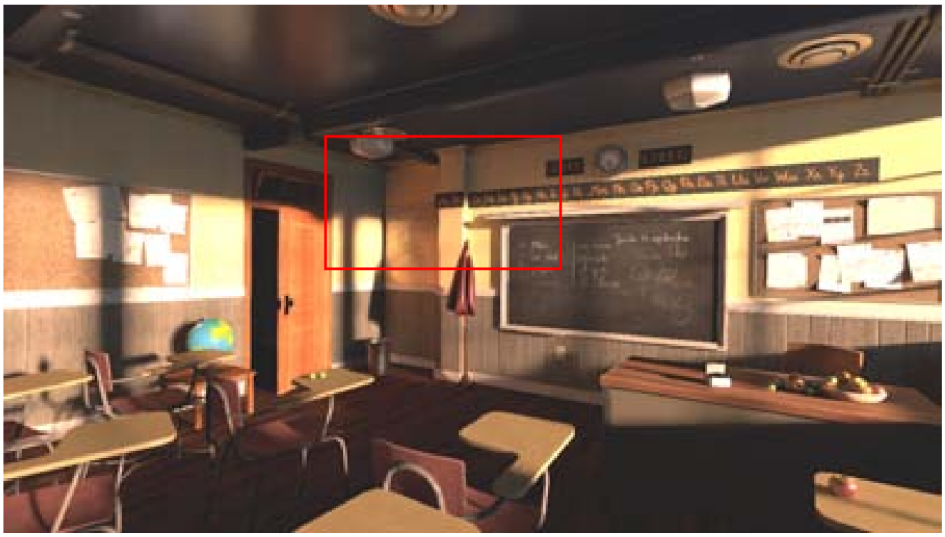
A-SVGF 自适应时空方差去噪滤波器

Dezeming Family

2021 年 5 月 6 日

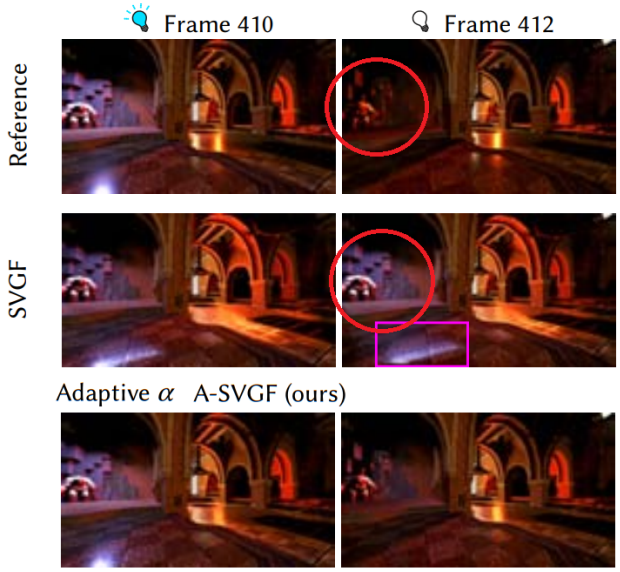


目录

一 介绍

前段时间看了篇论文 [1]，觉得这个技术应该是可以说比较流行的图形学去噪方法的一个综合改进版本了，所以打算好好分析一下技术要点。虽然技术来自论文，但是为了更好的描述和总结，本文每个字都是我亲手敲的，叙述方法和论文也有不同，对于读者来说应该可以作为脱离论文的独立学习教材了。

当前的时序滤波器会带来 artifacts，例如 ghosting 和 lag，ghosting 表示不应该出现的上一帧的信息出现在当前帧，而 lag 表示为如下图，当光源突然关闭时，SVGF 后处理的图像还会亮着，这是由于指数平均衰减系数 α 造成的。



A-SVGF 滤波器能够可靠地检测采样信号的突然变化，自适应地计算时序积累因子 α，从而删除过时的历史信息。

预备知识：SVGF[1]、TAA 时序超采样、交叉双边滤波器（联合双边滤波器）。

二 基本方法

我们从时间梯度的定义开始，并将其与随机抽样相结合。然后讨论了它们在稀疏位置上的有效计算，然后是一个重建步骤。由此产生的稠密梯度场用于计算每像素的权重。

基本方法部分其实非常简单，无非就是重投影筛选，为了详细描述发生了什么，下面会仔细进行记录。注意有些时候，我们选择的方法是没有太多理论根据的，有时候只是直觉，或者“启发式”（heuristic）的。

2 1 时间梯度

我们首先需要明确的一点是，计算时间梯度与 TAA 中的重投影获取有效的历史样本并不完全是同一回事。本节讲解了怎么获取时间梯度的方法，其实主要是细节问题，方法主观性也很大，不过为了完整性我还是详细介绍一下。

时序样本的复用需要进行时序重投影，定义第 i 帧的第 j 个像素的表面采样表示为 $G_{i,j}$，重投影有两种方法：前向投影和反向投影。

前向投影就是把先前帧的样本投影到当前帧：$\overrightarrow{G}_{i-1,j}$，反向投影就是把当前帧的样本投影到先前帧的位置：$\overleftarrow{G}_{i,j}$。

定义 f 为着色函数，则时序梯度可以表示为：

$$f_i(G_{i,j}) - f_{i-1}(\overleftarrow{G}_{i,j}) \ or \ f_i(\overrightarrow{G}_{i-1,j}) - f_{i-1}(G_{i-1,j}) \tag{二.1}$$

反向方法很直观，但重投影还需要知道先前帧的一些内容，比如 $\overleftarrow{G}_{i,j}$ 只有到了第 i 帧才知道，因此需要保留所有状态，来计算着色值。而前向方法只需要保留先前帧的着色样本，然后再在当前帧计算投影到当前帧位置以后的值即可，因此我们使用前向投影来定义时序梯度：

$$\delta_{i,\overrightarrow{j}} = f_i(\overrightarrow{G}_{i-1,j}) - f_{i-1}(G_{i-1,j}) \tag{二.2}$$

用一个现成的、类似于当前帧的样本替换着色样本 $f_i(\overrightarrow{G}_{i-1,j})$ 似乎看着很不错。但是，这通常会给采样位置引入子像素偏移。根据论文研究这种额外的空间偏移经常导致时间梯度的绝对高估，因此投影并计算着色值是有必要的。

2 2 稳定的随机采样

本节的意义在于保证更小的计算方差，因此会复用随机数，不过这里复用的是随机种子。

我们假设着色函数不止依赖于当前表面采样，还依赖于一个随机数 $\xi_{i,j}$：

$$\delta_{i,\overrightarrow{j}} = f_i(\overrightarrow{G}_{i-1,j}, \xi_{i,\overrightarrow{j}}) - f_{i-1}(G_{i-1,j}, \xi_{i-1,j}) \tag{二.3}$$

因此我们可以得到时序梯度的方差：

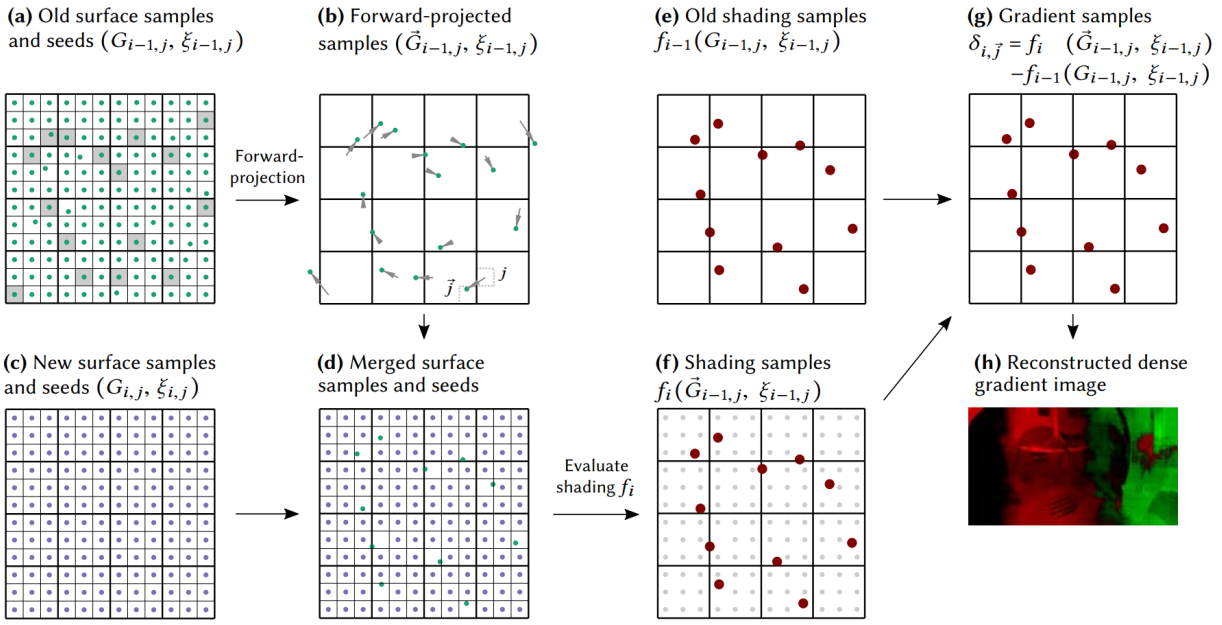
$$Var(\delta_{i,\overrightarrow{j}}) = Var(f_i(\overrightarrow{G}_{i-1,j}, \xi_{i,\overrightarrow{j}})) + Var(f_{i-1}(G_{i-1,j}, \xi_{i-1,j})) \tag{二.4}$$

$$-2 \cdot Cov(f_i(\overrightarrow{G}_{i-1,j}, \xi_{i,\overrightarrow{j}}), f_{i-1}(G_{i-1,j}, \xi_{i-1,j})) \tag{二.5}$$

为了更少的噪声（使用大的协方差来抵消方差），我们令 $\xi_{i,\overrightarrow{j}} := \xi_{i-1,j}$，即重投影获得先前帧使用的随机数。（至于为什么重用随机数更好，可以从直觉上感受一下）。我们不是存储完整的随机数序列，而是为每像素的伪随机数生成器存储随机种子。

2 3 构建梯度样本

先总体概述一下时序梯度的采样：



分开的表面和着色采样子集被重投影（a,b,e）。然后重投影的表面样本合并到新的可见性缓冲区（c,d）中。将重新投影的着色样本与新着色样本相结合产生梯度样本（f,g）。每 3x3 层最多有一个梯度样品。重建步骤将这些散射和噪声样本转换为稠密去噪梯度图像（h）。

在每一帧中，首先渲染一个新的可见性 buffer，产生新的随机数 seed（上图 c）。我们只重新利用我们的着色预算的一部分以稀疏地评估梯度样本，而不是提供每像素的时间梯度样本。我们每层定义 3x3 像素，从先前帧中随机选择一个像素 j，根据分层抽样，我们用 aliasing 来权衡时序不相干噪声（上图 a）。

然后我们使用前向投影，计算它们在当前帧的屏幕位置（上图 b），和 TAA 一样，我们根据深度 buffer 来丢弃那些在当前帧被遮住的重投影样本。其他的表面样本的 seed 倍融合到新的可见性 buffer 的合适像素位置 $\vec{j}$ 里（上图 d）。每层里（3x3 像素），我们只允许不超过 1 个梯度样本，然而，重投影可能会映射多个样本到一个层里，因此我们只融合那个首先被计算重投影到该层的样本到可见性 buffer 里。

我们从先前帧重投影着色样本，与前面方法相同，不使用插值（上图 e）。当前帧的着色函数计算重投影以后的样本的着色值（上图 f）。然后通过减法操作就能得到梯度样本（上图 g）。

表面样本重投影然后得到的着色样本对于新的一帧来说都是可用的着色样本。它们都是在一个像素内采样得到的可见性表面，只不过采样位置不在像素中心。因此，我们的策略不会在帧缓冲区中引入需要填充的间隙。然而，由新的表面样本和随机数产生的着色样本是优选的，如果重复使用随机数，则时间滤波器获得的新信息较少。此外，如果随机数中的某些是前一帧的残余，则它们的低差异性会减弱。层的尺寸为 2x2 时这些问题相当明显，但在 3x3 时，重投影样品的比例足够小。

虽然着色采样是密集的，但是梯度采样是稀疏的。通过构造，我们每个层最多有一个样本，但由于重投影和深度遮挡关系，可能样本之间有间隙。

2 4　重建时序梯度

我们当前只是有时序梯度的样本，但是它不但很稀疏，而且还有噪声，我们需要用重建方法来获得时序梯度的稠密的去噪估计。重建过程需要有效率、稳定、边缘保留，并且支持较大的滤波范围，因此使用边缘保留多孔小波滤波器，梯度重建使用联合双边滤波（joint-bilateral，或者称为交叉 cross 双边滤波）。

对于没有梯度样本的层，梯度设置为 0，这样就把稀疏的梯度样本填充好了，之后还要进行滤波。我们先回顾一下 SVGF[2] 中的照明估计的滤波方法，然后该方法会用到梯度估计上。

每个层的照明估计被初始化为该层全部样本的 luminance 的平均值，记为 $\hat{l}^{(0)}$，初始方差估计 $Var(\hat{l}^{(0)})$ 是一个层内的方差。我们令迭代次数 $k \in 0, 1, 2, 3, 4$，重建后的亮度即为：

$$\hat{l}^{(k+1)}(p) = \frac{\sum_{q\in\Omega} h^{(k)}(p,q) w^{(k)}(p,q) \hat{l}^{(k)}(q)}{\sum_{q\in\Omega} h^{(k)}(p,q) w^{(k)}(p,q)} \tag{二.6}$$

各个权重 w 的计算和 SVGF[2] 完全一样，这里就不再提了。

时序梯度使用和 luminance 相同的滤波方法（实践表明权重也和 luminance 滤波一样）：

$$\hat{\delta}^{(k+1)}(p) = \frac{\sum_{q\in\Omega} h^{(k)}(p,q)w^{(k)}(p,q)\hat{\delta}^{(k)}(q)}{\sum_{q\in\Omega} h^{(k)}(p,q)w^{(k)}(p,q)} \tag{二.7}$$

2 5　时序积累因子的控制

我们已经有了重建好的时序梯度，现在我们要控制时序滤波的因子了，首先加入标准化因子：

$$\Delta_{i,\overrightarrow{j}} = max(f_i(\overrightarrow{G}_{i-1,j}, \xi_{i-1,j}), f_{i-1}(G_{i-1,j}, \xi_{i-1,j})) \tag{二.8}$$

因为空的层梯度设置为了 0，并使用了联合双边滤波产生 $\hat{\Delta}_i(p)$，我们定义密度和标准化历史权重：

$$\lambda(p) := min\big(1, \frac{|\hat{\delta}_i(p)|}{\hat{\Delta}_i(p)}\big) \tag{二.9}$$

上式的意义在于让 λ 小于等于 1。之后我们定义自适应时序积累因子为：

$$\alpha_i(p) := (1-\lambda(p))\cdot\alpha + \lambda(p) \tag{二.10}$$

三　实际应用方法

虽然光线跟踪提供了一个简单的基于物理的解决方案来渲染阴影，但实时应用程序中的光线预算非常低。为了获得足够的质量，需要广泛的空间和时间滤波。使用 OptiX 渲染光线跟踪的软阴影，每像素一条光线，同时三个随机数用于选择一个面光源（需要一个随机数）和光源上的一个采样点（需要两个随机数），这些伪随机数使用蓝噪声抖动采样结合 Halton 序列采样。

对于重建，我们使用了 A-SVGF，我们的实现与最初的技术不同之处在于边缘感知多孔小波使用一个简单的 3×3box 过滤器。在变换的五个级别上，这将导致有效的过滤区域为 49×49 像素。我们发现，这几乎没有降低质量，但产生了一个显着的加速。

固定时间累积因子使用的是我们的自适应因子，我们可以减少动态部分中的 ghosting artifacts，并增加静态部分中的有效样本数。当历史样本被删除时，A-SVGF 在空间上进行方差估计。这往往会引入更强的空间模糊，但总体噪声仍然可以接受。

实时路径跟踪中，用于软阴影的分布式光线跟踪方法非常通用。例如，我们可以将其应用于渲染光泽反射、环境光遮挡和漫反射全局照明。与 SVGF 使用的路径跟踪器非常相似，主要的可见性仍然通过光栅化处理到可见性 buffer 中。我们追踪一条光线来收集一次间接照明的反弹。我们在第一次散射事件后应用路径空间正则化 [3]。此外，我们从两个表面交点（初始交点和反弹一次以后的交点）的每个表面跟踪阴影光线。我们再次使用 A-SVGF 进行重建。但是，我们不会将直接照明和间接照明分开进行单独过滤，这使得过滤更有效，但也更具挑战性。

提高递归自编码器（RAE）的时间稳定性：递归自动编码器提供了一种不同的方法来重建实时路径跟踪结果 [4]。我们将其与上面描述的光线和路径跟踪一起使用。虽然输出的单个帧看起来很吸引人，但一个常见的 artifact 在时间上是不稳定的，低频噪声。时序 anti-aliasing 无法移除此 artifact，因为它对大范围的噪声是不可见的。因此，我们的自适应递归自编码器（A-RAE）采用了附加的时间滤波来提高结果的稳定性。利用噪声路径跟踪输出进行时间梯度估计，从而得到自适应累积因子。然后利用该累积因子对递归自编码器的输出进行时间累积。

参考文献

[1] Schied C , Gradient Estimation for Real-time Adaptive Temporal Filtering Proceedings of the ACM on Computer Graphics and Interactive Techniques, 2018, 1(2): 1-16.

[2] Schied C , Salvi M , Kaplanyan A , et al. Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination[C]// High Performance Graphics. ACM, 2017.

[3] Anton S. Kaplanyan and Carsten Dachsbacher. 2013. Path Space Regularization for Holistic and Robust Light Transport. Computer Graphic Forum (Proc. of Eurographics) 32, 2 (2013), 63 – 72. https://doi.org/10.1111/cgf.12026

[4] Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder. ACM Transactions on Graphics (Proc. SIGGRAPH) 36, 4 (2017), 98:1–98:12. https://doi.org/10.1145 3072959.3073601