Frequency Analysis and Sheared Reconstruction for Rendering Motion Blur

Kevin Egan [*] Columbia University
Yu-Ting Tseng Columbia University
Nicolas Holzschuch INRIA — LJK
Frédo Durand MIT CSAIL
Ravi Ramamoorthi UC Berkeley

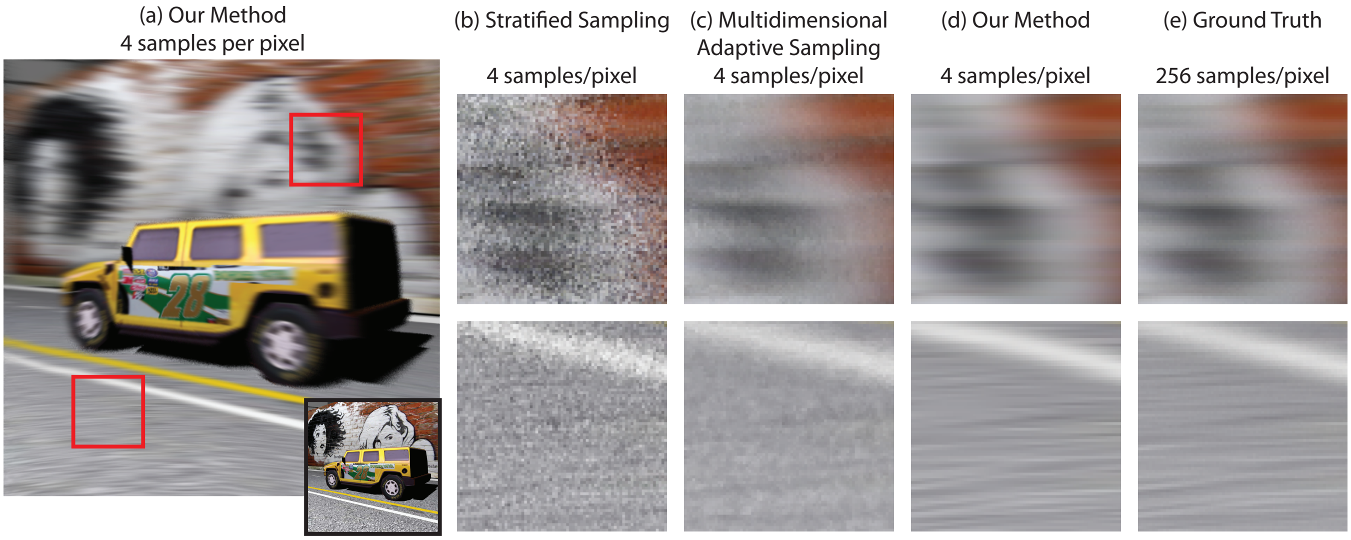


Figure 1: (a) Our method using an average of only 4 samples per pixel over the image. A static rendering of the scene is inset in the lower right and closeups are shown in (b-e). Stratified sampling in (b) is very noisy at this low sample count. Multidimensional Adaptive Sampling [Hachisuka et al. 2008] in (c) performs much better, but still has some noise, especially in fast-moving high-frequency textures, such as the mural (top) and ground (bottom closeup). Our technique in (d) produces a high-quality image with minimal noise that closely matches ground truth (e). Figure 7 shows details for our sheared filter.

Abstract

Motion blur is crucial for high-quality rendering, but is also very expensive. Our first contribution is a frequency analysis of motion-blurred scenes, including moving objects, specular reflections, and shadows. We show that motion induces a shear in the frequency domain, and that the spectrum of moving scenes can be approximated by a wedge. This allows us to compute adaptive space-time sampling rates, to accelerate rendering. For uniform velocities and standard axis-aligned reconstruction, we show that the product of spatial and temporal bandlimits or sampling rates is constant, independent of velocity. Our second contribution is a novel sheared reconstruction filter that is aligned to the first-order direction of motion and enables even lower sampling rates. We present a rendering algorithm that computes a sheared reconstruction filter per pixel, without any intermediate Fourier representation. This often permits synthesis of motion-blurred images with far fewer rendering samples than standard techniques require.

Keywords: motion blur, frequency analysis, space-time, light transport, anti-aliasing, reconstruction, filter, sampling

1 Introduction

Motion blur is important for creating synthetic images that match physical cameras, and for eliminating temporal aliasing in animations. As the velocity increases, more samples are usually required to render motion-blurred images. This is frustrating since the complexity and spatial frequencies in the final image actually *decrease* due to the blurring or filtering from motion (see Figures 1 and 2).

We seek to accelerate the rendering of motion-blurred scenes by a combination of adaptive sampling and a new sheared filter. Our main contribution is an analysis of the frequency content of scenes in space-time. This theoretical analysis enables us to derive the bandwidth, required sampling rate, and reconstruction filters for accurate rendering. We make the following contributions:

Space-Time Fourier Theory: We develop our frequency analysis in Sec. 3 with three key visual effects: movement of objects and surface texture, rotations of the BRDF and lighting, and moving shadows. We find similar mathematical forms in all cases: the final motion-blurred signal undergoes a *shear* in space-time and a corresponding shear in the frequency domain. For a given range of velocities, the Fourier spectrum can be approximated by a wedge.

Spatial and Temporal Bandlimits and Sampling Rates: This analysis allows us to derive required spatial and temporal sampling rates (Sec. 4), enabling adaptive sampling. In fact, we show that, using a conventional (axis-aligned) aliasing and shutter filter, and for uniform velocities, the product of spatial and temporal sampling rates is essentially constant, independent of the speed of motion.

Sheared Reconstruction Filter: We further demonstrate that we can sample more sparsely and pack frequency replicas much tighter if we use a new sheared (not axis-aligned) reconstruction filter, which conforms to the frequency wedge (Sec. 5), and follows the first-order direction of motion in the primal domain.

[*] e-mail:ktegan@cs.columbia.edu

Practical Space-Time Rendering Algorithm: Our motion-blur rendering method (Sec. 6) first estimates frequency bounds by sparsely sampling the scene. The algorithm then computes per-pixel sheared filters and sampling rates, without requiring any explicit computation of Fourier spectra. As shown in Figure 1, it can produce high-quality results with low sample counts.

2 Related Work

Motion Blur Rendering: Motion-blur rendering often relies on sampling the shutter interval, e.g. [Korein and Badler 1983; Cook et al. 1984; Haeberli and Akeley 1990; Cammarano and Jensen 2002; Akenine-Möller et al. 2007] and high-quality sampling patterns can improve results [Mitchell 1991]. The Reyes architecture [Cook et al. 1987] reduces costs by shading at one time instant but densely sampling visibility through time. The Maya rendering system computes shading and visibility separately to capture changing illumination and reduce noise [Sung et al. 2002].

Closest to our work is the general Multi-Dimensional Adaptive Sampling (MDAS) method [Hachisuka et al. 2008], which adaptively samples based on contrast in the multi-dimensional integrand. They approximate anisotropic filters with finite differences and a modified nearest-neighbor. In contrast, we predict local frequency information with each sample and utilize sheared reconstruction filters. A comparison of the practical results is made in Sec. 6.4; our method is somewhat better on fast-moving high-frequency signals, as in Figure 1. Furthermore, our paper makes important theoretical contributions by analyzing motion blur in the frequency domain, which leads to key insights for sampling rates and anisotropic filters that may be relevant to MDAS as well.

Multi-dimensional lightcuts [Walter et al. 2006] groups point light sources and shading samples, including samples in time for motion blur, into hierarchical graphs. This method is orthogonal to ours, since they reuse similar surface and lighting samples within one pixel, while we consider sheared reconstruction filters that can span multiple pixels.

Image-space solutions blur based on the motion field at a single instant [Potmesil and Chakravarty 1983; Max and Lerner 1985]. They can be efficient but often require segmentation into layers, provide only an approximation, and are prone to artifacts. Our sheared reconstruction is related but operates on the full space-time domain and adapts to the content to yield accurate results.

Other methods have used modified filters for motion blur. Catmull [1984] suggests scaling the pixel anti-aliasing filter to match the motion, but it relies on analytic filtering of polygons. Anisotropic texture filtering has also been used in real-time rendering [Loviscach 2005]. Both of these methods define a stretched space-only filter instead of our sheared space-time filter.

Light Transport Analysis: Our analysis builds on plenoptic sampling [Chai et al. 2000; Isaksen et al. 2000], and the frequency and gradient analysis of light transport [Durand et al. 2005; Soler et al. 2009; Ramamoorthi et al. 2007]. In particular, we use the concept of light transport shears in the frequency domain [Durand et al. 2005] and a wedge for the final spectrum [Chai et al. 2000]. We extend these space-angle methods to consider motion and space-time. Other work has touched on the sheared space-time spectra of translating signals, although not for rendering [Christmas 1998; Levin et al. 2008]. We go further in deriving explicit sampling rates, a theorem showing that the total sampling rate (in space and time) is approximately constant for axis-aligned filters, developing a sheared reconstruction filter, and in considering specularities and shadows.

$g(x,y)$	2D spatial signal (such as a planar texture)
$f(x,y,t)$	Time-Varying signal (moving object or texture)
$h(x,y,t)$	Time-Varying motion-blurred signal (image)
$f(x,t), h(x,t)$	1D time-varying signals for simplicity
$w(t)$	Temporal response of shutter
$\Omega_x^{\max}, \Omega_t^{\max}$	Max spatial, temporal frequencies (in $g(x)$,$w(t)$)
Ω_x^*, Ω_t^*	Spatial, temporal frequency bandlimit
Ω^*	Net frequency bandlimit (total samples needed)

Table 1: *Notation for the key variables in the paper. The frequency analysis will use capital letters for Fourier transforms of the quantities shown here e.g., $F(\Omega_x, \Omega_t)$ denotes the Fourier transform of $f(x,t)$. Other notation is introduced in Secs. 3.2-3.3 to discuss BRDF effects and shadows.*

3 Space-Time and Fourier Theory

We analyze the key visual effects in motion blur. We first examine the frequency content of a moving signal and show that it yields a space-time shear. General light transport involves shearing, convolution and other operations on spectra [Durand et al. 2005], and it is beyond the scope of this paper to generalize all of them to the time domain. Instead, we focus on the three most common phenomena—object motion, BRDF reflection, and moving shadows. We show that, in space-time, all three effects have strikingly similar mathematical forms, which allows for a general treatment of motion blur as a *shear* in the space-time and Fourier domain.

For simplicity, most of the analysis is done for a 1D scanline, but the main insights carry over to 2D images and 3D space-time (with anisotropic shears following the direction of motion). An index of notation for the most important symbols is in Table 1.

3.1 Moving Object: Translating Signal

Consider a 2D signal $g(x,y)$, which can be thought of as a texture. The concept of "texture" here is general, and can also include geometric effects like silhouette boundaries. This signal is translated through time by $x_0(t)$ and $y_0(t)$,

$$f(x,y,t) = g(x - x_0(t), y - y_0(t)). \tag{1}$$

The motion-blurred signal or image is then given by

$$h(x,y,t) = \int_{-\infty}^{\infty} f(x,y,t')w(t-t')\,dt', \tag{2}$$

where $w(t)$ is the shutter response over time, responsible for motion blur. For Fourier analysis, it is useful to define all integrals over the infinite temporal domain. We consider h to be a continuous signal for analysis—in practice, the final rendering step will point-sample h in time to generate individual motion-blurred frames.

We first study the canonical case of translation with uniform velocities, so that $x_0(t) = at$ and $y_0(t) = bt$,

$$f(x,y,t) = g(x - at, y - bt). \tag{3}$$

For simplicity, consider a 1D scanline as in Figure 2(a):

$$\begin{aligned} f(x,t) &= g(x - at) \\ h(x,t) &= \int f(x,t')w(t-t')dt'. \end{aligned} \tag{4}$$

The basic setup is as shown in the top row of Figure 2, with Figure 2(b) being the final motion-blurred image. Figure 2(c) is a space-time diagram for a static scene ($a = 0$). In this case, there is no variation along the time (vertical) dimension. In Figure 2(d), we see the time-varying effects of motion. As is expected from Equations 3 and 4, this is a *shear* along the spatial x direction. The effects of the shutter in Figure 2(e) are a blurring or filtering across

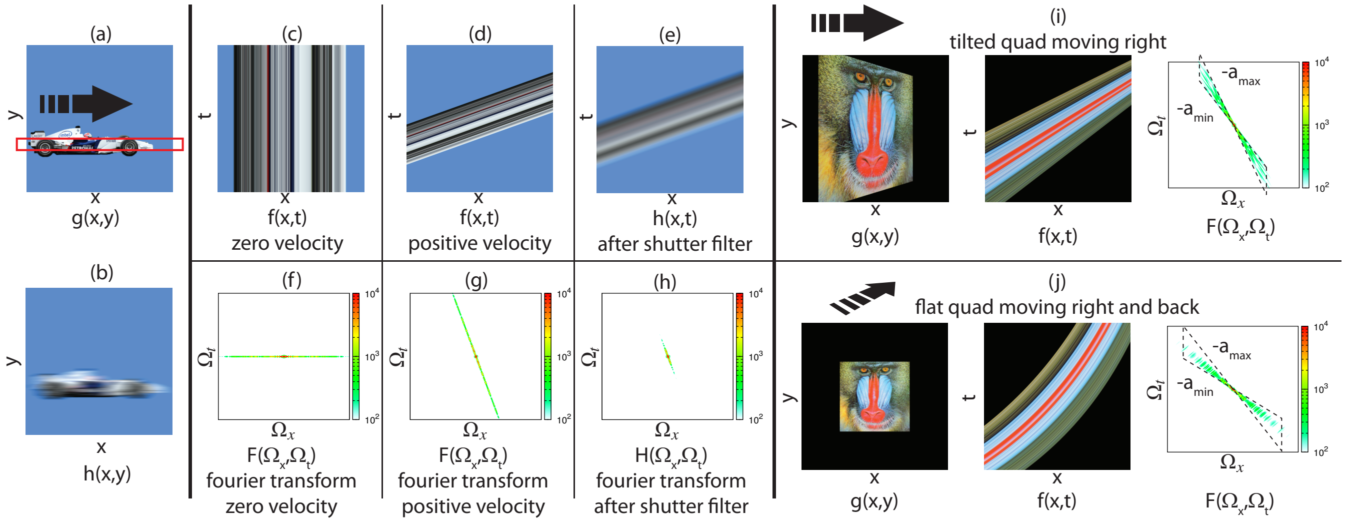


Figure 2: *Space-Time and Fourier domain plots for a moving object. (a) Original signal $g(x, y)$; the scanline used for graphs (c), (d), and (e) is outlined in red. (b) (below (a)) $h(x, y, t)$ for a single instant in time; this is our final motion-blurred image. (c) A graph of $f(x, t)$ with zero velocity (a static image). In this case, there is no variation along the time or vertical axis. (d) $f(x, t)$ with positive uniform velocity, leading to a shearing along the spatial dimension. (e) $h(x, y, t)$ is obtained by applying a vertical blur along the time axis corresponding to the shutter filter. (f), (g) and (h) are the respective Fourier transforms of (c), (d) and (e). Note that (h) has frequencies in time restricted to $\Omega_t \in [-\Omega_t^{\max}, \Omega_t^{\max}]$ based on the shutter filter. (i) Because of perspective, the velocities change across space. (j) Because of perspective, velocities change across time. The frequency spectra span a wedge based on the minimum and maximum velocities.*

the vertical time dimension. Figure 2(f-h), shows the corresponding frequency spectra, which we now derive analytically.

Fourier Analysis: To calculate the Fourier transform $\mathcal{F}(f(x, y, t))$, we first transform along x and y axes (denoted $\mathcal{F}_{x,y}$) to obtain an intermediate $F_t(\Omega_x, \Omega_y, t)$, and then transform along the time dimension. Therefore, we first calculate

$$F_t(\Omega_x, \Omega_y, t) = \mathcal{F}_{x,y}\left[g(x - x_0(t), y - y_0(t))\right]. \tag{5}$$

Since $x_0(t)$ and $y_0(t)$ depend only on time, they can be treated as constant shifts for the spatial Fourier transform above. By the standard theory of shifted Fourier transforms, F_t relates closely to $G(\Omega_x, \Omega_y)$ which is the Fourier transform of g,

$$F_t(\Omega_x, \Omega_y, t) = e^{-i2\pi(\Omega_x x_0(t)+\Omega_y y_0(t))}G(\Omega_x, \Omega_y). \tag{6}$$

Now consider translation with a uniform velocity a and b in the x and y directions, as per Equation 3. Applying the Fourier transform along the time axis

$$\begin{aligned} F(\Omega_x, \Omega_y, \Omega_t) &= G(\Omega_x, \Omega_y) \int e^{-i2\pi t\left(\Omega_x a+\Omega_y b+\Omega_t\right)}\, dt \\ &= G(\Omega_x, \Omega_y)\delta(\Omega_x a + \Omega_y b + \Omega_t). \end{aligned} \tag{7}$$

By translating the 2D signal (corresponding to a spatial shear in the space-time domain), we have sheared the signal along the temporal axis in the frequency domain (all non-zero frequencies lie on the plane $\Omega_x a + \Omega_y b + \Omega_t = 0$ in 3D Fourier space). This result also shows the coupling of spatial and temporal dimensions.

While our analysis applies fully to 2D signals, it is easier to expose with a single spatial dimension or a 1D signal per Equation 4,

$$\boxed{F(\Omega_x, \Omega_t) = G(\Omega_x)\delta(\Omega_x a + \Omega_t),} \tag{8}$$

restricting the frequency spectrum to a single line $\Omega_x a + \Omega_t = 0$, as seen in Figure 2(g). Note that Figure 2(g) is obtained by shearing the Fourier spectrum in Figure 2(f) along the time dimension, with the amount of shear given by the velocity a.

Finally, from Equation 4, we know that $h(x, t)$ is obtained from $f(x, t)$ simply by convolving with $w(t)$, which becomes a multiplication in the temporal frequency domain,

$$\boxed{H(\Omega_x, \Omega_t) = G(\Omega_x)\delta(\Omega_x a + \Omega_t)W(\Omega_t).} \tag{9}$$

As seen in Figure 2(h), the high temporal frequencies in Figure 2(g) are attenuated or removed, because W is the frequency spectrum of the low-pass shutter filter (in principle, only an infinite sinc function can be an exact low-pass filter, but most filters like gaussians allow one to define a practical threshold, such as capturing 99% of the energy).

Non-Uniform Velocities: For typical shutter speeds that cover a short time window, a uniform velocity is often a good approximation. However, there are cases where perspective, acceleration and occlusion effects cause variations in speed and spatially non-uniform velocities. An analytic Fourier transform cannot be obtained in these cases, but we can approximate its range, based on the non-negative minimum and maximum velocities $a \in [a_{\min}, a_{\max}]$.

Figure 2(i) shows a tilted quad moving to the right, where velocities change across space because of perspective. Analogously, Figure 2(j) shows a quad moving right and away from the camera, with velocities changing across time because of perspective. While the spectra are complicated, we find that most of the energy lies in the wedge bounded by shears corresponding to minimum $a_{\min}$ and maximum $a_{\max}$ velocities (Figures 2(i),2(j),6(a)). This is similar to the use of minimum and maximum depths to bound the frequency spectrum for image-based rendering [Chai et al. 2000].

3.2 BRDF Effects and Shading

We now consider the motion of reflections (and shadows in Sec. 3.3). We will obtain very similar mathematical forms as those just seen for moving objects. This is illustrated in Figure 3, which shows the shearing in spatial and frequency domains, analogous to Figure 2. Some readers may wish to skip the derivations on a first reading, and can move directly to Sec. 4 without loss of continuity.

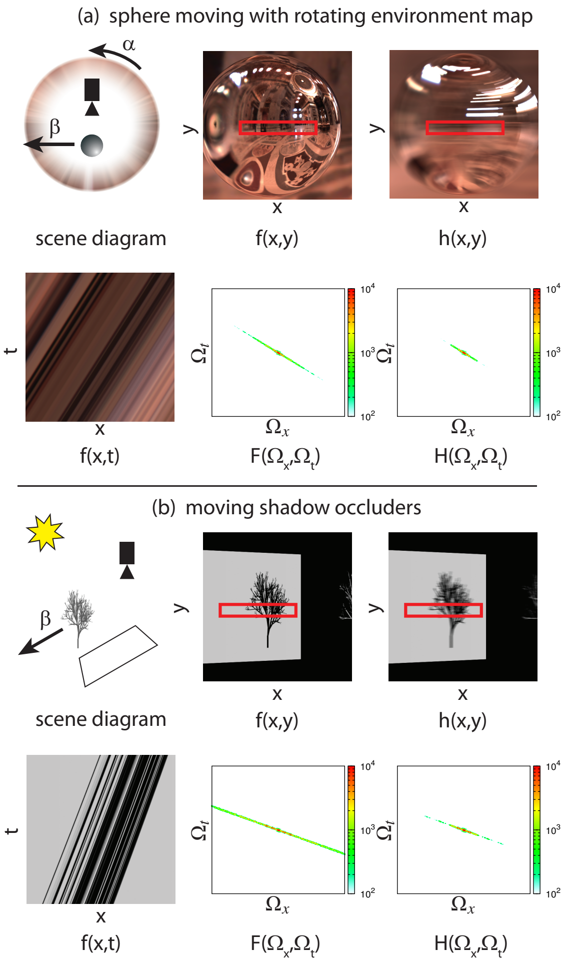


Figure 3: *(a) A moving surface, in this case a sphere, with a rotating environment map. As the object moves, the specular reflections are motion-blurred. (b) A moving shadow from blockers, in this case a tree. As the occluder moves, so does the occluded region, leading to motion-blurred shadows on the receiver. We obtain space-time and frequency-domain shears based on the effective pixel velocities. (Note that since our analysis is local, curved global paths for specular highlights and shadows are not an issue.)*

For simplicity, we consider flatland or 2D reflections, similar to [Durand et al. 2005; Ramamoorthi et al. 2007]. A diagram is shown in Figure 4. We write the standard reflection equation for $f(x,t)$, but extend it by considering its time-varying nature,

$$f(x,t) = \int l(\theta,t) r(2n(x,t) - \theta)\, d\theta, \qquad (10)$$

where $l(\theta,t)$ is the (time-varying) incident lighting[1] and r is a radially symmetric BRDF (like Lambertian or Phong), including the cosine term. As shown in Figure 4, we consider a single overhead view, so that the angle between lighting and reflected directions is given by $2n - \theta$ where n is the normal.

There are two sources of time-dependence or motion blur. First, the lighting may vary with time—for concreteness, we consider moving the lights. For distant illumination, this corresponds to a rotation, with α being the angular velocity. We can also linearize motions of local sources to a rotation and angular velocity,

$$l(\theta,t) = l(\theta - \theta_0(t)) = l(\theta - \alpha t). \qquad (11)$$

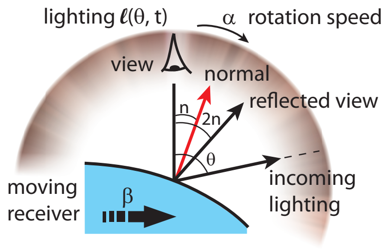


Figure 4: *BRDF effects and shading with motion blur. The basic (planar or flatland 2D) setup shows a complex lighting environment $l(\theta,t)$ that can rotate with angular velocity α. The surface can also move with speed β.*

Next, consider normal $n(x,t)$. If the object is translating,

$$n(x,t) = n(x - x_0(t)) = n(x - \beta t), \qquad (12)$$

where we now use β for the velocity of motion (to distinguish from a used previously). Finally, the normal can be locally linearized so that $n(x) = \kappa x + \eta$, with κ related to the surface curvature,[2]

$$n(x - \beta t) = \kappa(x - \beta t) + \eta = \kappa x - \kappa\beta t + \eta. \qquad (13)$$

Now, substituting Equations 11 and 13 into Equation 10 and using $\kappa' = 2\kappa$ and $\eta' = 2\eta$ to account for the factor of $2n(\cdot)$,

$$f(x,t) = \int l(\theta - \alpha t) r(\kappa' x - \beta\kappa' t - \theta + \eta')\, d\theta, \qquad (14)$$

The above equation can be integrated by substituting $\omega = \theta - \alpha t$,

$$f(x,t) = \int l(\omega) r\left([\kappa' x - (\alpha + \beta\kappa')t + \eta'] - \omega\right)\, d\omega. \qquad (15)$$

The right-hand side of the above equation is a convolution. Defining $\gamma = \alpha + \beta\kappa'$—where γ is the relative angular velocity of lighting and surface—and using $\otimes$ for convolution,

$$f(x,t) = (l \otimes r)(\kappa' x - \gamma t + \eta'), \qquad (16)$$

where the result is evaluated at $(\kappa' x - \gamma t + \eta')$.

It is possible to bring Equation 16 into the same form as Equation 4, unifying two seemingly quite different phenomena—motion-blurred texture/geometry and specular reflections. To do so, we simply need to define $g = l \otimes r$, so that in analogy to Equation 4,

$$\begin{aligned} f(x,t) &= g\left(\kappa'\left[x - \frac{\gamma}{\kappa'}t + \frac{\eta'}{\kappa'}\right]\right) \\ h(x,t) &= \int f(x,t') w(t - t') dt'. \end{aligned} \qquad (17)$$

In this case, the effective velocity a from Equation 4 is simply γ/κ', which is the effective spatial rate of motion (relative angular velocity divided by curvature). The η'/κ' term is only a constant offset, which will become a simple phase shift in Fourier space. The curvature κ' multiplies x to convert from spatial to angular coordinates.

Fourier Analysis: The convolution of lighting and BRDF in Equation 16 leads to a product in Fourier space,

$$F(\Omega_x, \Omega_t) = L\left(\frac{\Omega_x}{\kappa'}\right) R\left(\frac{\Omega_x}{\kappa'}\right) e^{i2\pi\Omega_x\eta'/\kappa'} \delta\left(\Omega_x \frac{\gamma}{\kappa'} + \Omega_t\right). \qquad (18)$$

The scale of κ' in the arguments of Equations 16 and 17 leads to the Fourier scale factors of $1/\kappa'$. Equation 18 is essentially identical to Equation 8 for moving objects, if we define effective velocity $a = \gamma/\kappa'$, and $G(\Omega_x) = (LR)(\Omega_x/\kappa')$. In both cases, the signal is a shear in both space-time and Fourier domains.

[1] The lighting can canonically be thought of as a distant environment map, but can also correspond to the local environment at $x = 0$ (assuming the spatial variation of lighting is moderate, such as mid-range illumination).

[2] Since the surface may be tilted with respect to the image scanline along which the spatial dimension x is measured, κ is actually the screen-space curvature, and differs by a cosine factor from the geometric curvature.

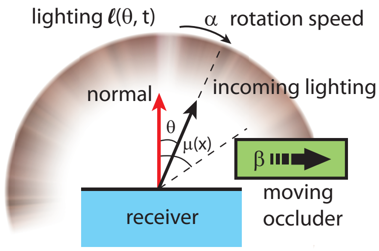


Figure 5: *Schematic for analysis of motion-blurred shadows. The lighting can move with angular velocity α. The occluder can also move with speed β, leading to a change in the extremal angle $\mu(x)$ for visibility.*

3.3 Visibility and Cast Shadows

We follow previous work [Soler and Sillion 1998; Ramamoorthi et al. 2004; Mahajan et al. 2007], which shows that canonical shadow effects are often described by convolutions.

We first define the binary visibility function $v(x, \theta)$ as

$$v(x, \theta) = s(\mu(x) - \theta), \tag{19}$$

where s is the Heaviside step function, and $\mu(x)$ is an extremal angle that defines the boundary between blocked and continuous regions, as shown in Figure 5. For simplicity, we consider only a single visibility discontinuity for each x, but a linear combination of functions can be used for general visibility [Ramamoorthi et al. 2007]. Consider relative motion β between the blocker and receiver,

$$s(\mu(x - x_0(t)) - \theta) = s(\mu(x - \beta t) - \theta). \tag{20}$$

We now locally linearize $\mu(x) \approx \nu x$ [Ramamoorthi et al. 2004]. In general, $|\nu| \sim \cos\mu / D$, where D is the distance to the blocker,

$$s(\mu(x - \beta t) - \theta) = s(\nu \cdot (x - \beta t) - \theta) = s(\nu x - \beta\nu t - \theta). \tag{21}$$

Finally, we define $l(\theta, t) = l(\theta - \alpha t)$ as in the BRDF case—effective values for angular velocity α can be computed for point and area lights, or environment maps. If we ignore the BRDF signal for the moment (cases with multiple surface, BRDF, and shadow signals are discussed later), we can write the reflection equation as

$$f(x, t) = \int l(\theta - \alpha t) s(\nu x - \beta\nu t - \theta)\, d\theta. \tag{22}$$

This has exactly the same form as Equation 14, only using s instead of the BRDF r, and ν instead of the curvature κ'. If we similarly define $\gamma = \alpha + \beta\nu$, we obtain analogous to Equation 16,

$$f(x, t) = (l \otimes s)(\nu x - \gamma t). \tag{23}$$

This can be put in the same form as the specularity and motion case (e.g., Equation 17), with effective velocity $a = \gamma / \nu$.

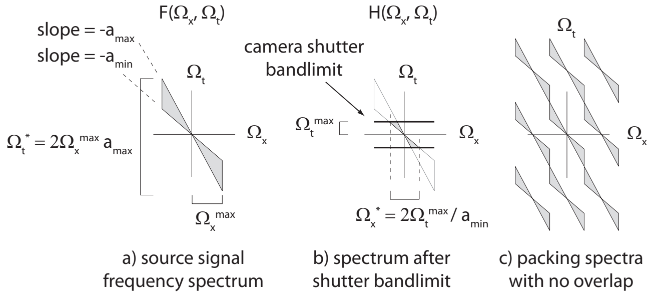
Fourier Analysis: The Fourier formula in the shadow case is very similar to that for BRDF effects in Equation 18,

$$F(\Omega_x, \Omega_t) = L\left(\frac{\Omega_x}{\nu}\right) S\left(\frac{\Omega_x}{\nu}\right) \delta\left(\Omega_x \frac{\gamma}{\nu} + \Omega_t\right), \tag{24}$$

which has an identical form to Equations 8 and 18 if we set the effective velocity $a = \gamma/\nu$ and $G(\Omega_x) = (LS)(\Omega_x/\nu)$. One can also similarly define the Fourier transform of the motion-blurred signal H for specularity and shadows, as per Equation 9.

4 Spatial and Temporal BandLimits

We now study the spatial and temporal bandlimits. Since the mathematical form is very similar for all the visual effects in Secs. 3.1-3.3 (provided we define an effective velocity a), from now on we focus on Equations 8 and 9. Figure 6 illustrates the main ideas.



a) source signal frequency spectrum

b) spectrum after shutter bandlimit

c) packing spectra with no overlap

Figure 6: *(a) Frequency spectrum of source signal $F(\Omega_x, \Omega_t)$ in space and time (Ω_x and Ω_t). We also mark the highest spatial frequency $\Omega_x^{\max}$, and the highest temporal frequency Ω_t^*, determined by the maximum velocity/shear $a_{\max}$. (b) The signal is bandlimited in time based on the camera shutter to temporal frequencies less than $\Omega_t^{\max}$. For images with medium to large amounts of motion blur, the spatial frequencies are also correspondingly filtered to Ω_x^*, depending on the minimum velocity $a_{\min}$. (c) Sampling introduces replicas of the base spectrum F. To achieve a low sampling rate we must bring the spectra as close as possible without aliasing.*

Time-Varying Signal $F(\Omega_x, \Omega_t)$**:** In general, the frequency spectrum is a wedge bounded by the minimum and maximum velocities/shears, as shown in Figure 6(a). From Equation 8, the spatial frequencies are bandlimited by $G(\Omega_x)$ so that $\Omega_x \in [-\Omega_x^{\max}, \Omega_x^{\max}]$, where $\Omega_x^{\max}$ is the highest spatial frequency in the signal g. Therefore, the temporal frequencies lie within $\Omega_t \in [-a_{\max}\Omega_x^{\max}, a_{\max}\Omega_x^{\max}]$, and the temporal frequency extent Ω_t^* is

$$\Omega_t^* = 2a_{\max}\Omega_x^{\max}. \tag{25}$$

According to the Nyquist theorem, we need to sample at this temporal rate to properly separate the Fourier domain replicas from sampling (Figure 6(c)). Otherwise, even after convolution with the low-pass camera shutter, the result would be inaccurate because of aliasing into low frequencies.[3]

Motion-Blurred Result $H(\Omega_x, \Omega_t)$**:** Finally, we convolve the time-varying signal with the camera shutter to obtain $h(x, t)$ and its associated Fourier transform per Equations 4 and 9. This leads to a low-pass filter along the vertical (time) axis as in Figure 6(b). Therefore, $\Omega_t \in [-\Omega_t^{\max}, \Omega_t^{\max}]$, where $\Omega_t^{\max}$ is the maximum frequency in the Fourier transform of the camera shutter $W(\Omega_t)$. Interestingly, the spatial frequencies are also bandlimited, since they must lie on the line $\Omega_x a + \Omega_t = 0$. Hence, it holds that:

$$\Omega_x^* = 2\frac{\Omega_t^{\max}}{a_{\min}} \tag{26}$$

Not surprisingly, the spatial frequency content is much lower due to motion blur.

The above result needs a small modification in the quasi-static case. If the velocity $a_{\min}$ is sufficiently small, the temporal frequencies Ω_t^* in Equation 25 is less than the filtering effect of the shutter response. Therefore, the motion blur filter has minimal impact on the signal (much as motion blur does not affect a static scene). In this case, we simply have $\Omega_x^* = 2\Omega_x^{\max}$. In general,

$$\Omega_x^* = 2\min\left(\frac{\Omega_t^{\max}}{a_{\min}}, \Omega_x^{\max}\right). \tag{27}$$

[3] It is possible to pack the replicas slightly closer together, using a separation between Ω_t^* and $\Omega_t^*/2$. This leads to aliasing in F, but avoids aliasing in the final lower-frequency motion-blurred result H. For simplicity, we avoid that discussion here, which only corresponds to a factor of at most 2. The sheared filter in Sec. 5 focuses primarily on non-axis-aligned reconstruction, but does also exploit this small factor.

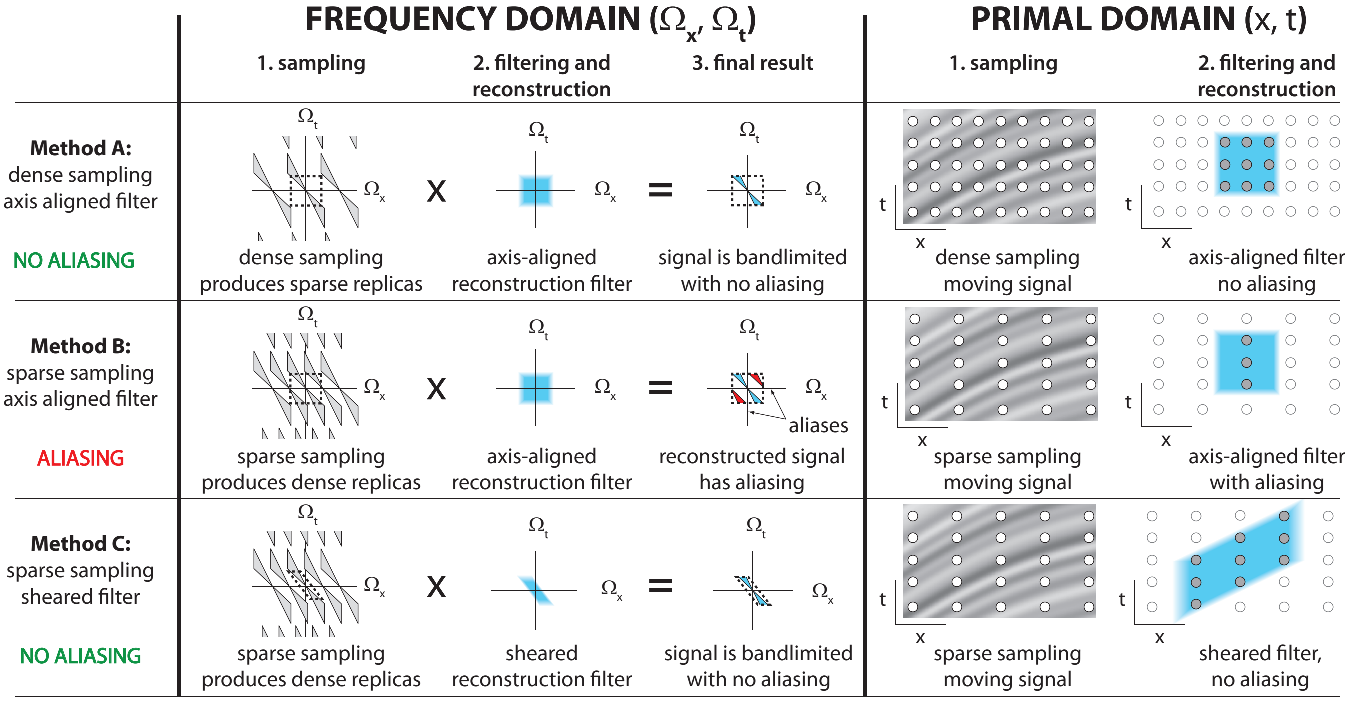

FREQUENCY DOMAIN (Ωx, Ωt)
PRIMAL DOMAIN (x, t)
1. sampling
2. filtering and reconstruction
3. final result
1. sampling
2. filtering and reconstruction
Method A: dense sampling axis aligned filter
NO ALIASING
Ωt
Ωx
X
=
dense sampling produces sparse replicas
axis-aligned reconstruction filter
signal is bandlimited with no aliasing
t
x
dense sampling moving signal
axis-aligned filter no aliasing
Method B: sparse sampling axis aligned filter
ALIASING
aliases
sparse sampling produces dense replicas
axis-aligned reconstruction filter
reconstructed signal has aliasing
sparse sampling moving signal
axis-aligned filter with aliasing
Method C: sparse sampling sheared filter
NO ALIASING
sparse sampling produces dense replicas
sheared reconstruction filter
signal is bandlimited with no aliasing
sparse sampling moving signal
sheared filter, no aliasing

Figure 7: 1. Sampling in the primal domain creates replicas in the frequency domain. The denser the sampling the further apart the Fourier-domain replicas are spaced. 2. The Fourier transform of the spatial reconstruction filter bandlimits and reconstructs the signal for display. 3. Filtering the samples in the primal domain is equivalent to multiplying the Fourier transforms of steps 1 and 2. If replicas overlap with the Fourier domain reconstruction filter, the final result would contain spurious frequencies (aliasing). In Method/Row A, a relatively dense sampling is used in space and time to separate the Fourier domain replicas. Method/Row B shows that a sparser sampling rate with an axis-aligned filter leads to aliasing. Method/Row C shows that using a sheared reconstruction filter, we can reconstruct a correct image using a sparse sampling rate.

Sampling Theorem: Sampling the time-varying signal f leads to replicas of $F(\Omega_x, \Omega_t)$ in the Fourier domain as shown in Figure 6(c). We must separate the replicas enough to avoid overlap or aliasing in reconstructing the motion-blurred signal H. Figure 7 shows this idea in both the space-time and frequency domains.

The exact separation of replicas needed depends on the reconstruction filter, and for now we consider a standard rectangular axis-aligned filter in the Fourier domain (Figures 7(A,B)). It is instructive to consider the product of spatial and temporal frequency ranges. By the Nyquist theorem, the number of samples needed is also proportional to these bandlimits. For simplicity, we use Equations 25 and 26 (ignoring for now the special case in Equation 27),

$$\Omega^* = (\Omega_x^*)(\Omega_t^*) = 4\frac{a_{\max}}{a_{\min}}\Omega_x^{\max}\Omega_t^{\max}. \tag{28}$$

In the limit where we have a uniform velocity with $a_{\max} = a_{\min}$, the space-time sampling rate becomes $\Omega^* = 4\Omega_x^{\max}\Omega_t^{\max}$, *independent of the velocity* a. This indicates that as the motion a gets faster, the needed temporal sampling rate $\Omega_t^* = 2a\Omega_x^{\max}$ increases, but the spatial sampling rate needed $(2/a)\Omega_t^{\max}$ decreases correspondingly due to the spatial filtering or blurring of moving objects and texture.

5 Sheared Reconstruction Filter

We have taken a first step in finding spatial and temporal bandlimits. These bandlimits can directly be used to accelerate motion-blur rendering by adaptive sampling. We may sparsely sample in space and time according to Equations 25 and 27, then scale the standard one pixel wide axis-aligned reconstruction (spatial antialiasing and temporal shutter response) filter to reconstruct the sparse data.

However, Figure 6(c) and Figure 7(A) show the corresponding packing of replicas in Fourier space and illustrate that they still have a lot of free space between them. We seek to achieve sparse sampling, which means bringing the replicas tighter together. Packing replicas too tightly while using an axis-aligned filter will cause aliasing (Figure 7(B)). We now introduce a sheared filter that allows for much tighter packing of replicas and lower sampling densities (Sec. 5.1). It is based on two important observations: the shape of the spectrum is slanted and is best matched by a sheared filter, and we need to prevent overlap only in the central part of the wedge that is within the shutter bandwidth. Finally, we take a critical step towards a practical algorithm by deriving the sheared filter in the primal space-time domain (Sec. 5.2). This is done simply by appropriately transforming a standard axis-aligned filter (Figure 8(d)).

5.1 Sheared Filter and Sampling

As can be seen in Figure 8(a), we are really interested in the central wedge of frequencies for $H(\Omega_x, \Omega_t)$. Given the spectrum's wedge shape, it is best to separate the central spectrum from the replicas by using a non-axis-aligned parallelogram as the reconstruction filter, as shown in Figure 7(C) and Figure 8(a). Figures 8(b) and (c) show two ways of tightly packing the replicas, which we discuss next. Note that the frequency spectra for $F(\Omega_x, \Omega_t)$ do in fact alias in this reconstruction (shown in red). However, the final low-pass filtered form from motion blur $H(\Omega_x, \Omega_t)$ does not. The amount of free space in the Fourier domain is considerably reduced, compared to Figure 6(c), enabling lower sampling rates.

Intuitive Sampling Strategy 1—Pack Space Replicas First: The first sampling method we examine packs replicas tightly in

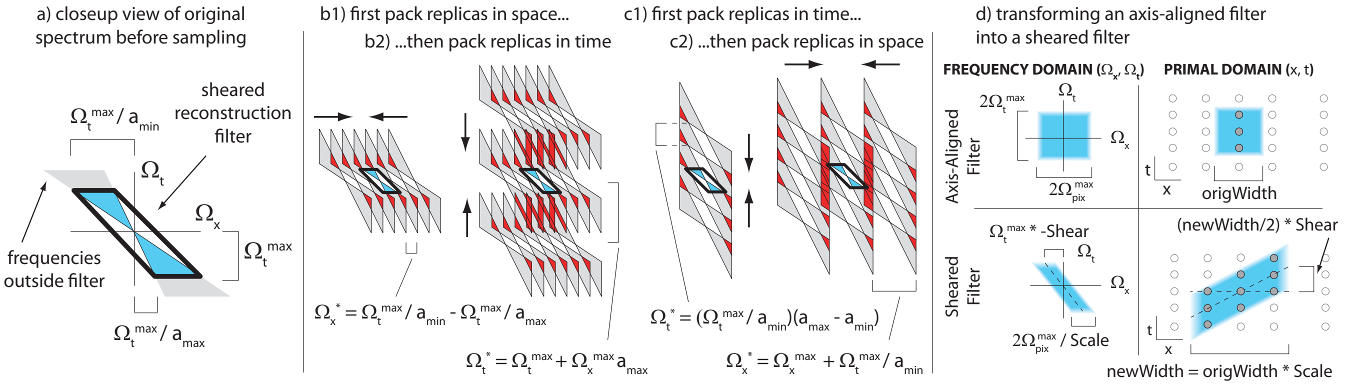

a) closeup view of original spectrum before sampling
sheared reconstruction filter
Ω_t^{max} / a_{min}
Ω_t
Ω_x
Ω_t^{max}
frequencies outside filter
Ω_t^{max} / a_{max}
b1) first pack replicas in space...
b2) ...then pack replicas in time
$\Omega_x^* = \Omega_t^{max} / a_{min} - \Omega_t^{max} / a_{max}$
$\Omega_t^* = \Omega_t^{max} + \Omega_x^{max} a_{max}$
c1) first pack replicas in time...
c2) ...then pack replicas in space
$\Omega_t^* = (\Omega_t^{max} / a_{min})(a_{max} - a_{min})$
$\Omega_x^* = \Omega_x^{max} + \Omega_t^{max} / a_{min}$
d) transforming an axis-aligned filter into a sheared filter
FREQUENCY DOMAIN (Ω_x, Ω_t)
PRIMAL DOMAIN (x, t)
Axis-Aligned Filter
$2\Omega_t^{max}$
Ω_t
Ω_x
$2\Omega_{pix}^{max}$
t
x
origWidth
Sheared Filter
Ω_t^{max} * -Shear
Ω_t
Ω_x
$2\Omega_{pix}^{max}$ / Scale
(newWidth/2) * Shear
t
x
newWidth = origWidth * Scale

Figure 8: *(a) Zoomed-in view of the frequency wedge and the sheared reconstruction filter. The distances between the Ω_t axis and the near and far points of the sheared filter are shown. Only the blue frequency content inside of the sheared reconstruction filter will be output for display. (b1) and (b2) show packing of replicas as tightly as possible first in space, then in time. (c1) and (c2) show packing of replicas as tightly as possible first in time, then in space. (d) Transforming an axis-aligned filter into a sheared filter. (d Top) We start with any standard axis-aligned filter in the frequency and primal domains. (d Bottom) We then consider the scale and shear in the frequency domain, applying the opposite scale and shear in the space-time domain (Equations 33 and 34).*

Ω_x, then in Ω_t, as shown in Figure 8(b1-b2). This technique more closely follows Sec. 4 and is useful for developing our intuition for the benefits obtained from the sheared filter. Our practical algorithm uses the second sampling strategy, developed next, of packing the time replicas first.

First we compute the spatial sampling rate or bandlimit Ω_x^*. From simple trigonometry, Figure 8(b1), and Equation 26,

$$\Omega_x^* = \Omega_t^{\max}\left(\frac{1}{a_{\min}} - \frac{1}{a_{\max}}\right), \tag{29}$$

which is a significantly lower frequency (and hence sampling rate) than in Equation 26 when $a_{\min}$ is close to $a_{\max}$. Indeed, for nearly uniform velocity $a_{\min} \approx a_{\max}$, we obtain $\Omega_x^* \to 0$ (assuming the reconstruction filter extends infinitely far in space-time). As seen in Figure 8(b2), we must next pack the temporal replicas to determine Ω_t^*, and can then compute an overall bandlimit, $\Omega^* = \Omega_x^* \Omega_t^*$.

Practical Sampling Strategy 2—Pack Time Replicas First: It is also possible to proceed the other way, first packing along the temporal axis and then along the spatial axis (an illustration is in Figures 8(c1) and (c2)). This formulation gives essentially the same overall sampling rate Ω^* as the first sampling strategy above, and has advantages in practical applications where we usually want the spatial samples denser than the temporal samples—with very few time samples required for high-quality motion blur. Having dense spatial sampling makes it easier to find high-frequency spatial discontinuities that can be caused by static occluders. Note that the sheared filter itself is the same in both cases.

From the geometry of Figures 8(c1) and (c2), we can derive

$$\Omega_t^* = \Omega_t^{\max}\left(\frac{a_{\max}}{a_{\min}} - 1\right) \qquad \Omega_x^* = \Omega_x^{\max} + \frac{\Omega_t^{\max}}{a_{\min}}, \tag{30}$$

with the product being given by

$$\Omega^* = \Omega_x^* \Omega_t^* = \left(\frac{a_{\max}}{a_{\min}} - 1\right)\Omega_x^{\max}\Omega_t^{\max} + \left(\frac{a_{\max}}{a_{\min}} - 1\right)\frac{(\Omega_t^{\max})^2}{a_{\min}}, \tag{31}$$

which is also proportional to the total number of samples needed.

With motion greater than 1 pixel per frame $a_{\min}\Omega_x^{\max} > \Omega_t^{\max}$, and the first term above will be dominant. Equation 31 is now

$$\boxed{\Omega^* \approx \left(\frac{a_{\max}}{a_{\min}} - 1\right)\Omega_x^{\max}\Omega_t^{\max}.} \tag{32}$$

The crucial benefit over Equation 28 is the use of $a_{\max}/a_{\min} - 1$ instead of $a_{\max}/a_{\min}$. If maximum and minimum velocities at a pixel for a given frame are similar, sheared reconstruction can be significantly more efficient. On the other hand, for pixels with significant occlusions or large velocity changes so $a_{\max}/a_{\min} \gg 1$, we cannot do much better than falling back to a standard rectilinear filter.

5.2 Sheared Filter in Primal Domain

So far, we have considered frequency analysis, but practical rendering algorithms do not directly compute frequency spectra. Fortunately, we can create a sheared reconstruction filter directly in the space-time domain. We simply apply the corresponding transforms to any standard axis-aligned filter composed of a spatial antialiasing filter and the temporal shutter response (see Figure 8(d)).

Specifically, the original axis-aligned filter has some spatial bandlimit $\Omega_{\text{pix}}^{\max}$ ($\approx$ 0.5 wavelengths per pixel) that we scaled (Figure 8(a)) to a diameter of $\Omega_t^{\max}(1/a_{\min} - 1/a_{\max})$. Based on Fourier theory, we must scale by the inverse in the primal domain:

$$\text{Scale} = \left(\frac{\Omega_t^{\max}}{2\Omega_{\text{pix}}^{\max}}\left(\frac{1}{a_{\min}} - \frac{1}{a_{\max}}\right)\right)^{-1}. \tag{33}$$

The shear of the filter in the Fourier domain is based on the filter intercepts $\Omega_t^{\max}/a_{\min}$ and $\Omega_t^{\max}/a_{\max}$ (Figure 8(a)). In the Fourier domain the shear in Ω_x per unit Ω_t is the average of $-1/a_{\min}$ and $-1/a_{\max}$. Again, based on Fourier theory, we need to apply the opposite shear in the primal domain (shearing in time per unit x):

$$\text{Shear} = \frac{1}{2}\left(\frac{1}{a_{\max}} + \frac{1}{a_{\min}}\right). \tag{34}$$

The shear corresponds to the direction of average motion in the space-time domain, with the filter "following the motion." The scale depends on the complexity of motion—the filter is larger (with a corresponding low sampling rate), the closer $a_{\min}$ and $a_{\max}$ are.

6 Algorithm and Results

We describe one approach for using these theoretical results—a simple practical method that uses sheared reconstruction filters to greatly reduce sample counts. While the analysis is in the Fourier domain, the actual practical algorithm need not explicitly compute spectra, and operates directly on space-time image samples.

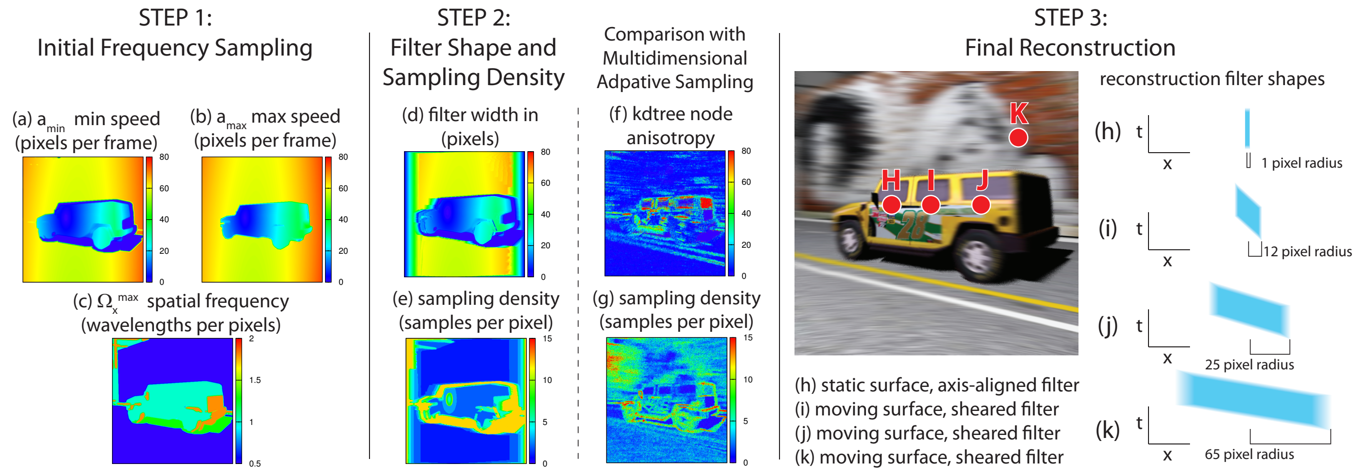


Figure 9: *Illustration of our three-stage algorithm. The scene is shown in Figure 1. In Step 1, we do an initial sampling to compute velocities* $[a_{\min}, a_{\max}]$ *and maximum spatial frequencies* $\Omega_x^{\max}$*. These are visualized in (a,b) and (c) respectively. Note that areas of the image where* $a_{\min} \approx a_{\max}$ *will require very low sample counts as seen in (e). In Step 2, we determine the sheared filter shape, and the sampling densities at each pixel. The filter radius is shown in (d); note the very large filters for the background. (e) visualizes the total number of samples* Ω^* *at a pixel, and is seen to be high only in regions where the motion is non-uniform, such as the ends of the car (occlusion) and shadows moving over a textured surface (multiple signals). The edges of the image also require higher sampling. For areas of uniform motion a low sample count (often close to 1 per pixel) suffices. We also compare with MDAS, which adapts to the silhouette edges of the car, but has a more uniform sample distribution, and much less anisotropy in the filters. Finally, Step 3 shows the sheared filter shape for representative pixels. In general (i,j,k), sheared filters of different widths are used, based on the speed of motion. For the special case of a static surface (sharp with minimal motion blur) like (h), our method gracefully reduces to a 1-pixel axis-aligned filter as required.*

Our method involves a three-stage process, shown in Figure 9. First, we do an initial sparse sampling to compute the effective velocities $[a_{\min}, a_{\max}]$ and frequency bounds $\Omega_x^{\max}$ (Sec. 6.1). Second, we determine a single sheared reconstruction filter for each pixel, along with spatial and temporal sampling densities Ω_x^* and Ω_t^* (Sec. 6.2). Our third stage involves a final round of sampling, and for each pixel we do a single application of the computed sheared filter to reconstruct the pixel's final color (Sec. 6.3). There are a few additional special cases and implementation details in the appendix.

Our sheared reconstruction filter uses the sampling formulation in Figure 8(c). This method samples sparsely in time (packing the replicas tightly in the temporal frequency domain), but densely in space. For frequencies $\Omega_t^{\max}$ and $\Omega_x^{\max}$ in practical images, we sample every pixel of the frame at least once, but with many fewer samples than are required for the same quality output using standard Monte Carlo sampling. The source code for our program can be found at `http://www.cs.columbia.edu/cg/mb/`, and we include a Renderman shader that computes the relevant velocities and frequency bounds in our supplementary material.

6.1 Stage 1: Velocity/Frequency Bounds

We start by sparsely computing local frequency information at each pixel. We sample the scene with N samples per pixel (our implementation uses $N = 2$). This cost is minimal, since it is less than what we would need to render a single antialiased image of a static scene. Moreover, only velocity information is required from the samples; all shading is computed in a separate pass in step 3. At each sample we compute the image space signal direction, velocity bounds $[a_{\min}, a_{\max}]$, and signal bandlimit $\Omega_x^{\max}$.

Computing Effective Velocities and Bandlimits: We use surface shaders to compute the image-space velocities within the renderer for each of the three key signals: object motion, BRDF shading, and shadows.

For object motion, the effective velocity a is simply the instantaneous screen-space velocity for the surface (including projection and perspective effects). Assuming the surfaces use mip-mapping to bandlimit texture frequencies, we simply set $\Omega_x^{\max}$ to a maximum frequency of one wavelength per pixel.

The velocities and bandlimits for BRDF shading and shadows are based on Equations 18 and 24 respectively,

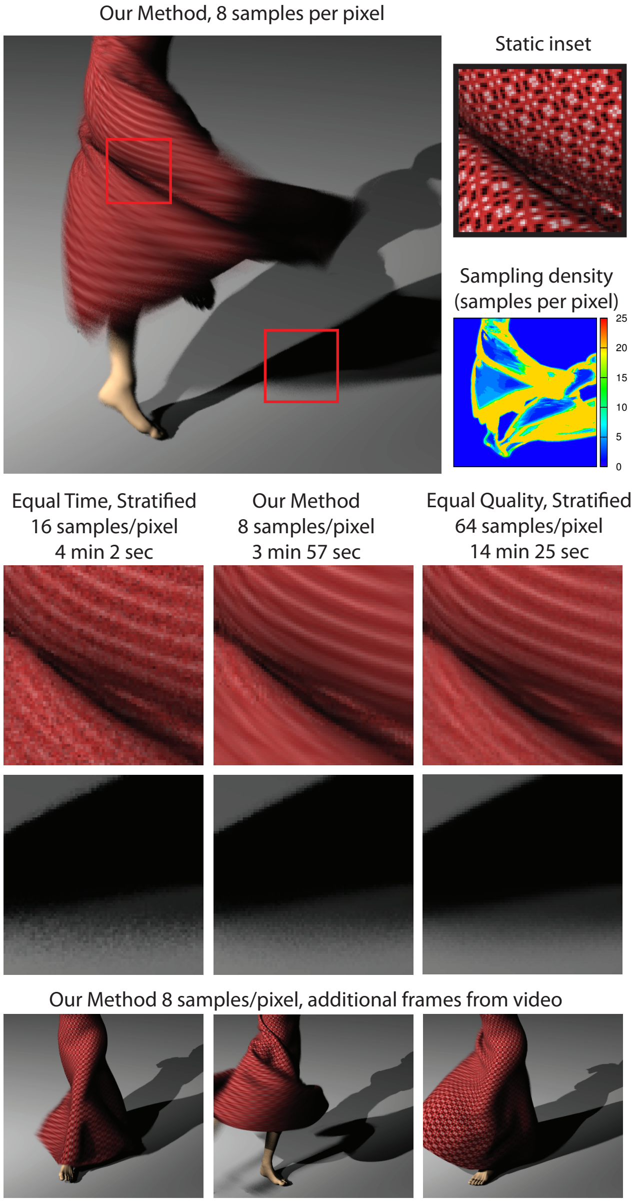
$$a_{shading} = \frac{\gamma}{\kappa'} = \frac{\alpha\kappa' + \beta}{\kappa'} \qquad \Omega^{\max}_{x,shading} = \min\left(L^{\max}\kappa', R^{\max}\kappa'\right) \quad (35)$$

$$a_{shadow} = \frac{\gamma}{\nu} = \frac{\alpha\nu + \beta}{\nu} \qquad \Omega^{\max}_{x,shadow} = \min\left(L^{\max}\nu, S^{\max}\nu\right), \quad (36)$$

where α is the angular velocity of the lighting, β is the linear velocity of the object or blocker, $\kappa' = 2\kappa$ is twice the screen-space curvature, and $|\nu| \sim \cos\mu/D$, where D is the distance to the blocker. These values can be calculated entirely inside the shader if the programmable shading language supports shader derivatives. Details on units and computing frequency bandlimits are in the appendix.

Velocity and Frequency Bounds: After initial sampling, we compute a frequency bound for each pixel that captures frequency information across the entire frame. $[a_{\min}, a_{\max}]$ are simply the minimum and maximum velocities of all samples inside the pixel. Similarly, $\Omega_x^{\max}$ is simply the maximum frequency of all samples. Values for the scene in Figure 1 are shown in Figure 9(a,b,c).

Multiple Signals: Our theory focuses on the case where all signals (object motion, lighting, shadows) that affect a given pixel translate along a single direction. Therefore, in the special case that any of the frequency samples at a pixel has a direction vector that differs greatly in angle from the others, we conservatively bound the frequencies by setting $a_{\min}$ for that pixel to 0 (the computations of $a_{\max}$ and $\Omega_x^{\max}$ are unaffected). When we are calculating a single frequency sample, a similar adjustment is occasionally required when multiple signals (more than one of surface texture, BRDF shading and shadows) have significant amplitude and frequency. Details for this case are discussed in the appendix.



Our Method, 8 samples per pixel

Static inset

Sampling density (samples per pixel)

Equal Time, Stratified 16 samples/pixel 4 min 2 sec

Our Method 8 samples/pixel 3 min 57 sec

Equal Quality, Stratified 64 samples/pixel 14 min 25 sec

Our Method 8 samples/pixel, additional frames from video

Figure 10: *A scene of a ballerina with fast and varying motions. The dress is deforming as the dancer kicks, causing many non-uniform motions that stress the abilities of any motion-blur algorithm. Note that we focus samples on the most difficult areas: occlusion of the bottom of the dress, overlapping shadows, and areas of the dress that come out of shadow as the dancer rotates. It is clear from the insets that our method does not blur frequency content perpendicular to the direction of motion.*

6.2 Stage 2: Sheared Filters and Sampling Rates

Based on the velocities and the frequency information from our initial sampling, we compute sheared reconstruction filters and sampling densities for each pixel. These are visualized in Figures 9(d) and (e) for an example scene. To derive properties for sheared filters in Sec. 5, we first determined the shape of the filter in Fourier space, and then determined how tightly we could pack replicas. Similarly, in our practical implementation, for each pixel, we first compute the widest possible reconstruction filter, and then determine the lowest possible sampling rate that avoids aliasing.

Computing the Shape of the Sheared Filter: To create an optimal sheared filter we use Equations 33 and 34 to scale and shear the user's preferred axis-aligned reconstruction filter. In image space, both the scale and shear operate strictly along the direction of motion, and the axis perpendicular to motion is unaffected.

To provide intuition, consider the case of nearly constant velocity where $a_{\min} \approx a_{\max} = a$. In this case, the space-time shear (Equation 34) is just $1/a$, as expected. The scale tends to infinity (since $1/a_{\min} \approx 1/a_{\max}$ in Equation 33)—we can use a very wide sheared filter in this case, since the velocity is constant. Indeed, very wide filters are used in Figure 9(d) for much of the car, and especially the background, which have nearly uniform velocity.

A special case arises for slow-moving signals (as indicated by Equation 27), and its handling is discussed in the appendix. The final computation of filter widths is also complicated by the fact that once we select a filter size, we may include incompatible pixels inside the filter. For instance, in the example above, if $a_{\min} \approx a_{\max}$, the scale should be very large, but this wide filter may contain other pixels with a greater range of $[a_{\min}, a_{\max}]$. Computation of a final filter shape may require an iterative process where we eventually use a smaller filter size. Details are given in the appendix.

Computing Sample Densities: In most cases, we can compute the sampling rates Ω_x^* and Ω_t^* directly from Equation 30. For scenes with moderate complexity, Ω_x^* and Ω_t^* usually require at least one sample per pixel per frame. To compute the sampling rate for a 2D image we must also include frequencies along the spatial axis perpendicular to motion. These frequencies should have little or no velocity, so we use the spatial bandlimit for static signals with an axis-aligned filter, $(2\Omega_x^{\max})$ (Equation 27):

$$\text{Pixel Samples} = \Omega^* = (\Omega_x^*)(\Omega_t^*)(2\Omega_x^{\max}). \qquad (37)$$

In practice, we also cap the maximum number of samples for a pixel (usually to 4× the average number of samples per pixel).

The number of samples depends on both spatial complexity and motion complexity (how much it differs from uniform velocity). More samples will be given both where the motion varies ($a_{\max}/a_{\min}$ is large), and also where there are high spatial frequencies (complex textures or shadows/highlights with high $\Omega_x^{\max}$). Equation 37 provides a natural way to allocate samples to different visual effects.

6.3 Stage 3: Final Sampling and Reconstruction

The final sampling density for a pixel is simply the maximum density required by reconstruction filters that overlap that pixel. For sample placement across both space and time within a pixel, we use a 3D Halton sequence [Halton 1960]. For low sampling densities (less than 8 samples per pixel), we do not jitter, and we mirror the Halton sequence at odd pixels. For higher sampling rates, we add a jittered offset. We limit our sample points to lie inside the shutter bounds to show compatibility with traditional rendering pipelines. Future implementations could find gains by sampling across time and sharing samples between frames of an animation.

We send the computed space-time sample locations to the renderer for processing, and read back the shaded results for each sample. Finally, we reconstruct the motion-blurred image using the sheared filters computed in step 2. Note that we do only one reconstruction per pixel, with a single application of the sheared filter for that pixel—this filter combines reconstruction, spatial antialiasing, and motion-blur integration over time. Figure 9(h,i,j,k) shows these filters for some representative image pixels. In most cases, these are sheared, with the size of the filter determined by the complexity of the motion (or getting clipped by the camera shutter bounds). In the special case of static regions (Figure 9(h)), the filter reduces to an axis-aligned filter of 1 pixel width, as it must.

6.4 Results

We modified the Pixie renderer [Arikan 2009] to use our space-time sample placement algorithm for ray tracing color information in stage 3. We now describe the results obtained by our algorithm and compare to a stratified sampling Monte Carlo approach using jittering, and to the recent MDAS technique [Hachisuka et al. 2008]. All images were rendered at a resolution of 512×512 with a single core on a 1.8GHz Core 2 Duo processor.

Scenes: Figures 1, 9, and 13 show a scene with a rotating camera, and a moving object (the car). Following a common photographic technique, the camera follows the car's motion to keep it sharp (but not completely stationary), while the other areas have considerable motion blur. The lighting comes from a moderately distant source. Figure 10 is intended to be a stress test of our system, with a deforming dress and multiple non-rigid motions. The dress has a high-frequency texture, which leads to different patterns depending on the direction of motion. Moreover, we have mid-field lighting from two sources that cast overlapping moving shadows, and surfaces moving in and out of shadow as well as self-occluding.

Finally, Figure 11 shows an example with a motion-blurred glossy reflection of the moving background (the teapot is shiny with Phong exponent 100). This scene demonstrates that we can handle curved motion paths and global illumination effects. (Note that calculating motion-blurred global illumination effects requires that the shader can calculate the movement of indirect lighting.) The scene also has motion-blurred reflections of near sources, and sharp shadows on moving surfaces (shadow of the spout).

Evaluation and Comparison to Stratified Monte Carlo: In Figure 1b, stratified Monte Carlo sampling with 4 samples/pixel leads to considerable noise, especially in the motion-blurred areas of the background. In contrast, our method (Figures 1(a,d)) produces a high-quality result even at this very low sample count, with only minimal noise at difficult shadow boundaries. It would require at least an order of magnitude more samples to match it with direct Monte Carlo. Similar conclusions can be drawn from Figure 10. Our implementation properly computes motion blur and preserves high frequencies on the dress perpendicular to the direction of motion. Our method can also be used directly to produce motion-blurred sequences, as shown in the supplementary animation (stills in Figure 10). Note also that proper motion-blurred lighting and shadows are computed in Figures 1, 10 and 11.

Finally, the sheared filter can also be used by itself as a light-weight addition with standard Monte Carlo sampling and rendering (Figure 12). Applying sheared reconstruction to the standard (non-adaptive) stratified sampling pattern dramatically improves areas of uniform motion like the dress (Figure 12(b)). Of course, also using our adaptive sampling enables more samples at the dress silhouette, leading to a reduction in noise (Figure 12(c)).

Sampling Densities and Filter Widths: The filter widths and sampling densities for our method are shown in Figures 9(d,e) and in Figure 10. Interestingly, in the car scene, very few samples are needed for much of the image, where motion although fast, is almost uniform. Note in Figure 9(e) that an average of close to only 1 sample per pixel suffices on much of the car body, background painting, and road. Samples are thus concentrated near the silhouette boundaries and edges of cast shadows, where the motions are very complex (also the case in Figure 10). The widths of the sheared reconstruction filter in Figure 9(d) clearly show how large our sheared filter can be in regions of nearly uniform motion such as the background, and much of the car body. Moreover, our method can gracefully fall back to axis-aligned reconstruction with 1 pixel-wide filters, in difficult or nearly static areas of the car (Figure 9h).

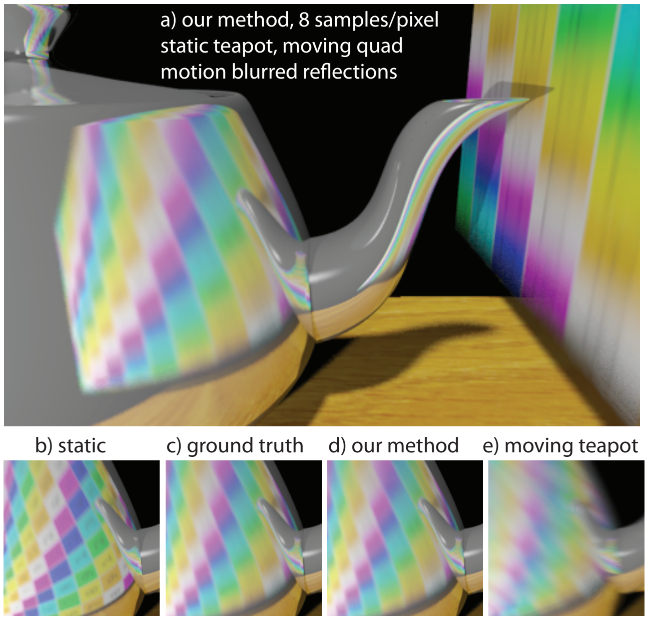

a) our method, 8 samples/pixel
static teapot, moving quad
motion blurred reflections
b) static
c) ground truth
d) our method
e) moving teapot

Figure 11: *Shiny teapot (Phong exponent 100) with glossy reflections, rendered with an average of 8 samples/pixel. (a) Full image. The insets below show (b) static image, (c) ground truth, (d) our method, (e) our method where the teapot is also moving.*

Comparison to Multi-Dimensional Adaptive Sampling: For comparisons on the car scene, we directly used MDAS software which is a plugin to the PBRT renderer [Pharr and Humphreys 2004]. We did not do this comparison for the ballerina because the base PBRT renderer does not currently support deforming meshes.

First, consider the sampling rates in Figure 9. MDAS (Figure 9g) has a much more uniform sampling density, with only a little more importance given to the edges of the car. Our technique is able to sample many motions more sparsely, allowing it to properly focus on difficult areas. Similarly, while MDAS takes anisotropy into account (Figure 9f), their kD-tree cells never shear and do not scale as much as our sheared filter does. In contrast, our reconstruction filter can rotate in any direction in image space and is sheared in time to better match the direction of motion.

Figure 13 shows what happens as we increase sample count from an average of 2 per pixel to 8 per pixel (4 samples/pixel is shown in Figure 1). At low sample counts (top row of Figure 13), we are already nearly perfect in the background and ground plane because of our wide sheared reconstruction filter, while MDAS is very noisy. On the other hand, MDAS is somewhat less noisy near shadow boundaries, since our method often needs to fall back to axis-aligned reconstruction for these complex motions. This is because MDAS discovers areas of coherence through numerical measurements, whereas we rely on conservative frequency bounds. At moderate sample counts (bottom row of Figure 13), both methods are close to converged, but MDAS still has a little noise on the background mural.

Timings and Overheads: At 8 samples per pixel, the car scene took our algorithm a total of 3 min, 8 sec, while the ballerina took 3 min, 57 sec. The time for reconstruction using sheared filters is about half the total for the car, and one-third for the ballerina. As one point of comparison, we are competitive with MDAS (a total of 3 min, 23 sec on the car, with the overhead for reconstruction being about one third of total running time). Moreover, MDAS has significant memory overheads. In our tests MDAS required 1GB of memory to render a 512×512 image with 8 samples per pixel. Our

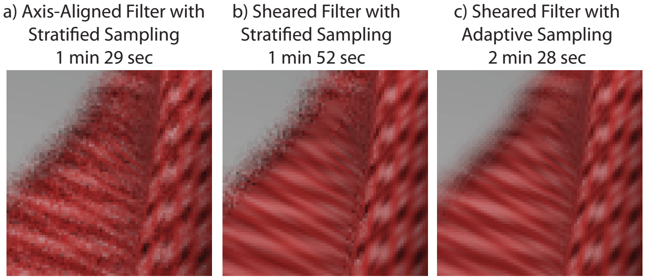

a) Axis-Aligned Filter with Stratified Sampling 1 min 29 sec
b) Sheared Filter with Stratified Sampling 1 min 52 sec
c) Sheared Filter with Adaptive Sampling 2 min 28 sec

Figure 12: *An inset from the left video frame in Figure 10. (a) Standard Monte Carlo stratified sampling exhibits fairly uniform noise. (b) Monte Carlo stratified sampling, combined with our sheared space-time filter. Noise has been reduced in areas of uniform motion. (c) Our adaptive method samples densely around difficult regions so that the noise in (b) at the silhouette of the dress is reduced.*

memory overhead is fairly low. To store all samples in memory for a 512×512 image requires 36MB during sparse sampling ($N = 2$), and 32MB during final sampling (using 8 samples per pixel).

In scenes with more complex shading, our method has much lower overhead (only 15% of the 25 minute rendering time for Figure 11). Even with very simple materials, such as Figure 10, we are only twice as slow as standard Monte Carlo (some of this is from overhead, and an equal amount from the fact that our ray-tracing phase focuses on difficult regions by design, which take more time). A visual equal quality comparison shows a net wall-clock speedup of more than 3.5× by our method in Figure 10.

Limitations: Our current implementation uses a line segment as the anisotropic filter shape. For highly-curved motion paths this may lead to over-blurring. Note that we do test all samples inside a filter to measure non-linearities in speed (divergence of $a_{\min}$ and $a_{\max}$) and direction. In particular, we set $a_{\min}$ to 0 if any direction differs significantly in angle from the others. These tests will cause our method to reduce filter sizes and increase sampling rates near areas of highly curved motion. With more complete motion information from the renderer, future implementations should be able to filter along curved paths.

Our method can resort to axis-aligned reconstruction and dense sampling in difficult cases, such as a shadow moving over a static textured surface. However, these areas are also difficult in most other motion-blur techniques. Moreover, our method can quickly converge on simpler parts of the scene, and then focus almost all of its sample budget on the difficult regions.

Similar to any practical rendering application, our system makes approximations during reconstruction that can lead to aliasing, (such as using a windowed gaussian filter instead of an infinitely wide sinc filter). Because wide filters overlap and share information, adjacent pixels employing wide filters will share aliasing data. The net visual effect is usually a small distortion in the image relative to ground truth. Note that these aliases are low frequency, and are visually hard to detect, nor do they cause temporal artifacts as seen in the video. Because of this, our method will often achieve an excellent visual match with ground truth, but a relatively high measure of mean squared error.

Like most adaptive techniques, our implementation does an initial sampling and can therefore miss information from very fast-moving or thin objects. In the future, we could try using space-time bounding boxes to more conservatively bound occlusions.

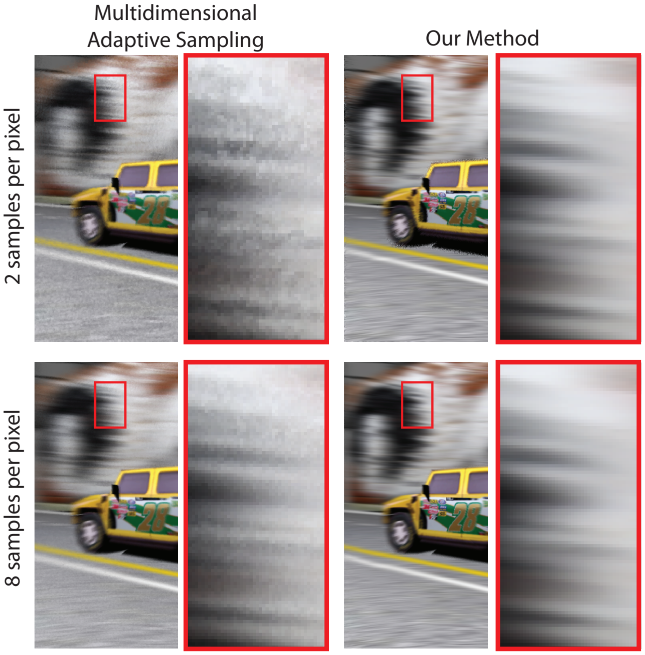


Figure 13: *We show further results for our method and MDAS with 2 samples per pixel and 8 samples per pixel. For 2 samples per pixel our method does well for areas with uniform motion (in fact many areas use one sample per pixel and are in their final state), but does poorly for pixels where we detect a large difference between $a_{\min}$ and $a_{\max}$. At 8 samples per pixel our method devotes all of the new samples to the difficult areas, and many places such as the shadows improve dramatically. MDAS by comparison starts out with fairly uniform noise, and then improves evenly over all areas of the image. The insets show that in areas with fairly uniform motion our method computes high-quality results at extremely low sample counts.*

7 Conclusion

We have presented a frequency-space analysis of time-varying signals and images, as needed for motion blur. We have shown that most motion-blur effects can be analyzed as shears in the space-time and Fourier domains. Our analysis gives precise guidelines for an intuitive observation: integrating over the shutter blurs a moving signal in the spatial dimension. This analysis in turn leads to a novel sheared reconstruction filter that again formalizes an intuitive notion: a moving sample should contribute not just to the current spatial location (pixel), but to other pixels at corresponding time instances, depending on its velocity.

For future work we would like to analyze a larger class of indirect lighting effects, as well as develop a general time-varying frequency analysis for light transport. We would also like to generalize our insights to other problems involving sheared signals.

We have developed the first time-dependent Fourier theory in rendering. Since the time dimension is of increasing importance in image synthesis, we consider this a significant step in a key area. We foresee many future developments that couple new theoretical advances with novel space-time sampling and filtering strategies.

Acknowledgements

This work was supported in part by an ONR Young Investigator Award N00014-07-1-0900, ONR grant N00014-09-1-0741, NSF Graduate Fellowship, NSF grants 0446916, 044756, and 0701775, two Sloan Research Fellowships, a Microsoft New Faculty Fellowship, and the INRIA Associate Research Team ARTIS/MIT. This research was started while N. Holzschuch was on a sabbatical at Cornell University, funded by INRIA. LJK is UMR 5224, a joint research laboratory of CNRS, INRIA, INPG, U. Grenoble I and U. Grenoble II. We are grateful to Pixar for PRMan licenses and to Intel and NVIDIA for equipment donations. The ballerina animation is courtesy of Daniel Genrich; the car model is courtesy of TurboSquid user graphicdoom; the graffiti photo in Figure 1 is courtesy of Wikipedia user Jongleur100; the racecar photo in Figure 2 is courtesy of Brian Snelson; and the St. Peter light probe is courtesy of Paul Debevec. Thanks to Toshiya Hachisuka for use and support of MDAS code, and Okan Arikan for support of Pixie.

References

Akenine-Möller, T., Munkberg, J., and Hasselgren, J. 2007. Stochastic Rasterization using Time-Continuous Triangles. In *Graphics Hardware*, 7–16.

Arikan, O., 2009. Pixie - Open Source RenderMan. http://www.renderpixie.com.

Cammarano, M., and Jensen, H. W. 2002. Time Dependent Photon Mapping. In *EG Symposium on Rendering*, 135–144.

Catmull, E. 1984. An Analytic Visible Surface Algorithm for Independent Pixel Processing. In *Computer Graphics (Proceedings of SIGGRAPH 84)*, ACM, vol. 18, 109–115.

Chai, J., Tong, X., Chan, S., and Shum, H. 2000. Plenoptic Sampling. In *Proceedings of SIGGRAPH 2000*, ACM, 307–318.

Christmas, W. J. 1998. Spatial Filtering Requirements for Gradient-Based Optical Flow Measurement. In *British Machine Vision Conference*, 185–194.

Cook, R. L., Porter, T., and Carpenter, L. 1984. Distributed Ray Tracing. In *Computer Graphics (Proceedings of SIGGRAPH 84)*, ACM, vol. 18, 137–145.

Cook, R. L., Carpenter, L., and Catmull, E. 1987. The Reyes Image Rendering Architecture. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, ACM, vol. 21, 95–102.

Durand, F., Holzschuch, N., Soler, C., Chan, E., and Sillion, F. X. 2005. A Frequency Analysis of Light Transport. *ACM Transactions on Graphics (SIGGRAPH) 24*, 3, 1115–1126.

Hachisuka, T., Jarosz, W., Weistroffer, R., Dale, K., Humphreys, G., Zwicker, M., and Jensen, H. 2008. Multidimensional Adaptive Sampling and Reconstruction for Ray Tracing. *ACM Transactions on Graphics (SIGGRAPH) 27*, 3, 33:1–33:10.

Haeberli, P., and Akeley, K. 1990. The Accumulation Buffer: Hardware Support for High-Quality Rendering. In *Computer Graphics (Proceedings of SIGGRAPH 90)*, ACM, vol. 24, 309–318.

Halton, J. H. 1960. On the Efficiency of Certain Quasi-Random Sequences of Points in Evaluating Multi-Dimensional Integrals. *Numerische Mathematik 2*, 1, 84–90.

Isaksen, A., McMillan, L., and Gortler, S. J. 2000. Dynamically Reparameterized Light Fields. In *Proceedings of SIGGRAPH 2000*, ACM, 297–306.

Korein, J., and Badler, N. 1983. Temporal Anti-Aliasing in Computer Generated Animation. In *Computer Graphics (Proceedings of SIGGRAPH 83)*, ACM, vol. 17, 377–388.

Levin, A., Sand, P., Cho, T. S., Durand, F., and Freeman, W. T. 2008. Motion-Invariant Photography. *ACM Transactions on Graphics (SIGGRAPH) 27*, 3, 71:1–71:9.

Loviscach, J. 2005. Motion Blur for Textures by Means of Anisotropic Filtering. In *EG Symposium on Rendering*, 105–110.

Mahajan, D., Shlizerman, I. K., Ramamoorthi, R., and Belhumeur, P. 2007. A Theory of Locally Low Dimensional Light Transport. *ACM Transactions on Graphics (SIGGRAPH) 27*, 3, 62:1–62:10.

Max, N. L., and Lerner, D. M. 1985. A Two-and-a-Half-D Motion-Blur Algorithm. In *Computer Graphics (Proceedings of SIGGRAPH 85)*, ACM, vol. 19, 85–93.

Mitchell, D. 1991. Spectrally Optimal Sampling for Distribution Ray Tracing. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, ACM, vol. 25, 157–164.

Pharr, M., and Humphreys, G. 2004. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann.

Potmesil, M., and Chakravarty, I. 1983. Modeling Motion Blur in Computer-Generated Images. In *Computer Graphics (Proceedings of SIGGRAPH 83)*, ACM, vol. 17, 389–399.

Ramamoorthi, R., Koudelka, M., and Belhumeur, P. 2004. A Fourier Theory for Cast Shadows. In *European Conference on Computer Vision 2004*, I–146–I–162.

Ramamoorthi, R., Mahajan, D., and Belhumeur, P. 2007. A First-Order Analysis of Lighting, Shading, and Shadows. *ACM Transactions on Graphics 26*, 1, 2:1–2:21.

Soler, C., and Sillion, F. 1998. Fast Calculation of Soft Shadow Textures Using Convolution. In *Proceedings of SIGGRAPH 98*, ACM, 321–332.

Soler, C., Subr, K., Durand, F., Holzschuch, N., and Sillion, F. 2009. Fourier Depth of Field. *ACM Transactions on Graphics 28*, 2, 18:1–18:18.

Sung, K., Pearce, A., and Wang, C. 2002. Spatial-Temporal Antialiasing. *IEEE Transactions on Visualization and Computer Graphics 8*, 2, 144–153.

Walter, B., Arbree, A., Bala, K., and Greenberg, D. P. 2006. Multidimensional Lightcuts. *ACM Transactions on Graphics (SIGGRAPH) 25*, 3, 1081–1088.

Appendix: Implementation Details and Special Cases

Computing Velocities and Bandlimits (Sec. 6.1): Any angular/spatial units can be used as long as $\kappa, \alpha, L^{\max}, R^{\max}, S^{\max}, \nu$ and the corresponding trigonometry functions all use the same units. Corresponding maximum frequencies or bandlimits are expressed in inverse pixel and time units. Analogously, all velocities are calculated by the shader in pixel distances per unit time, and account for projection effects. Calculations of frequency bandlimits are specific to the shading model being used and can be done either before or during rendering. For example, an implementation may dynamically relate the specularity and frequency of a BRDF using an analytic function, but precompute $L^{\max}$ for an environment map by running a Fourier transform. For surface points with occlusion, $S^{\max}$ will usually have infinite frequencies so we can simply set $\Omega^{\max}_{x,shadow} = L^{\max}\nu$.

Finally, note that our implementation sparsely samples frequency information, so we use advection to gather nearby frequency samples that may overlap with the current pixel at a different moment in time.

Multiple Signals (Sec. 6.1): The final color for a single sample is obtained by multiplying the surface texture, BRDF and shadow signals, so we need to bound the frequencies of that product. Occasionally, samples may have more than one signal (texture, BRDF or shadow) each with significant amplitude and frequency. In Fourier space, the product of the signals corresponds to a convolution of spectra. If all signals have similar velocities, the frequencies will lie along the same Fourier line, as will the final spectrum—we can simply add the individual signal bandlimits to obtain $\Omega_x^{\max}$.

However, when two different signals have different effective velocities, we can obtain a spectrum unlike the wedges in Figures 2(i) and 2(j). One example is a textured surface moving vertically and a shadow moving horizontally. We can bound this convolved spectrum by setting $a_{\min} = 0$, take the maximum value of $a_{\max}$, and sum all relevant values of $\Omega_x^{\max}$.

Low Velocities and Axis-Aligned Filters (Sec. 6.2): From Equation 34, we know that the applied shear grows large as $a_{\min}$ decreases. For slow-moving signals there is a crossing point where using a standard axis-aligned filter is preferable. In Equation 27, we saw that the spatial bandlimit Ω_x^* can be less than $\Omega_t^{\max}/a_{\min}$ when $a_{\min}$ is small. For this reason, we fall back to the standard axis-aligned filter when $(\text{FreqSpacing} + \Omega_t^{\max}/a_{\min}) > \Omega_x^{\max}$ (see below for details).

Filter Width (Sec. 6.2): After computing $a_{\max}$, $a_{\min}$, and $\Omega_x^{\max}$ for the samples inside of the current pixel, we can compute a filter shape using Equations 33 and 34. However, if this new filter overlaps with other pixels we must recompute $a_{\max}$, $a_{\min}$, and $\Omega_x^{\max}$ for all pixels inside the filter. The more samples inside the filter, the greater $a_{\max}$, $a_{\min}$, and $\Omega_x^{\max}$ will diverge. For this reason the widest possible filter width may be smaller than the width originally computed using samples only inside the current pixel (this is common when a filter is close to an occlusion discontinuity). We do a binary search to find the widest possible filter width, searching between a scale of 1.0 on the low end, and the scale predicted initially by samples inside the current pixel on the high end.

In cases where the final filter radius (ActualPrimalRadius) is not as wide as the ideal size (IdealPrimalRadius), the shear is unchanged (Equation 34), but the scale is changed (Equation 33). We must adjust our sampling rates (Equation 30 and Figure 8(c)) to account for the fact that we have effectively added FreqSpacing to the radius of our reconstruction filter along the Ω_x axis in the Fourier domain:

$$\text{RadiusRatio} = \text{IdealPrimalRadius}/\text{ActualPrimalRadius}$$

$$\text{FreqSpacing} = (\text{RadiusRatio} - 1)\Omega_{\text{pix}}^{\max}/\text{Scale}.$$

When $(\text{FreqSpacing} + \Omega_t^{\max}/a_{\min}) > \Omega_x^{\max}$ one corner of the filter has passed beyond $\Omega_x^{\max}$ and we switch to using an axis-aligned filter (along with Equations 25, 27, and 37). If we are using a sheared filter we have

$$\Omega_t^* = \left(\text{FreqSpacing} + \frac{\Omega_t^{\max}}{a_{\min}}\right) a_{\max} - \Omega_t^{\max} \tag{38}$$

$$\Omega_x^* = \Omega_x^{\max} + \frac{\Omega_t^{\max}}{a_{\min}} + \text{FreqSpacing}. \tag{39}$$