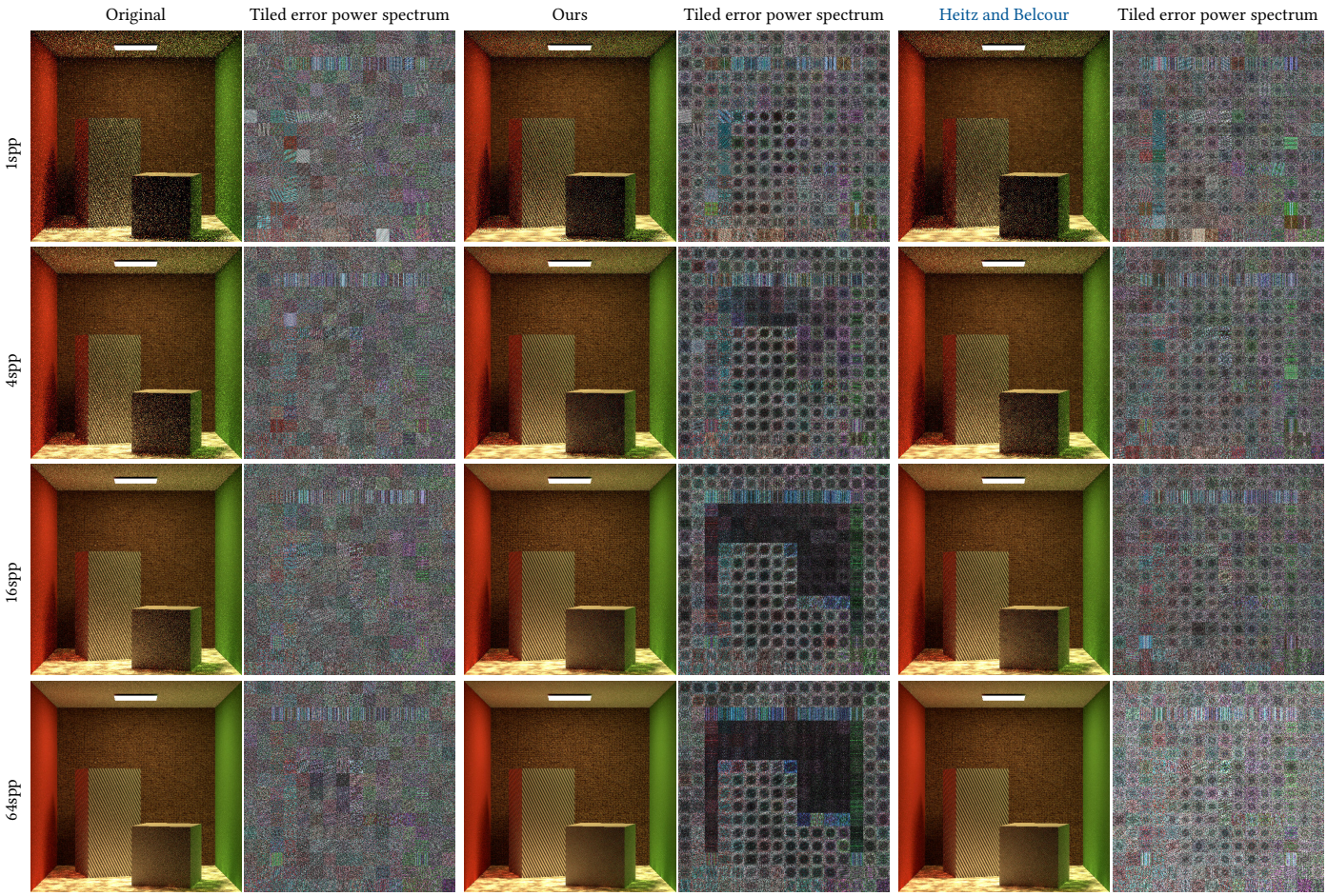
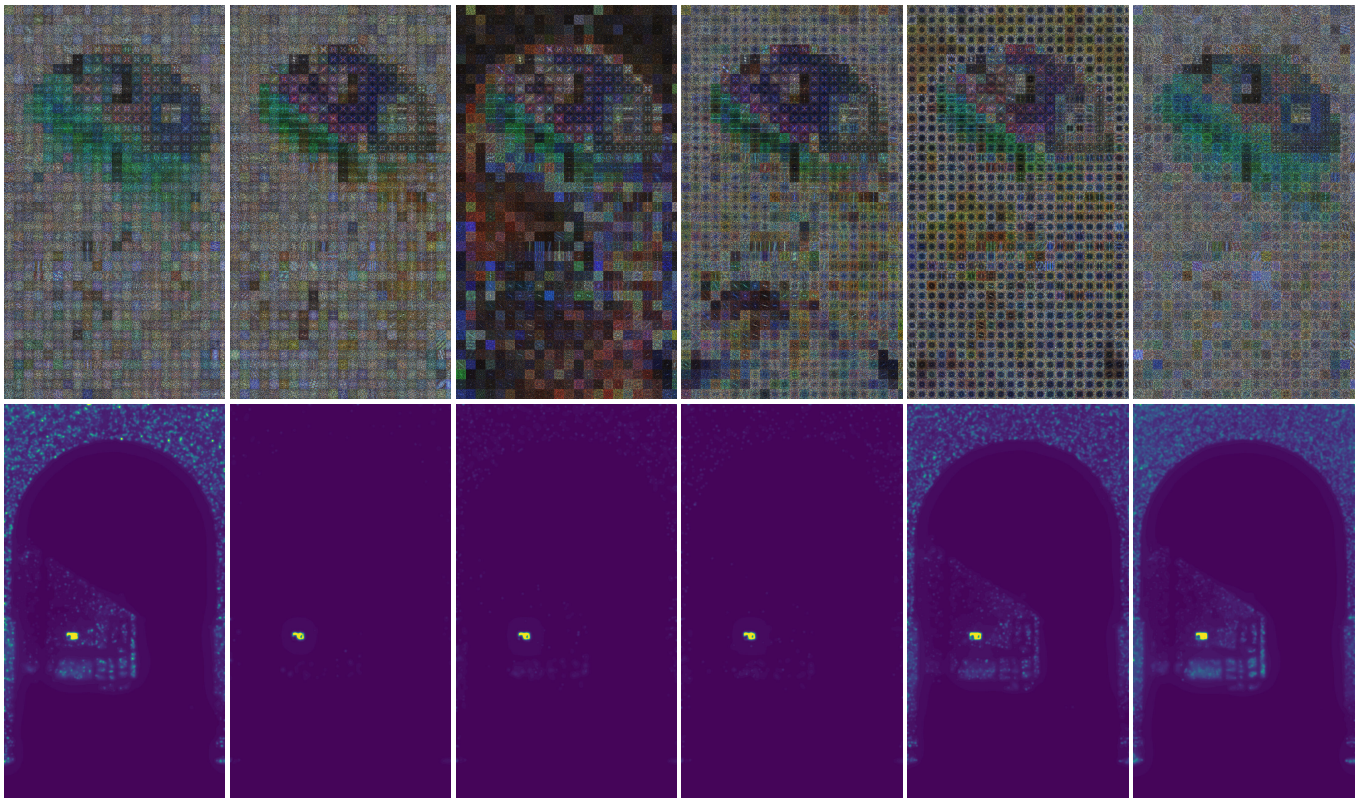


Fig. 7. The textured Cornell box scene is compared over different samples per pixel using our texture handling approach in Section 2. The improvements are visible for all spp over Heitz and Belcour's permutation approach using demodulation. This is especially apparent when comparing the tiled error spectra. Note that the spectra are not normalized to 1.



(a) Random (1 spp)	(b) Random (4-spp average)	(c) Dithering (ours)	(d) Error diffusion (ours)	(e) Optimization (ours)	(f) Histogram [2019]
MSE: 1.88×10^{-2}	8.88×10^{-3}	7.80×10^{-3}	$\mathbf{6.87 \times 10^{-3}}$	1.05×10^{-2}	1.88×10^{-2}
pMSE: 7.79×10^{-3}	5.59×10^{-3}	5.36×10^{-3}	5.08×10^{-3}	$\mathbf{4.45 \times 10^{-3}}$	6.69×10^{-3}

Fig. 8. In the main paper compare all vertical methods on the *Wooden staircase* scene. All of our methods achieve lower pMSE than the baseline (the averaged image), while the permutation method increases the error both in terms of MSE and pMSE. The tiled error power spectra images confirm the pMSE ranking and provide a visualization of the local pMSE distribution. We also show S-CIELAB error visualizations which suggest that pointwise error is heavily weighted in S-CIELAB, which does not make it a very good predictor for the perceptual quality related to the noise distribution, unlike HDR-VDP-2.