EUROGRAPHICS 2018 / D. Gutierrez and A. Sheffer
(Guest Editors)

Volume 37 (2018), Number 2

Approximate Program Smoothing Using Mean-Variance Statistics, with Application to Procedural Shader Bandlimiting

Y. Yang[1] and C. Barnes[1,2]

[1]University of Virginia, USA [2]Adobe Research

Abstract

This is a supplemental document to the paper "Approximate Program Smoothing Using Mean-Variance Statistics,with Application to Procedural Shader Bandlimiting." We present in this supplemental document additional plots of time and error for shaders, formulas, derivations, and proofs. Note that the main paper and supplemental use a single consistent set of references.

8. Time and Error Plots for Planar Geometry

Please see Figure 7 on the next page, where we present time and error plots for all 21 shaders that were defined in the main paper, applied to planar geometry.

9. Table of Smoothed Formulas

In Table 3, we show a table of functions and their corresponding convolutions with box and Gaussian kernels. These are needed for the approximations we developed in the main paper. This table can be viewed as an extension of the table presented in Dorn et al. [DBLW15], with some errors fixed. Note that in particular, for each function $f(x)$, we report not only the result of smoothing $f(x)$ but also smoothing $f^2(x)$ (e.g. if we report $\cos(x)$ then we also report $\cos^2(x)$). This is needed to determine the standard deviations output by a given compute stage for the adaptive Gaussian approximation rule.

In Table 3, we give that bandlimiting x^n by a Gaussian is a generalized Hermite polynomial $He_n^{[\alpha]}(x)$. This can be derived from a property of generalized Hermite polynomials: the nth noncentral moment of a Gaussian distribution X with expected value μ and variance σ is a generalized Hermite polynomial [Wik17]:

$$He_n^{[\alpha]}(x) = \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{n!}{(n-2k)!k!}(-2)^{-k}x^{n-2k}\alpha^k \tag{12}$$

10. Multivariate Smoothed Functions

In this section, we extend the analysis for the adaptive Gaussian rule from the main paper Section 4.2 to some additional multivariate functions. We explain how to derive formulas for modulo, comparisons, and a ternary selection function.

The modulo function, $f_{\text{mod}}(a,b) = a\%b$, can be rewritten as $f_{\text{mod}}(a,b) = b \cdot \text{fract}(\frac{a}{b})$. Here, fract$(x)$ is the fractional part of x. We make the simplifying assumption that the second argument b of mod is an ordinary (non-random) variable (so $\sigma_B = 0$), to obtain:

$$\begin{aligned}\mu_{\text{mod}}^2 &= \mu_B \cdot \widehat{\text{fract}}(\frac{\mu_A}{\mu_B}, \frac{\sigma_A^2}{\mu_B^2}) \\ \sigma_{\text{mod}} &= \mu_B^2 \cdot \widehat{\text{fract}^2}(\frac{\mu_A}{\mu_B}, \frac{\sigma_A^2}{\mu_B^2}) - \mu_{mod}^2\end{aligned} \tag{13}$$

Comparison functions $(>, \geq, <, \leq)$ are approximated by converting them to univariate functions including the Heaviside step function $H(x)$. As an example, the function greater than $(>)$ can be rewritten as $f_>(a,b) = H(a-b)$.

One other important multi-variate function we approximated is the ternary select(a,b,c) function, which returns b if a is nonzero, otherwise c. We approximated this in the same manner as Dorn et al. [DBLW15] as a linear interpolation based on binary operators: select$(a,b,c) = a \cdot b + (1-a) \cdot c$.

11. Correlation Coefficients for Multivariate Functions

In this section, we describe rules to compute the correlation coefficient ρ, which is briefly discussed in Section 4.2. Specifically, we are given a binary function $f(a,b)$, which receives two inputs a and b, with associated random variables A and B, respectively. We discuss the following two rules: a) assume ρ is constant and estimate by sampling and b) compute ρ under the assumption that computations are affine.

Estimate ρ by sampling. In a training stage, we use n samples to approximate ρ of two random variables A and B. The samples drawn from these two distributions are represented as a_i and b_i with corresponding sample mean $\bar{a}$ and $\bar{b}$. Thus, ρ can be estimated by:

$$\rho = \frac{\sum_{i=1}^n (a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^n (a_i - \bar{a})^2}\sqrt{\sum_{i=1}^n (b_i - \bar{b})^2}} \tag{14}$$

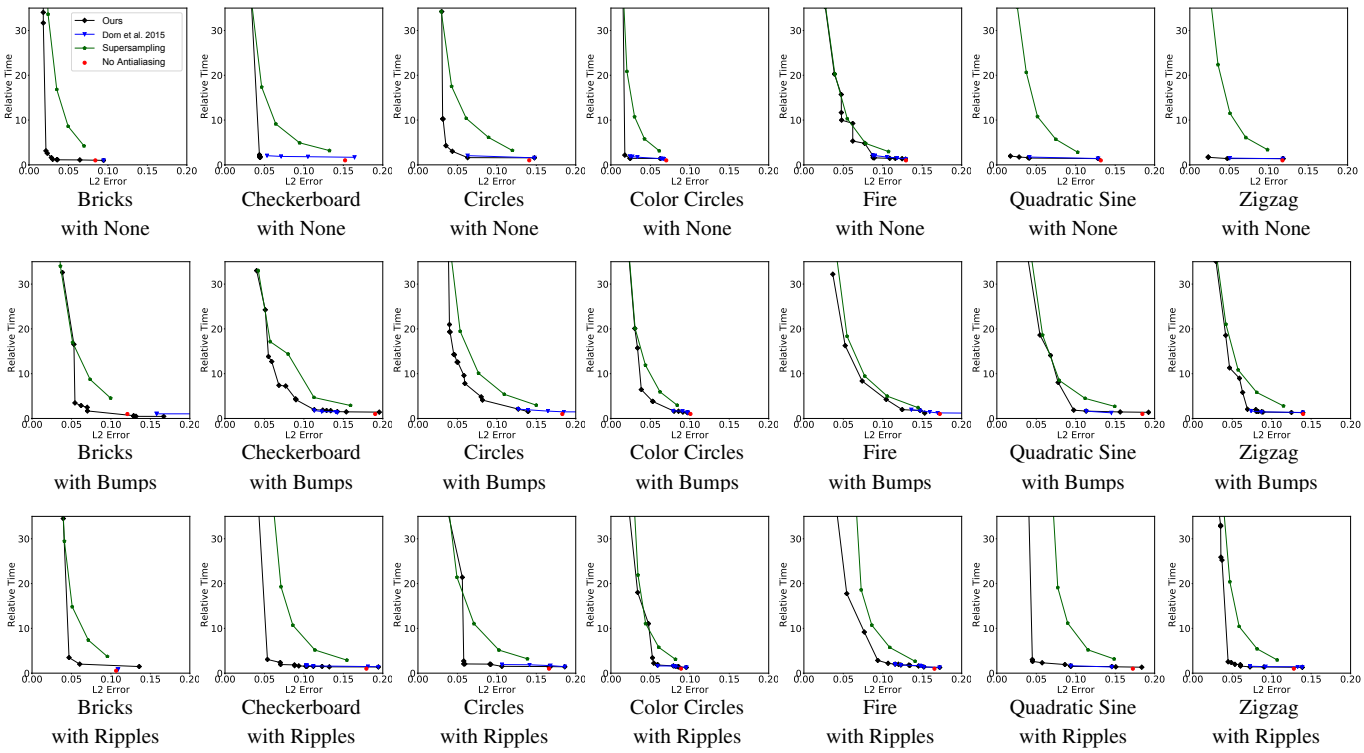

Figure 7: *Time versus error plots for the 21 shaders defined in the body of the paper, and applied to planar geometry. Here we show the Pareto frontier of program variants that optimally trade off running time and L^2 error. We show results for our method, Dorn et al [DBLW15], supersampling with varying numbers of samples, and the input shader without antialiasing. Note that our approach typically has significantly less error than Dorn et al [DBLW15] and is frequently an order of magnitude faster than supersampling for comparable error.*

Estimate ρ by an affine assumption. When we calculate ρ under this rule, we assume the variables a and b input to f are affine transformations of the variables x_1, ..., x_n which are input to the program. Under this assumption, a and b can be expressed as:

$$a = a_c + \sum_{i=1}^{n} a_i x_i, \qquad b = b_c + \sum_{i=1}^{n} b_i x_i \tag{15}$$

In Equation (15), a_i and b_i are coefficients of the affine transformation, and a_c and b_c end up not mattering for the ρ computation, so we ignore these constants. In our implementation, we find a_i and b_i by taking the gradient, via the automatic differentiation of the expression nodes a and b with respect to the inputs x_i. Here ρ is computed as:

$$\rho = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \tag{16}$$

12. Smoothing Result for Periodic Functions

In this section, we derive a convenient formula that gives the bandlimited result for any periodic function if its integral within a single period is known. We extend the analysis of fract() made by Dorn et al. [DBLW15] to any periodic function. We use Heckbert's technique of repeated integration [Hec86] to derive the convolution of a periodic function with a box kernel.

Specifically, we assume the periodic function $f(x)$ has period T and its first and second integrals within one period are also known. These are denoted as $F_p(x)$ and $F_{p2}(x)$, respectively.

$$\begin{aligned} F_p(x) &= \int_0^x f(u)du \\ F_{p2}(x) &= \int_0^x F_p(u)du \\ x &\in [0,T) \end{aligned} \tag{17}$$

Using Equation (17), we derive the first and second integral of $f(x)$ as follows.

$$\begin{aligned} F(x) &= \int_0^x f(u)du \\ &= \left(\left\lfloor \frac{x}{T} \right\rfloor + 1\right) \cdot F_p(T) - \int_{x-T\cdot\left\lfloor \frac{x}{T} \right\rfloor}^{T} f(u)du \\ &= \left(\left\lfloor \frac{x}{T} \right\rfloor + 1\right) \cdot F_p(T) - F_p(T) + F_p\left(x - T \cdot \left\lfloor \frac{x}{T} \right\rfloor\right) \\ &= \left\lfloor \frac{x}{T} \right\rfloor \cdot F_p(T) + F_p\left(x - T \cdot \left\lfloor \frac{x}{T} \right\rfloor\right) \end{aligned} \tag{18}$$

$$
\begin{aligned}
F_2(x) &= \int_0^x F(u)du \\
&= \int_0^x \left\lfloor \frac{u}{T} \right\rfloor \cdot F_p(T)du + \int_0^x F_p(u - T \cdot \left\lfloor \frac{u}{T} \right\rfloor)du \\
&= F_p(T) \cdot T \sum_{i=0}^{\left\lfloor \frac{x}{T} \right\rfloor - 1} i + \left(x - T \left\lfloor \frac{x}{T} \right\rfloor\right) \cdot \left\lfloor \frac{x}{T} \right\rfloor \cdot F_p(T) + \\
&\quad \left\lfloor \frac{x}{T} \right\rfloor \cdot F_{p2}(T) + F_{p2}\left(x - T \cdot \left\lfloor \frac{x}{T} \right\rfloor\right) \\
&= F_p(T) \cdot \left(\frac{T \cdot (q-1) \cdot q}{2} + (x - T \cdot q) \cdot q \right) + \\
&\quad F_{p2}(T) \cdot q + F_{p2}(x - T \cdot q) \\
&\text{Here, } q = \left\lfloor \frac{x}{T} \right\rfloor .
\end{aligned} \tag{19}
$$

Using Heckbert's result, the convolution of the periodic function $f(x)$ with a box kernel that has support $[-\sqrt{3}\sigma, \sqrt{3}\sigma]$ (corresponding to a uniform kernel with standard deviation σ) is:

$$
\hat{f}(x,\sigma) = \frac{F(x+\sqrt{3}\sigma) - F(x+\sqrt{3}\sigma)}{2\sqrt{3}\sigma} \tag{20}
$$

The convolution of the periodic function $f(x)$ with a tent kernel that has support $[-\sqrt{6}\sigma, \sqrt{6}\sigma]$ (corresponding to a uniform kernel with standard deviation σ) is:

$$
\hat{f}(x,\sigma) = \frac{F_2(x+\sqrt{6}\sigma) - 2 \cdot F_2(x) + F_2(x-\sqrt{6}\sigma)}{6\sigma^2} \tag{21}
$$

13. Proof of Second Order Approximation for a Single Composition

Here we show for a univariate function, applying function composition using our adaptive Gaussian approximation from Section 4.2 is accurate up to the second order in standard deviation σ. Suppose we wish to approximate the composition of two functions: $f(x) = f_2(f_1(x))$, where $f_1, f_2 : \mathbb{R} \to \mathbb{R}$. Assume the input random variable is $X_0 \sim \mathcal{N}(x, \sigma^2)$: the Gaussian kernel centered at x. The output from f_1 is an intermediate value in the computation: we can represent this with another random variable $X_1 = f_1(X_0)$. Similarly, the output random variable $X_2 = f_2(X_1)$. We conclude that $f(X_0) = f_2(f_1(X_0)) = f_2(X_1) = X_2$.

We apply Equation (6) and Equation (7) to f_1, and obtain the following mean and standard deviation.

$$
\begin{aligned}
\mu_{X_1} &= \hat{f}_1(x,\sigma^2) = f_1(x) + \frac{1}{2}\sigma^2 f_1^{''}(x) + \mathcal{O}(\sigma^4) \\
\widehat{f_1^2}(x,\sigma^2) &= f_1^2(x) + \frac{1}{2}\sigma^2 \frac{\partial^2}{\partial x^2}(f_1^2(x)) + \mathcal{O}(\sigma^4) \\
&= f_1^2(x) + \frac{1}{2}\sigma^2 (2 f_1 f_1^{''} + 2(f_1^{'})^2)(x) + \mathcal{O}(\sigma^4) \\
\sigma_{X_1}^2 &= \widehat{f_1^2}(x,\sigma^2) - \hat{f}_1(x,\sigma^2)^2 \\
&= \sigma^2 (f_1^{'})^2(x) + \mathcal{O}(\sigma^4)
\end{aligned} \tag{22}
$$

Using our composition rule, X_1 is approximated as a normal distribution using the mean and standard deviation calculated from Equation (22). That is, we approximate X_1 as being distributed as $\mathcal{N}(\mu_{X_1}, \sigma_{X_1}^2)$. Similarly, μ_{X_2}, which is the output we care about, can be computed based on Equation (7), Equation (22), and repeated Taylor expansion in σ around $\sigma = 0$.

$$
\begin{aligned}
\mu_{X_2} &= \hat{f}_2(\hat{f}_1(x,\sigma^2), \sigma_{X_1}^2) \\
&= f_2(f_1(x) + \frac{1}{2}\sigma^2 f_1^{''}(x) + \mathcal{O}(\sigma^4)) + \\
&\quad \frac{1}{2}\sigma_{X_1}^2 f_2^{''}(\hat{f}_1(x,\sigma^2)) + \mathcal{O}(\sigma_{X_1}^4) \\
&= f(x) + \frac{1}{2}\sigma^2 f_2^{'}(f_1(x)) f_1^{''}(x) + \\
&\quad \frac{1}{2}\sigma^2 f_2^{''}(f_1(x))(f_1^{'})^2(x) + \mathcal{O}(\sigma^4) \\
&= f(x) + \frac{1}{2}\sigma^2 f^{''}(x) + \mathcal{O}(\sigma^4)
\end{aligned} \tag{23}
$$

Comparing Equation (23) with Equation (7), the function composition in our framework agrees up to the second order term in the Taylor expansion.

We conclude that our approximation is accurate up to the second order in standard deviation for a single composition of univariate functions. The same property for additional compositions of univariate functions can be shown by induction.

14. Short Tuning

In Figure 8, we show 5 results for short tuning where the tuner has run for limited time. We log tuner output at the end of each generation, and use the first available results after the tuner has run for 10 minutes. We compare the results of short tuning with full tuning: the tuner is run by default for 20 generations. We described how we choose the result for our full tuning in Section 7.1. Similarly, we selected a result for our short tuning with sufficiently low error.

15. Geometry Transfer

In Figure 9, we show 6 results for geometry transfer: we tune shaders on one geometry, and use the Pareto frontier of tuned program variants to render the same shader on a different geometry. We compare the results of geometry transfer with directly training the same geometry, as is described in Section 7. We described how we choose the result of our method, Dorn et al. [DBLW15] and supersampling in Section 7.1. We then selected a result for geometry transfer that has time and error that is comparable to the directly trained result we previously selected.

References

[DBLW15] DORN J., BARNES C., LAWRENCE J., WEIMER W.: Towards automatic band-limited procedural shaders. In *Computer Graphics Forum* (2015), vol. 34, Wiley. 1, 2, 3, 6

[Hec86] HECKBERT P. S.: Filtering by repeated integration. In *ACM SIGGRAPH Computer Graphics* (1986), vol. 20, ACM, pp. 315–321. 2

[Wik17] WIKIPEDIA: Hermite polynomials — wikipedia, the free encyclopedia, 2017. [Online; accessed 20-May-2017]. URL: https://en.wikipedia.org/w/index.php?title=Hermite_polynomials&oldid=778044979. 1

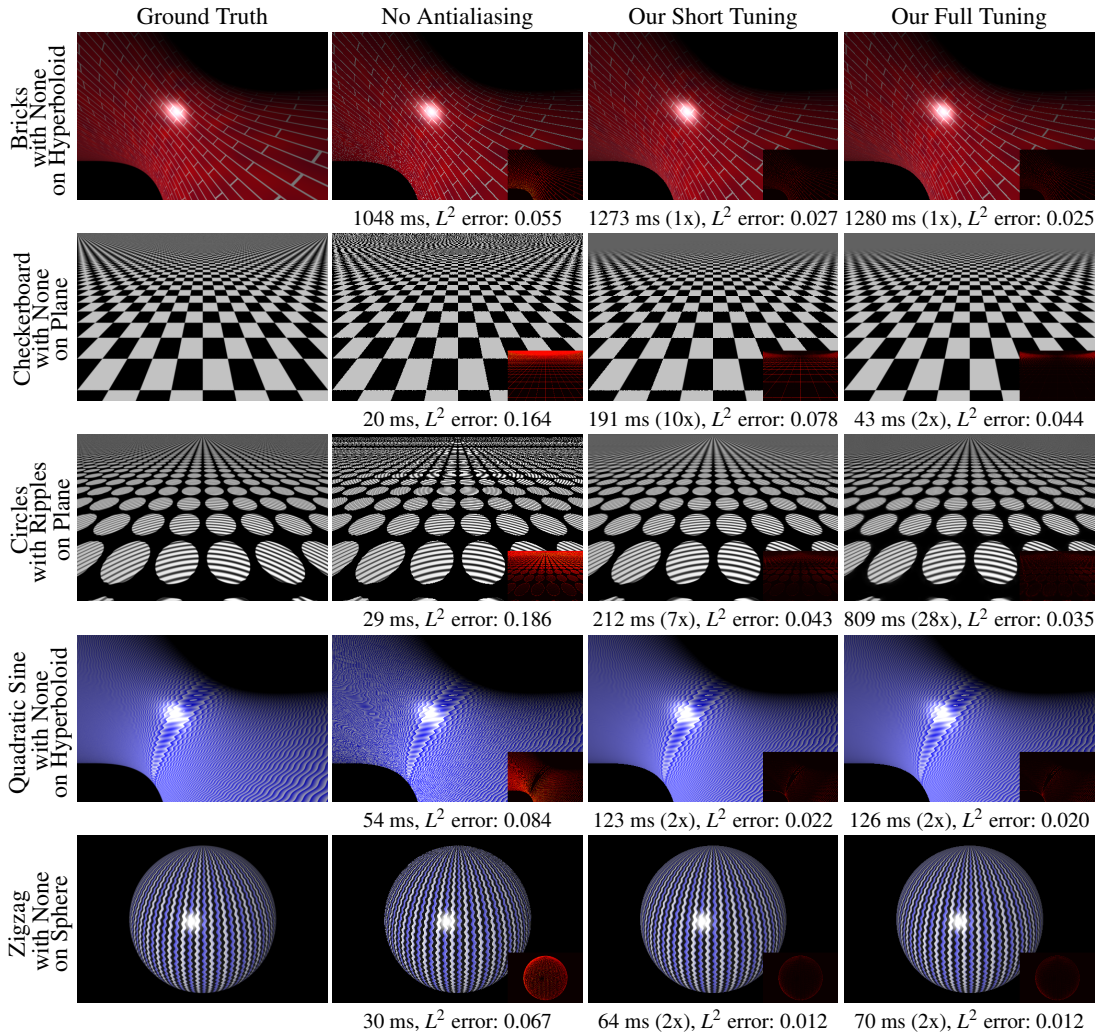

Figure 8: *Results for short tuning. Shaders are tuned for about 10 minutes ("Our Short Tuning") and compared to tuning for 20 generations ("Our Full Tuning"). Note that many aliasing patterns can be reduced after short tuning.*

Table 3: *A table of univariate functions, and their corresponding bandlimited result, using a box kernel B and a Gaussian G. The box kernel is the PDF of the uniform random variable $U[-\sqrt{3}\sigma,\sqrt{3}\sigma]$. The Gaussian kernel is the PDF of the random variable $\mathcal{N}(0,\sigma^2)$. Each random variable has standard deviation σ. We define $\mathrm{sinc}(x)=\sin(x)/x$, and the Heaviside step function $H(x)$ is 0 for $x\leq 0$ and 1 for x positive. Note that functions with undefined regions, such as x^p for negative or fractional p have σ limited as described in Section 4.4.*

Function $f(x)$	Bandlimited with box kernel: $\hat{f}^B(x,\sigma^2)$	Bandlimited with Gaussian kernel: $\hat{f}^G(x,\sigma^2)$
$x^p, p\neq -1$	$\frac{1}{\sqrt{12}\sigma(p+1)}\left[(x+\sqrt{3}\sigma)^{p+1}-(x-\sqrt{3}\sigma)^{p+1}\right]$	$He_p^{[-\sigma^2]}(x)$
x^{-2}	$(x^2-3\sigma^2)^{-1}$	
x^{-1}	$\frac{1}{\sqrt{12}\sigma}\log\left\lvert\frac{x+\sqrt{3}\sigma}{x-\sqrt{3}\sigma}\right\rvert$	
x	x	x
x^2	$x^2+\sigma^2$	$x^2+\sigma^2$
x^3	$x^3+3x\sigma^2$	$x^3+3x\sigma^2$
x^4	$x^4+6x^2\sigma^2+\frac{9}{5}\sigma^4$	$x^4+6x^2\sigma^2+3\sigma^4$
x^5	$x^5+10x^3\sigma^2+9x\sigma^4$	$x^5+10x^3\sigma^2+15x\sigma^4$
x^6	$x^6+15x^4\sigma^2+27x^2\sigma^4+\frac{27}{7}\sigma^6$	$x^6+15x^4\sigma^2+45x^2\sigma^4+15\sigma^6$
x^7	$x^7+21x^5\sigma^2+63x^3\sigma^4+27x\sigma^6$	$x^7+21x^5\sigma^2+105x^3\sigma^4+105x\sigma^6$
x^8	$x^8+28x^6\sigma^2+126x^4\sigma^4+108x^2\sigma^6+9\sigma^8$	$x^8+28x^6\sigma^2+210x^4\sigma^4+420x^2\sigma^6+105\sigma^8$
$\sin(x)$	$\sin(x)\,\mathrm{sinc}(\sqrt{3}\sigma)$	$\sin(x)e^{-\frac{\sigma^2}{2}}$
$\cos(x)$	$\cos(x)\,\mathrm{sinc}(\sqrt{3}\sigma)$	$\cos(x)e^{-\frac{\sigma^2}{2}}$
$\tan(x)$	$\frac{-1}{\sqrt{12}\sigma}\log\left\lvert\frac{\cos(x+\sqrt{3}\sigma)}{\cos(x-\sqrt{3}\sigma)}\right\rvert$	
$\sinh(x)$	$\frac{1}{\sqrt{12}\sigma}(\cosh(x+\sqrt{3}\sigma)-\cosh(x-\sqrt{3}\sigma))$	$\frac{1}{2}(e^{x+\frac{1}{2}\sigma^2}-e^{-x+\frac{1}{2}\sigma^2})$
$\cosh(x)$	$\frac{1}{\sqrt{12}\sigma}(\sinh(x+\sqrt{3}\sigma)-\sinh(x-\sqrt{3}\sigma))$	$\frac{1}{2}(e^{x+\frac{1}{2}\sigma^2}+e^{-x+\frac{1}{2}\sigma^2})$
$\tanh(x)$	$\frac{1}{\sqrt{12}\sigma}(\log(\cosh(x+\sqrt{3}\sigma))-\log(\cosh(x-\sqrt{3}\sigma)))$	
$\sinh^2(x)$	$\frac{1}{8\sqrt{3}\sigma}(-4\sqrt{3}\sigma+\sinh(2\sqrt{3}\sigma-2x)+\sinh(2\sqrt{3}\sigma+2x))$	
$\cosh^2(x)$	$\frac{1}{8\sqrt{3}\sigma}(4\sqrt{3}\sigma+\sinh(2\sqrt{3}\sigma-2x)+\sinh(2\sqrt{3}\sigma+2x))$	
$\tanh^2(x)$	$\frac{1}{2\sqrt{3}\sigma}(2\sqrt{3}\sigma-\tanh(\sqrt{3}\sigma-x)-\tanh(\sqrt{3}\sigma+x))$	
e^x	$\frac{1}{\sqrt{12}\sigma}\left(e^{x+\sqrt{3}\sigma}-e^{x-\sqrt{3}\sigma}\right)$	$e^{x+\frac{1}{2}\sigma^2}$
$\sin^2(x)$	$\frac{1}{2}-\frac{1}{2}\cos(2x)\,\mathrm{sinc}(\sqrt{12}\sigma)$	$\frac{1}{2}-\frac{1}{2}\cos(2x)e^{-2\sigma^2}$
$\cos^2(x)$	$\frac{1}{2}+\frac{1}{2}\cos(2x)\,\mathrm{sinc}(\sqrt{12}\sigma)$	$\frac{1}{2}+\frac{1}{2}\cos(2x)e^{-2\sigma^2}$
$\tan^2(x)$	$\frac{1}{\sqrt{12}\sigma}\left(\tan(x+\sqrt{3}\sigma)-\tan(x-\sqrt{3}\sigma)\right)-1$	
$H(x)$	$\begin{cases}0 & x\leq-\sqrt{3}\sigma\\ \frac{x}{2\sqrt{3}\sigma}+\frac{1}{2} & -\sqrt{3}\sigma\leq x\leq\sqrt{3}\sigma\\ 1 & x\geq\sqrt{3}\sigma\end{cases}$	$\frac{1}{2}(1+\mathrm{erf}\,\frac{x}{\sqrt{2}\sigma})$
$\mathrm{fract}(x)$	$\frac{1}{\sqrt{48}\sigma}(\mathrm{fract}^2(x+\sqrt{3}\sigma)+\lfloor x+\sqrt{3}\sigma\rfloor-$ $\mathrm{fract}^2(x-\sqrt{3}\sigma)-\lfloor x-\sqrt{3}\sigma\rfloor)$	
$\mathrm{fract}^2(x)$	$\frac{1}{\sqrt{108}\sigma}(\mathrm{fract}^3(x+\sqrt{3}\sigma)+\lfloor x+\sqrt{3}\sigma\rfloor-$ $\mathrm{fract}^3(x-\sqrt{3}\sigma)-\lfloor x-\sqrt{3}\sigma\rfloor)$	
$\lfloor x\rfloor$	$x-\widehat{\mathrm{fract}}(x)$	
$\lfloor x\rfloor^2$	$\widehat{x^2}+\widehat{\mathrm{fract}^2}(x)-F(x+\sqrt{3}\sigma)+F(x-\sqrt{3}\sigma)$ where $F(x)=2(\frac{\lfloor x\rfloor}{3}+\frac{\lfloor x\rfloor(\lfloor x\rfloor-1)}{4}+\frac{\lfloor x\rfloor\widehat{\mathrm{fract}^2}(x)}{2}+\frac{\widehat{\mathrm{fract}^3}(x)}{3})$	
$\lceil x\rceil$	$x+\widehat{\mathrm{fract}}(-x)$	
$\lceil x\rceil^2$	$\widehat{\lfloor -x\rfloor^2}$	

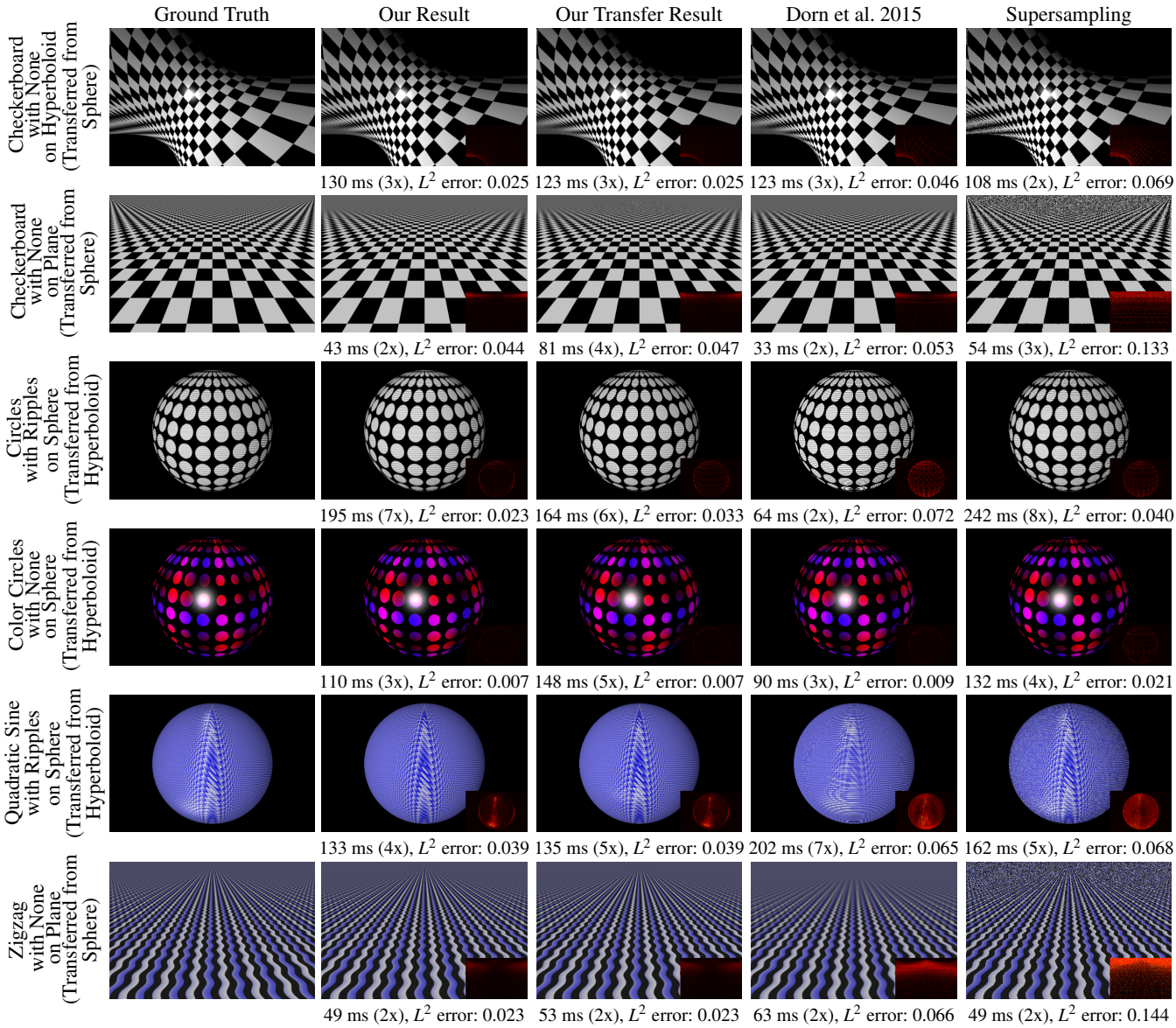

Figure 9: *Results for geometry transfer. Shaders trained on a source geometry are applied to a target geometry ("Our Transfer Result") and compared to re-training on the target geometry. Note that both transfer between curved geometries and from curved geometry to plane give good antialiasing results and have significant less error than Dorn et al. [DBLW15]. Note also that transfer results between curved geometries are typically competitive to direct training both in time and error.*