5. ANTI-ALIASING

5.1. INTRODUCTION

The geometric transformation of digital images is inherently a sampling process. As with all sampled data, digital images are susceptible to aliasing artifacts. This chapter reviews the antialiasing techniques developed to counter these deleterious effects. The largest contribution to this area stems from work in computer graphics and image processing, where visually complex images containing high spatial frequencies must be rendered onto a discrete array. In particular, antialiasing has played a critical role in the quality of texture-mapped and ray-traced images. Remote sensing and medical imaging, on the other hand, typically do not deal with large scale changes that warrant sophisticated filtering. They have therefore neglected this stage of the processing.

Aliasing occurs when the input signal is undersampled. There are two solutions to this problem: raise the sampling rate or bandlimit the input. The first solution is ideal but may require a display resolution which is too costly or unavailable. The second solution forces the signal to conform to the low sampling rate by attenuating the high frequency components that give rise to the aliasing artifacts. In practice, some compromise is reached between these two solutions [Crow 77, 81].

5.1.1. Point Sampling

The naive approach for generating an output image is to perform point sampling, where each output pixel is a single sample of the input image taken independently of its neighbors (Fig. 6.1). It is clear that information is lost between the samples and that aliasing artifacts may surface if the sampling density is not sufficiently high to characterize the input. This problem is rooted in the fact that intermediate intervals between samples, which should have some influence on the output, are skipped entirely.

The Star image is a convenient example that overwhelms most resampling filters due to the infinitely high frequencies found toward the center. Nevertheless, the extent of the artifacts is related to the quality of the filter and the actual spatial transformation.

Figure 6.1 shows two examples of the moire effects that can appear when a signal is undersampled using point sampling. In Fig. 6.2a, one out of every two pixels in the Star image was discarded to reduce its dimension. In Fig. 6.2b, the artifacts of undersampling are more pronounced as only one out of every four pixels is retained. In order to see the small...
images more clearly, they are magnified using cubic spline reconstruction. Clearly, these examples show that point sampling behaves poorly in high frequency regions.

Figure 5.2: Aliasing due to point sampling. (a) 1/2 and (b) 1/4 scale.

There are some applications where point sampling may be considered acceptable. If the image is smoothly-varying or if the spatial transformation is mild, then point sampling can achieve fast and reasonable results.

Aliasing can be reduced by point sampling at a higher resolution. This raises the Nyquist limit, accounting for signals with higher bandwidths. Generally, though, the display resolution places a limit on the highest frequency that can be displayed, and thus limits the Nyquist rate to one cycle every two pixels. Any attempt to display higher frequencies will produce aliasing artifacts such as moire patterns and jagged edges. Consequently, antialiasing algorithms have been derived to bandlimit the input before resampling onto the output grid.

5.1.2. Area Sampling

The basic flaw in point sampling is that a discrete pixel actually represents an area, not a point. In this manner, each output pixel should be considered a window looking onto the input image. Rather than sampling a point, we must instead apply a low-pass filter (LPF) upon the projected area in order to properly reflect the information content being mapped onto the output pixel. This approach, depicted in Fig. 6.4, is called area sampling and the projected area is known as the preimage. The low-pass filter comprises the prefiltering stage. It serves to defeat aliasing by bandlimiting the input image prior to resampling it onto the output grid. In the general case, prefiltering is defined by the convolution integral

$$g(x,y) = \iint f(u,v) \cdot h(x-u, y-v) \cdot du \cdot dv,$$

where $f$ is the input image, $g$ is the output image, $h$ is the filter kernel, and the integration is applied to all $[u,v]$ points in the preimage.

Images produced by area sampling are demonstrably superior to those produced by point sampling. Figure 5.4 shows the Star image subjected to the same downsampling transformation as that in Figure 5.2. Area sampling was implemented by applying a box filter (i.e., averaging) the Star image before point sampling. Notice that antialiasing through area
Digitales Video

sampling has traded moire patterns for some blurring. Although there is no substitute to high resolution imagery, filtering can make lower resolution less objectionable by attenuating aliasing artifacts.

Area sampling is akin to direct convolution except for one notable exception: independently projecting each output pixel onto the input image limits the extent of the filter kernel to the projected area. As we shall see, this constraint can be lifted by considering the bounding area which is the smallest region that completely bounds the pixel's convolution kernel. Depending on the size and shape of convolution kernels, these areas may overlap. Since this carries extra computational cost, most area sampling algorithms limit themselves to the restrictive definition which, nevertheless, is far superior to point sampling. The question that remains open is the manner in which the incoming data is to be filtered. There are various theoretical and practical considerations to be addressed.

![Figure 5.3: Area sampling.](image)

![Figure 5.4: Aliasing due to area sampling. (a) 1/2 and (b) 1/4 scale.](image)
5.1.3. **Space-Invariant Filtering**

Ideally, the sinc function should be used to filter the preimage. However, as discussed in Chapters 4 and 5, an FIR filter or a physically realizable IIR filter must be used instead to form a weighted average of samples. If the filter kernel remains constant as it scans across the image, it is said to be *space-invariant*.

Fourier convolution can be used to implement space-invariant filtering by transforming the image and filter kernel into the frequency domain using an FFT, multiplying them together, and then computing the inverse FFT. For wide space-invariant kernels, this becomes the method of choice since it requires $O(N \log_2 N)$ operations instead of $O(MN)$ operations for direct convolution, where $M$ and $N$ are the lengths of the filter kernel and image, respectively. Since the cost of Fourier convolution is independent of the kernel width, it becomes practical when $M > \log_2 N$. This means, for example, that scaling an image can best be done in the frequency domain when excessive magnification or minification is desired. An excellent tutorial on the theory supporting digital filtering in the frequency domain can be found in [Smith 83]. The reader should note that the term "excessive" is taken to mean any scale factor beyond the processing power of fast hardware convolvers. For instance, current advances in pipelined hardware make direct convolution reasonable for filter neighborhoods as large as 17 x 17.

5.1.4. **Space-Variant Filtering**

In most image warping applications, however, *space-variant* filters are required, where the kernel varies with position. This is necessary for many common operations such as perspective mappings, nonlinear warps, and texture mapping. In such cases, space-variant FIR filters are used to convolve the preimage. Proper filtering requires a large number of preimage samples in order to compute each output pixel. There are various sampling strategies used to collect these samples. They can be broadly categorized into two classes: regular sampling and irregular sampling.

5.2. **REGULAR SAMPLING**

The process of using a regular sampling grid to collect image samples is called *regular sampling*. It is also known as *uniform sampling*, which is slightly misleading since an irregular sampling grid can also generate a uniform distribution of samples. Regular sampling includes point sampling, as well as the supersampling and adaptive sampling techniques described below.

5.2.1. **Supersampling**

The process of using more than one regularly-spaced sample per pixel is known as *supersampling*. Each output pixel value is evaluated by computing a weighted average of the samples taken from their respective preimages. For example, if the supersampling grid is three times denser than the output grid (i.e., there are nine grid points per pixel area), each output pixel will be an average of the nine samples taken from its projection in the input image. If, say, three samples hit a green object and the remaining six samples hit a blue object, the composite color in the output pixel will be one-third green and two-thirds blue, assuming a box filter is used.
Digitales Video

Supersampling reduces aliasing by bandlimiting the input signal. The purpose of the high-resolution supersampling grid is to refine the estimate of the preimages seen by the output pixels. The samples then enter the prefiltering stage, consisting of a low-pass filter. This permits the input to be resampled onto the (relatively) low-resolution output grid without any offending high frequencies introducing aliasing artifacts. In Figure 5.5 we see an output pixel subdivided into nine subpixel samples which each undergo inverse mapping, sampling the input at nine positions. Those nine values then pass through a low-pass filter to be averaged into a single output value.

![Figure 5.5: Supersampling.](image)

The impact of supersampling is easily demonstrated in the following example of a checkerboard projected onto an oblique plane. Figure 5.6 shows four different sampling rates used to perform an inverse mapping. In Figure 5.6a, only one checkerboard sample per output pixel is used. This contributes to the jagged edges at the bottom of the image and to the moire patterns at the top. They directly correspond to poor reconstruction and antialiasing, respectively. The results are progressively refined as more samples are used to compute each output pixel.

There are two problems associated with straightforward supersampling. The first problem is that the newly designated high frequency of the prefiltered image continues to be fixed. Therefore, there will always be sufficiently higher frequencies that will alias. The second problem is cost. In our example, supersampling will take nine times longer than point sampling. Although there is a clear need for the additional computation, the dense placement of samples can be optimized. Adaptive supersampling is introduced to address these drawbacks.

### 5.2.2. Adaptive Supersampling

In *adaptive supersampling*, the samples are distributed more densely in areas of high intensity variance. In this manner, supersamples are collected only in regions that warrant their use. Early work in adaptive supersampling for computer graphics is described in [Whitted 80]. The strategy is to subdivide areas between previous samples when an edge, or some other high frequency pattern, is present. Two approaches to adaptive supersampling have been described in the literature. The first approach allows sampling density to vary as a function of local image variance [Lee 85, Kajiya 86]. A second approach introduces two levels of sampling densities: a regular pattern for most areas and a higher-density pattern for regions demonstrating high frequencies. The regular pattern simply consists of one sample per output pixel. The high density pattern involves local supersampling at a rate of 4 to 16 samples per pixel. Typically, these rates are adequate for suppressing aliasing artifacts.
A strategy is required to determine where supersampling is necessary. In [Mitchell 87], the author describes a method in which the image is divided into small square supersampling cells, each containing eight or nine of the low-density samples. The entire cell is supersampled if its samples exhibit excessive variation. In [Lee 85] the variance of the samples is used to indicate high frequency. It is well known, however, that variance is a poor measure of visual perception of local variation. Another alternative is to use contrast, which more closely models the nonlinear response of the human eye to rapid fluctuations in light intensities [Caelli 81]. Contrast is given as

\[ C = \frac{I_{\text{max}} - I_{\text{min}}}{I_{\text{max}} + I_{\text{min}}} \]  

**Figure 5.6:** Supersampling an oblique checkerboard. (a) 1, (b) 4, (c) 16, and (d) 256 samples per output pixel. Images have been enlarged with pixel replication.
Digitales Video

Adaptive sampling reduces the number of samples required for a given image quality. The problem with this technique, however, is that the variance measurement is itself based on point samples, and so this method can fail as well. This is particularly true for sub-pixel objects that do not cross pixel boundaries. Nevertheless, adaptive sampling presents a far more reliable and cost-effective alternative to supersampling.

An example of the effectiveness of adaptive supersampling is shown in Fig. 6.8. The image, depicting a bowl on a wooden table, is a computer-generated picture that made use of bilinear interpolation for reconstruction and box filtering for antialiasing. Higher sampling rates were chosen in regions of high variance. For each output pixel, the following operations were taken. First, the four pixel corners were projected into the input. The average of these point samples was computed. If any of the corner values differed from that average by more than some user-specified threshold, then the output pixel was subdivided into four subpixels. The process repeats until the four corners satisfy the uniformity condition. Each output pixel is the average of all the computed input values that map onto it.

5.2.3. Reconstruction from Regular Samples

Each output pixel is evaluated as a linear combination of the preimage samples. The low-pass filters shown in Figure 5.3 and Figure 5.5 are actually reconstruction filters used to interpolate the output point. They share the identical function of the reconstruction filters discussed in Chapter 5: they bandlimit the sampled signal (suppress the replicated spectra) so that the resampling process does not itself introduce aliasing. The careful reader will notice that reconstruction serves two roles:

Figure 5.7: A ray-traced image using adaptive supersampling.

Reconstruction filters interpolate the input samples to compute values at nonintegral positions. These values are the preimage samples that are assigned to the supersampling grid.
The very same filters are used to interpolate a new value from the dense set of samples collected in step (1). The result is applied to the output pixel.

When reconstruction filters are applied to interpolate new values from regularly spaced samples, errors may appear as observable derivative discontinuities across pixel boundaries. In antialiasing, reconstruction errors are more subtle. Consider an object of constant intensity which is entirely embedded in pixel \( p \), i.e., a sub-pixel sized object. We will assume that the popular triangle filter is used as the reconstruction kernel. As the object moves away from the center of \( p \), the computed intensity for \( p \) decreases as it moves towards the edge. Upon crossing the pixel boundary, the object begins to contribute to the adjacent pixel, no longer having an influence on \( p \). If this motion were animated, the object would appear to flicker as it crossed the image. This artifact is due to the limited range of the filter. This suggests that a wider filter is required, in order to reflect the object's contribution to neighboring pixels.

One ad hoc solution is to use a square pyramid with a base width of 2x2 pixels. This approach was used in [Blinn 76,] an early paper on texture mapping. In general, by varying the width of the filter a compromise is reached between passband transmission and stopband attenuation. This underscores the need for high-quality reconstruction filters to prevent aliasing in image resampling.

Despite the apparent benefits of supersampling and adaptive sampling, all regular sampling methods share a common problem: information is discarded in a coherent way. This produces coherent aliasing artifacts that are easily perceived. Since spatially correlated errors are a consequence of the regularity of the sampling grid, the use of irregular sampling grids has been proposed to address this problem.

5.3. DIRECT CONVOLUTION

Whether regular or irregular sampling is used, direct convolution requires fast space-variant filtering. Most of the work in antialiasing research has focused on this problem. They have generally addressed approximations to the convolution integral of Eq. 5.1.

In the general case, a preimage can be of arbitrary shape and the kernel can be an arbitrary filter. Solutions to this problem have typically achieved performance gains by adding constraints. For example, most methods approximate a curvilinear preimage by a quadrilateral. In this manner, techniques discussed in Chapter 3 can be used to locate points in the preimage. Furthermore, simple kernels are often used for computational efficiency. For consistency with the texture mapping literature from which they are derived, we shall refer to the input and output coordinate systems as texture space and screen space, respectively.

5.4. PREFILTERING

The direct convolution methods impose minimal constraints on the filter area (quadrilateral, ellipse) and filter kernel (precomputed lookup table entries). Additional speedups are possible if further constraints are imposed. Pyramids and preintegrated tables are introduced to approximate the convolution integral with a constant number of accesses. This compares favorably against direct convolution which requires a large number of samples that grow proportionately to preimage area. As we shall see, though, the filter area will be limited to
squares or rectangles, and the kernel will consist of a box filter. Subsequent advances have extended their use to more general cases with only marginal increases in cost.

5.4.1. Pyramids

Pyramids are multi-resolution data structures commonly used in image processing and computer vision. They are generated by successively bandlimiting and subsampling the original image to form a hierarchy of images at ever decreasing resolutions. The original image serves as the base of the pyramid, and its coarsest version resides at the apex. Thus, in a lower resolution version of the input, each pixel represents the average of some number of pixels in the higher resolution version.

The resolution of successive levels typically differs by a power of two. This means that successively coarser versions each have one quarter of the total number of pixels as their adjacent predecessors. The memory cost of this organization is modest: $1 + 1/4 + 1/16 + \cdots = 4/3$ times that needed for the original input. This requires only 33% more memory.

To filter a preimage, one of the pyramid levels is selected based on the size of its bounding square box. That level is then point sampled and assigned to the respective output pixel. The primary benefit of this approach is that the cost of the filter is constant, requiring the same number of pixel accesses independent of the filter size. This performance gain is the result of the filtering that took place while creating the pyramid. Furthermore, if preimage areas are adequately approximated by squares, the direct convolution methods amount to point sampling a pyramid. This approach was first applied to texture mapping in [Catmull 74] and described in [Dungan 78].

There are several problems with the use of pyramids. First, the appropriate pyramid level must be selected. A coarse level may yield excessive blur while the adjacent finer level may be responsible for aliasing due to insufficient bandlimiting. Second, preimages are constrained to be squares. This proves to be a crude approximation for elongated preimages. For example, when a surface is viewed obliquely the texture may be compressed along one dimension. Using the largest bounding square will include the contributions of many extraneous samples and result in excessive blur. These two issues were addressed in [Williams 83] and [Crow 84], respectively, along with extensions proposed by other researchers.

Williams proposed a pyramid organization called *mip map* to store color images at multiple resolutions in a convenient memory organization [Williams 83]. The acronym "mip" stands for "multum in parvo," a Latin phrase meaning "many things in a small place." The scheme supports trilinear interpolation, where both intra- and inter-level interpolation can be computed using three normalized coordinates: $u$, $v$, and $d$. Both $u$ and $v$ are spatial coordinates used to access points within a pyramid level. The $d$ coordinate is used to index, and interpolate between, different levels of the pyramid. This is depicted in Figure 5.8.
The quadrants touching the east and south borders contain the original red, green, and blue (RGB) components of the color image. The remaining upper-left quadrant contains all the lower resolution copies of the original. The memory organization depicted in Figure 5.8 clearly supports the earlier claim that memory cost is $4/3$ times that required for the original input. Each level is shown indexed by the $[u,v,d]$ coordinate system, where $d$ is shown slicing through the pyramid levels. Since corresponding points in different pyramid levels have indices which are related by some power of two, simple binary shifts can be used to access these points across the multi-resolution copies. This is a particularly attractive feature for hardware implementation.

The primary difference between mip maps and ordinary pyramids is the trilinear interpolation scheme possible with the $[u,v,d]$ coordinate system. By allowing a continuum of points to be accessed, mip maps are referred to as pyramidal parametric data structures. In Williams’ implementation, a box filter (Fourier window) was used to create the mip maps, and a triangle filter (Bartlett window) was used to perform intra- and inter-level interpolation. The value of $d$ must be chosen to balance the tradeoff between aliasing and blurring. Heckbert suggests

$$d^2 = \text{MAX}\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 \cdot \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2\right)$$

where $d$ is proportional to the span of the preimage area, and the partial derivatives can be computed from the surface projection [Heckbert 83].

### 5.4.2. Summed-Area Tables

An alternative to pyramidal filtering was proposed by Crow in [Crow 84]. It extends the filtering possible in pyramids by allowing rectangular areas, oriented parallel to the coordinate axes, to be filtered in constant time. The central data structure is a preintegrated buffer of intensities, known as the **summed-area table**. This table is generated by computing a running

Anti-Aliasing
total of the input intensities as the image is scanned along successive scanlines. For every position \( P \) in the table, we compute the sum of intensities of pixels contained in the rectangle between the origin and \( P \). The sum of all intensities in any rectangular area of the input may easily be recovered by computing a sum and two differences of values taken from the table. For example, consider the rectangles \( R_0 \), \( R_1 \), \( R_2 \), and \( R \) shown in Fig. 6.9. The sum of intensities in rectangle \( R \) can be computed by considering the sum at \([x_1, y_1]\), and discarding the sums of rectangles \( R_0 \), \( R_1 \), and \( R_2 \). This corresponds to removing all area lying below and to the left of \( R \). The resulting area is rectangle \( R \), and its sum \( S \) is given as

\[
S = T[x_1, y_1] - T[x_1, y_0] - T[x_0, y_1] + T[x_0, y_0]
\]

where \( T[x, y] \) is the value in the summed-area table indexed by coordinate pair \([x, y]\).  

\[\text{Figure 5.9 Summed-area table calculation.}\]

Since \( T[x_1, y_0] \) and \( T[x_0, y_1] \) both contain \( R_0 \), the sum of \( R_0 \) was subtracted twice in Eq. 5.4. As a result, \( T[x_0, y_0] \) was added back to restore the sum. Once \( S \) is determined it is divided by the area of the rectangle. This gives the average intensity over the rectangle, a process equivalent to filtering with a Fourier window (box filtering).

There are two problems with the use of summed-area tables. First, the filter area is restricted to rectangles. This is addressed in [Glassner 86], where an adaptive, iterative technique is proposed for obtaining arbitrary filter areas by removing extraneous regions from the rectangular bounding box. Second, the summed-area table is restricted to box filtering. This, of course, is attributed to the use of unweighted averages that keeps the algorithm simple. In [Perlin 85] and [Heckbert 86a], the summed-area table is generalized to support more sophisticated filtering by repeated integration.

It is shown that by repeatedly integrating the summed-area table \( n \) times, it is possible to convolve an orthogonally oriented rectangular region with an \( n^{th} \)-order box filter (B-spline). Kernels for small \( n \) are shown in Fig. 5.10. The output value is computed by using \((n + 1)^2\) weighted samples from the preintegrated table. Since this result is independent of the size of the rectangular region, this method offers a great reduction in computation over that of direct convolution. Perlin called this a selective image filter because it allows each sample to be blurred by different amounts.

Repeated integration has rather high memory costs relative to pyramids. This is due to the number of bits necessary to retain accuracy in the large summations. Nevertheless, it allows us to filter rectangular or elliptical regions, rather than just squares as in pyramid techniques.
Digitales Video

Since pyramid and summed-area tables both require a setup time, they are best suited for input that is intended to be used repeatedly, i.e., a stationary background scene. In this manner, the initialization overhead can be amortized over each use. However, if the texture is only to be used once, the direct convolution methods raise a challenge to the cost-effectiveness offered by pyramids and summed-area tables.

5.5. DISCUSSION

This chapter has reviewed methods to combat the aliasing artifacts that may surface upon performing geometric transformations on digital images. Aliasing becomes apparent when the mapping of input pixels onto the output is many-to-one. Sampling theory suggests theoretical limitations and provides insight into the solution. In the majority of cases, increasing display resolution is not a parameter that the user is free to adjust. Consequently, the approaches have dealt with bandlimiting the input so that it may conform to the available output resolution.

By precomputing pyramids and summed-area tables, filtering is possible with only a constant number of computations, independent of the preimage area. Combining the partially filtered results contained in these data structures produces large performance gains. The cost, however, is in terms of constraints on the filter kernel and approximations to the preimage area. Designing efficient filtering techniques that support arbitrary preimage areas and filter kernels remains a great challenge. It is a subject that will continue to receive much attention.

References: