Discrepancy as a Quality Measure for Sample Distributions

Proceedings of Eurographics 91, June, 1991, pp183-193

Peter Shirley *
Department of Computer Science
Indiana University

Abstract

Discrepancy, a scalar measure of sample point equidistribution, is discussed in the context of Computer Graphics sampling problems. Several sampling strategies and their discrepancy characteristics are examined. The relationship between image error and the discrepancy of the sampling patterns used to generate the imaqge is established. The definition of discrepancy is extended to non-uniform sampling patterns.

CR Categories and Subject Descriptors: I.3.0 [Computer Graphics]: General; I.3.6 [Computer Graphics]: Methodology and Techniques.

Additional Key Words and Phrases: Sampling and Filtering, Monte Carlo Integration, Ray Tracing.

1 Introduction

One way we can calculate a computer image is to first find the color at many points on the image plane, and then somehow reconstruct a continuous image function from the colors at those points [10]. The rendering problem can also be viewed as a sampling problem in several one and two-dimensional spaces [2]. This sampling can be static [2], where all the sample points are known in advance, or adaptive [9, 4, 6, 10, 11], where new sample points are chosen in 'interesting' areas.

If static sampling is used, or adaptive sampling as done by Lee et al. [9], then we need a module to produce 'good' sets of N samples on a pixel, where N is a positive integer. Typically, N is between 4 and 25 for static sampling, and as high as 400 for adaptive sampling. This module can be viewed as a black box (abstract data type) that is independent from the rest of any rendering code that uses the samples.

An obvious question we should ask when deciding how to design the internals of the sample generation black box is, 'how do we test whether the sample points produced are good ones?'. This question has been discussed using the concepts of Monte Carlo integration [2], but this discussion did not provide a good quality measure for a set of sample points. Signal processing theory has also been used to examine different sampling strategies [1, 4], but this approach only provides quantitative information in the form of two-dimensional frequency distributions which are hard to interpret.

Our question, 'what are good sample point sets?', has been examined in depth in the field of numerical integration [15, 14]. The approach used in integration theory is to test whether a set of points is *equidistributed* by calculating the *discrepancy* of the set. This approach has the advantage of assigning a single quality number, the discrepancy, to every point set. This will allow us to order sample point sets by discrepancy, and say which one is 'best'. The basic idea of discrepancy is obvious enough that it has probably been invented and reinvented several times. For eaxmple, the idea, as well as the generalized discrepancy presented in Section 4, can be easily extracted from the Kolmogrov-Smirnov statistical test [8].

In this paper, discrepancy is used for evaluation in to computer graphics sampling problems. It is tested on several test images and sampling strategies. These tests are preliminary, and are not intended to imply that any of these sampling strategies is best. They are instead meant to show that the discrepancy measure is useful for quantitative evaluation of sampling strategies used for computer graphics.

In Section 2, several standard sampling strategies used in computer graphics are reviewed. In Section 3, these strategies are applied to four test images. In Section 3.1, the discrepancy measure is defined, and numerically applied to the sample distributions. These results are compared to the empirical behavior of the sampling strategies for the test images. In Section 4, the definition of discrepancy is generalized to include measurement of non-uniform sample point sets. The results are summarized in Section 5.

2 Sample Point Generation

The basic task in sample point generation is, given some desired number of samples, N, to find N sample points on the unit square. There are several ways to find these points that have been used in computer graphics. In this section,

*Department of Computer Science, Lindley Hall, Indiana University, Bloomington, IN 47405, USA. Email: shirley@cs.indiana.edu.

several of the most common sample generation techniques are discussed.

2.1 Random Sampling

The simplest way to choose N points (x_i, y_i) from the unit square is to pick every x_i and y_i independently by setting them to canonical random pairs (ξ_i, ξ_i'). A canonical random number is uniformly distributed on $[0, 1]$ [5]. This strategy, called *random* sampling, or sometimes *poisson* sampling, can be summarized as:

for $i = 0$ to $N - 1$
  $x_i = randfrom(0, 1)$
  $y_i = randfrom(0, 1)$

where $randfrom(a, b)$ returns a random number uniformly distributed on the interval $[a, b]$.

2.2 Regular Sampling

We could simply place the pairs evenly in a grid pattern (as is done in traditional quadrature methods). This will prevent clumping of samples, but may introduce spatial regularity into the error across many pixels. This *regular* sampling strategy can be summarized as:

for $i = 0$ to $n_x - 1$
  for $j = 0$ to $n_y - 1$
    $k = n_x i + j$
    $x_k = (i + 0.5)/n_x$
    $y_k = (j + 0.5)/n_y$

where n_x and n_y are chosen so that $n_x n_y = N$. This is often made easier by restricting N to be a perfect square.

2.3 Jittered Sampling

Jittered sampling is another name for classical Monte Carlo stratified sampling [2]. Typically, the unit square is partitioned into a set of equal area rectangles, and a point is chosen uniformly from each rectangle:

for $i = 0$ to $n_x - 1$
  for $j = 0$ to $n_y - 1$
    $k = in_x + j$
    $x_k = randfrom(i/n_x, (i + 1)/n_x)$
    $y_k = randfrom(j/n_y, (j + 1)/n_y)$

where, as in regular sampling, $n_x n_y = N$. There are other stratifications that could be used, such as the hexagonal grid suggested by Dippe and Wold [4], but the rectangular pattern is the easiest to implement.

2.4 Half-jittered Sampling

One problem with jittered sampling is that limited clumping can take place (up to four sample points could coincide). This problem can be lessened by choosing points nearer to the center of each square. Cook did this by using a truncated Gaussian distribution to choose from each square. A simpler (and probably worse) method is to choose samples that are *half-jittered*, where points are chosen uniformly within a rectangle, half the width of the full rectangle:

for $i = 0$ to $n_x - 1$
  for $j = 0$ to $n_y - 1$
    $k = in_x + j$
    $x_k = randfrom((i + 0.25)/n_x, (i + 0.75)/n_x)$
    $y_k = randfrom((j + 0.25)/n_y, (j + 0.75)/n_y)$

In half-jittered sample sets, there is less clumping, but more regularity than when simple jittered sampling is used.

2.5 Poisson Disk Sampling

A simple way to avoid clumping of sample points is to generate a sequence of samples, and reject a new sample if it is too close to an existing sample [4, 1]. This method, called Poisson disk sampling (also minimum-separation Poisson sampling), can be summarized as:

$i = 0$
while $i < N$
  $x_i = randfrom(0, 1)$
  $y_i = randfrom(0, 1)$
  $reject$ = **false**
  for $j = 0$ to $i - 1$
    if $[(x_i - x_j)^2 + (y_i - y_j)^2] < d^2$
      $reject$ = **true**
      break
  if not reject
    $i = i + 1$

where d is the minimum allowed separation (Poisson disk diameter). There is less clumping for large minimum separation, but too large a separation can cause ghosting or even make sampling impossible.

2.6 N-rooks Sampling

Cook et al.[2] generated sample distributions for a nine-dimensional space by separately partitioning four two-

dimensional subspaces (pixel area, camera lens area, reflection direction, shadow ray direction), and one one-dimensional subspace (time), and then associating the strata of each dimension at random. He called this technique uncorrelated jittering. This basic strategy has also been discussed in the book by Kalos [7].

This idea can be applied in two dimensions by randomly associating rows and columns of a N by N grid, so that each row and column will have one sample[13]. A particularly descriptive name for this strategy is *N-rooks sampling*, because an acceptable set of sample cells will be the squares of an N by N chessboard with N rooks that cannot capture each other in one move. The strategy can be summarized as follows:

for $i = 0$ to N
    $x_i = randfrom(i/N, (i+1)/N)$
    $y_i = randfrom(i/N, (i+1)/N)$
permute values in x_1 through x_N

where **permute** shuffles a group of values into a random order. There are two advantages of this type of sampling over conventional jitter. The first is that any number of samples can be taken with uncorrelated jitter, while conventional jitter performs poorly when N is prime. The second is that if the sampled field varies only in one dimension, that dimension is fully stratified.

3 Empirical Behavior of Sampling Strategies

Each of the sampling methods described in the last section was tested with 16 samples per pixel on each of four 128 by 96 pixel test images. The first image, called **checker** (Figure 1), is that of an infinite checkerboard. Since the uncorrelated jitter is well suited to images with predominantly horizontal and vertical lines, the second test figure, **checker-45** (Figure 2), is the same checkboard rotated 45° to avoid such lines. The third figure, **ball** (Figure 3), is a ball lit by two square light sources. This tests the sampling of the shadow ray directions. The final figure, **all** (Figure 4), has a ball, a perfect mirror, an imperfect mirror, and a finite aperture lens. All four of the figures shown were sampled with 400 jittered samples, which is sufficient to produce an image with relatively small error.

Some of the sampling methods produced regular errors that were visually disturbing. On the checkerboards this was particularly true; the regular and half-jittered sampling had visible aliasing artifacts. Figure 5 shows the regular sampling of **checker-45**, where the banding near the horizon is not present in the same image in the separately sampled image shown in Figure 6. This exemplifies the situation where it is the regularity of the error, rather than its magnitude, that is objectionable.

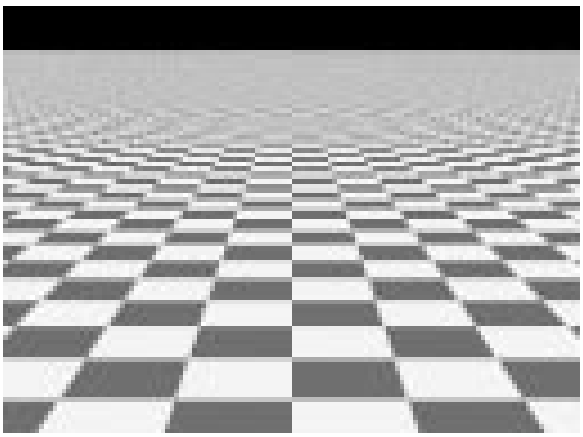

Figure 1: Test figure **checker**

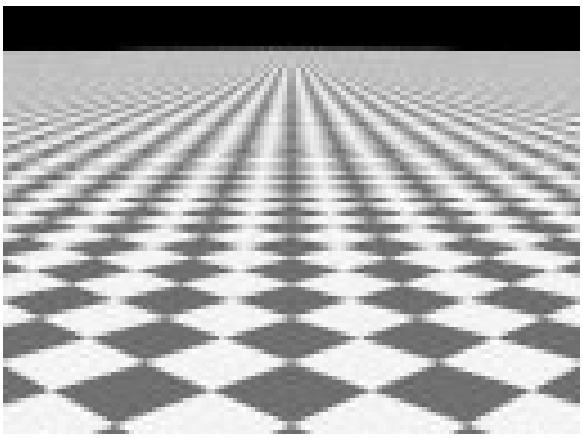

Figure 2: Test figure **checker-45**

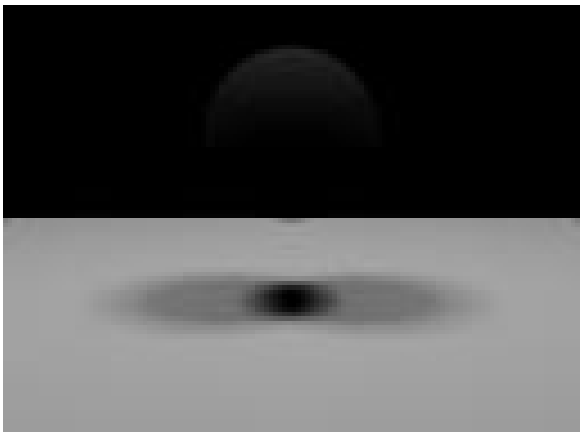

Figure 3: Test figure **ball**

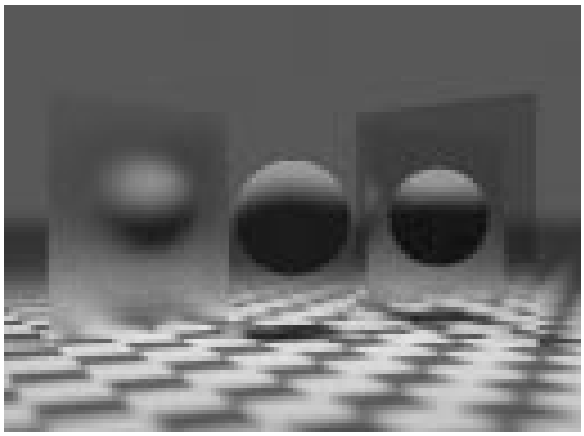

Figure 4: Test figure **all**

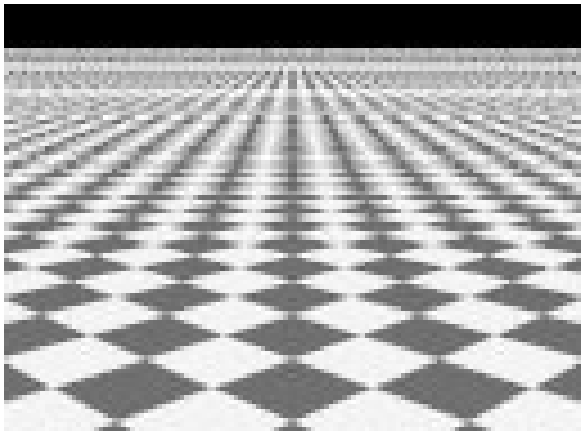

Figure 5: Test figure **checker-45** with 16 regular samples per pixel

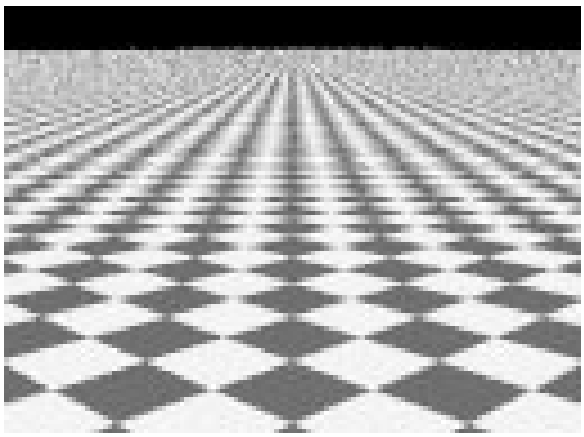

Figure 6: Test figure **checker-45** with 16 separate samples per pixel

sampling method	*ave(|E|)*	*SD(|E|)*	*max(|E|)*
separate	0.0163	0.0303	0.308
jittered	0.0216	0.0368	0.394
poisson ($d = 0.2$)	0.0221	0.0362	0.334
poisson ($d = 0.1$)	0.0259	0.0413	0.308
half-jittered	0.0263	0.0437	0.368
random	0.0303	0.0479	0.331
regular	0.0312	0.0526	0.390

Table 1: Pixel errors in luminance for **checker**

sampling method	*ave(|E|)*	*SD(|E|)*	*max(|E|)*
separate	0.0190	0.0296	0.286
jittered	0.0221	0.0340	0.291
poisson ($d = 0.2$)	0.0225	0.0343	0.333
poisson ($d = 0.1$)	0.0281	0.0416	0.304
half-jittered	0.0237	0.0445	0.410
regular	0.0273	0.0532	0.450
random	0.0315	0.0466	0.294

Table 2: Pixel errors in luminance for **checker-45**

The average absolute error in luminance for each sampling strategy is listed for each test image in Tables 1 through 4. Surprisingly, the separate (uncorrelated jitter) sampling performed best by the average error metric on three of the four images, and on test image **all** was only outperformed by regular and half-jittered strategies, both of which are prone to aliasing. The standard deviation and maximum error caused by separate sampling are also low relative to the other sampling strategies.

3.1 Discrepancy of Sampling Strategies

In the last section it was demonstrated that for some images the separate sampling strategy performs quite well, even compared to Poisson disk sampling. Overall, the performance rankings of the strategies (from best to worst) was approximately separate, jittered, half-jittered, regular, Poisson disk, and finally random.

The poor performance of Poisson disk goes against conventional wisdom. It would be nice to establish a quantitative metric for predicting the value of a particular strategy. It would also be a good idea to understand what we have done

sampling method	*ave(|E|)*	*SD(|E|)*	*max(|E|)*
separate	0.00324	0.0099	0.179
regular	0.00363	0.0113	0.177
half-jittered	0.00365	0.0114	0.151
jittered	0.00370	0.0110	0.160
poisson ($d = 0.2$)	0.00404	0.0123	0.170
poisson ($d = 0.1$)	0.00526	0.0180	0.226
random	0.00607	0.0214	0.266

Table 3: Pixel errors in luminance for **ball**

sampling method	*ave(|E|)*	*SD(|E|)*	*max(|E|)*
regular	0.0137	0.0235	0.242
half-jittered	0.0139	0.0234	0.205
separate	0.0148	0.0246	0.245
jittered	0.0150	0.0251	0.259
poisson ($d = 0.2$)	0.0156	0.0256	0.247
poisson ($d = 0.1$)	0.0172	0.0284	0.236
random	0.0190	0.0315	0.287

Table 4: Pixel errors in luminance for **all**

sampling method	*ave(D)*	*SD(D)*	*max(D)*
separate	0.162	0.0237	0.229
half-jittered	0.184	0.0187	0.243
jittered	0.193	0.0288	0.291
poisson ($d = 0.2$)	0.196	0.0332	0.290
regular	0.234	0.0000	0.234
poisson ($d = 0.1$)	0.245	0.0447	0.357
random	0.282	0.0557	0.428

Table 5: Discrepancies of different sampling strategies

by using sample sets that are not strictly random, since presumably we are doing a Monte Carlo integration. In numerical integration theory these non-random sample sets are called *quasi-random*, because they have some statistical qualities that make them acceptable substitutes for true random samples. Zeremba developed a theory to bound the error of an integration based on equidistribution properties of the sample set (assuming certain continuity properties of the integrand) [15]. In this section, Zeremba's equidistribution metric, *discrepancy*, is discussed in the context of the sampling strategies from the last section.

What discrepancy provides us is a number that is small for very equidistributed sample sets, and large for poorly distributed sets. Imagine a set of N sample points, (x_i, y_i) on the unit square. Given a point (a, b) on the square, the set of points (x, y) such that $x < a$ and $y < b$ will define a smaller square (with lower left corner $(0, 0)$ and upper right corner (a, b)) with area ab. Let n be the number of the N sample points that falls within that smaller square. If the sample points are reasonably equidistributed, we would expect n/N to be about ab. Zeremba uses this observation to define the

sampling method	*ave(S)*	*SD(S)*	*max(S)*
half-jittered	0.0463	0.00290	0.0537
separate	0.0467	0.00847	0.0812
jittered	0.0495	0.00192	0.0678
poisson ($d = 0.2$)	0.0540	0.00891	0.0844
regular	0.0600	0.00000	0.0600
poisson ($d = 0.1$)	0.0743	0.02140	0.1740
random	0.0877	0.02390	0.2080

Table 6: Root mean square discrepancies of different sampling strategies

sampling method	*ave(T)*	*SD(T)*	*max(T)*
separate	0.2132	0.0236	0.275
jittered	0.2555	0.0397	0.428
poisson ($d = 0.2$)	0.2613	0.0459	0.390
half-jittered	0.2608	0.0282	0.338
poisson ($d = 0.1$)	0.2921	0.0503	0.513
random	0.3434	0.0540	0.485
regular	0.3600	0.0000	0.360

Table 7: Stroud's discrepancies of different sampling strategies

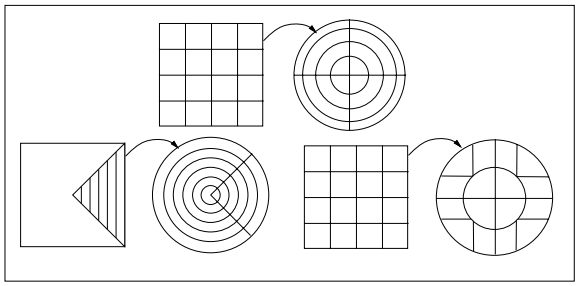

Figure 7: Two different ways to partition a circle

discrepancy, D, as the lowest upper bound of $|n/N - ab|$ for all (a, b). The average discrepancies of 100 sets of 16 samples for the various strategies are shown in Table 5. The table shows that the discrepancy of the separate samples is lowest, and the other sampling strategies are ranked in an order reasonably consistent with empirical behavior.

Zeremba points out that instead of taking the lowest upper bound of $|n/N - ab|$, we could take its root mean square value. The root mean square discrepancy, S, is shown in Table 6. Under this metric, the half-jittered sampling strategy slightly outperforms separate sampling. Again, the ordering is reasonably consistent with the observed behavior of the strategies.

Stroud has a slightly different definition of discrepancy: the discrepancy, T, is the lowest upper bound of $|n/N - (a - c)(b - d)|$, where c and d are the lower corner of the square, and n is the number of points within the square [14]. In other words, all squares are used, rather than just squares with one corner at the origin. This makes the discrepancy of a 90° rotation of a set of points invariant. Stroud's discrepancy for 100 sets of samples is shown in Table 7. Applying this definition of discrepancy leads to an evaluation of sampling strategies that accords closely with the observable degree of error.

These tests indicate that discrepancy may be a useful tool in evaluating sampling strategies. One shortcoming of Zeremba's definition is that it assumes a square domain. As shown in the top of Figure 7, a straight transformation from a evenly partitioned square to polar coordinates on a circle can stretch the strata. This implies that a good discrepancy in the canonical probability space does not guarantee good

equidistribution in the actual domain (such as the lens area). A special purpose transformation is shown in the bottom of Figure 7. This keeps the strata from distorting too much. Unfortunately, the definition of discrepancy does not extend to non-square domains, so the only justification for preferring the bottom stratification of the circle is visual inspection and intuition.

Discrepancy can also be applied to a set of sample points in one dimension. We can take the discrepancy of just the x or y components of the points. A low discrepancy of the x components implies that the points will provide good results in pixels with vertical lines. This is why N-rooks sampling does so well with horizontal and vertical lines; both the x and y components are fully jittered.

4 Non-uniform Sample Distributions

If a non-uniform 'filter' is used to sample a pixel [3], we would like to take weighted averages of sample point colors, or to distribute the samples nonuniformly. The second strategy, often called *importance sampling* [1], is preferred if no samples are shared by adjacent pixels.

If we use importance sampling, we need to generate sample points that are distributed according to the filter function. One way to produce such non-uniform points is to generate uniformly distributed points, and then 'warp' them by applying a transformation to the sample points that changes their location.

As a simple example, suppose we want the trivial 'box' weighting function, which simply samples the vicinity of the pixel area uniformly ($w(x,y) = 1/d^2$ on the square of side d centered oround the pixel). The warping function would take N sample points generated by a strategy described in Section 2, and apply the mapping $x_i = x_i - 0.5(d-1)$, and $y_i = y_i - 0.5(d-1)$. This assumes we are using a coordinate system where the pixel center is the origin and the pixel width is 1.

In practice we will want non-uniform weighting functions. For example, suppose we have the width 2 weighting function:

$$w(x,y) = (1-|x|)\,(1-|y|) \tag{1}$$

If we just wanted to generate independent random points with density w, we can apply standard techniques to transform canonical random numbers into these points [8]. Since this particular w is separable, we could generate x according to $w(x) = (1-|x|)$, and y the same way. We define the distribution function, F, associated with w:

$$F(x) = \int_{-1}^{x} (1-|x'|)dx' = \frac{1}{2} + x - \frac{1}{2}x|x| \tag{2}$$

To get our desired independent (x,y) distributed according to w, we simply take canonical (ξ_1, ξ_2) and apply $x_i = F^{-1}(x_i)$, and $y_i = F^{-1}(y_i)$, where F^{-1} is the inverse function of F.

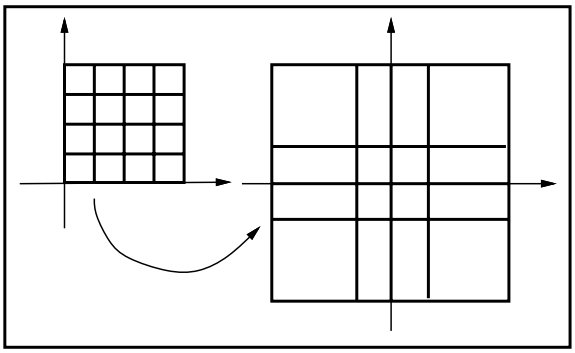

Figure 8: A transformation for nonuniform filter sampling.

This idea can be extended by taking our set of sample points and transforming them as if they were canonical. This will preserve some of the good qualities of the original sampling distribution, but will have our desired distribution properties. Applying this idea to the filter in Equation 1, we can take our n uniform sample points and apply the transformation:

$$x_i = \begin{cases} -1 + \sqrt{2x_i} & \text{if } x_i < 0.5 \\ 1 - \sqrt{2(1-x_i)} & \text{if } x_i \geq 0.5 \end{cases}$$

followed by a similar transformation involving y_i. This transformation on 16 jitter cells is shown in Figure 8. Note that this will transform nonuniformly within each cell, so the sample point selection must take place before the transformation.

The uniform sampling transformation can also apply to non-separable density functions, and to functions defined on non-Cartesian manifolds. This requires dealing with the joint distribution function and non-constant metrics, but otherwise the same techniques apply. Details can be found in [12, 13].

An immediate question is whether the concept of discrepancy can be extended to such non-uniform distributions. If we were generating N random points in some region R according to some probability density function $p(x)$, the expected number of points, n, within some $r \subset R$, is:

$$E(n) = N \int_{x \in r} p(x) d\mu(x)$$

where μ is the measure of R. We can make our new definition of discrepancy the lowest upper bound of $|n - E(n)|/N$ taken over some set of r_i where $r_i \subset R|$. Under this definition, we have Zeremba's when R is the unit square, p is uniform, and r_i is the set of all squares with the lower left hand corner of R. We have Stroud's if we allow the lower left hand corner to stray. We could generalize the root-mean-square in similar fashion.

In our filtering problem, we want points distributed according to the filter function. In our generalized discrepancy, p is this function. If we have a filter with a square support (non-zero region of R), then we can easily have the r_i

be all rectangles within the support. If p is separable (e.g. $(1-|x|)(1-|y|)$), and the transformation technique discussed earlier is used, then rectangles on the original unit square will map to new rectangles on R. The number of points in the original and transformed rectangle will be the same. This means the conventional discrepancy of the original sample point set will be the same as the generalized discrepancy of the transformed point set, so a 'good' uniform sample point set will transform into a 'good' nonuniform set.

The one-dimensional (projected vertical and horizontal) discrepancy could also be generalized. For any direction, we would expect the projections of points onto a line perpendicular to that direction to have a particular distribution. The generalized discrepancy for that projection would be an indication of how well the sample set would antialias edges with that orientation. A sample set with low discrepancy for many directions should be good at general images containing many edges.

5 Conclusion

The experiments in this paper indicate that discrepancy can be a useful metric for examining sampling issues in computer graphics. Discrepancy provides a single number that indicates something about the overall quality of a set of sample points.

Initial results using discrepancy indicate that N-rooks sampling may sometimes be the best simple strategy. Surprisingly, Poisson disk sampling was found to be one of the worst strategies in these experiments. The sensitivity of these results to disk radius and image characteristics should be explored more fully.

Future work should include investigation of which definition of discrepancy is best (perhaps disks would be better than rectangles), and whether there are any sample distributions with low projected discrepancies for several directions.

6 Acknowledgements

Thanks to William Brown, Jean Buckley, Dennis Gannon, Kelvin Sung, and William Kubitz (advisor for this project) for their help, reading, and encouragement.

References

[1] Robert L. Cook. Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, 5(1):51–72, January 1986.

[2] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed ray tracing. *Computer Graphics*, 18(4):165–174, July 1984. ACM Siggraph '84 Conference Proceedings.

[3] Franklin C. Crow. The aliasing problem in computer-generated shaded images. *Communications of the ACM*, 20(11):799–805, November 1977.

[4] Mark A. Z. Dippe and Erling Henry Wold. Antialiasing through stochastic sampling. *Computer Graphics*, 19(3):69–78, July 1985. ACM Siggraph '85 Conference Proceedings.

[5] John H. Halton. A retrospective and prospective of the monte carlo method. *SIAM Review*, 12(1):1–63, January 1970.

[6] James T. Kajiya. The rendering equation. *Computer Graphics*, 20(4):143–150, August 1986. ACM Siggraph '86 Conference Proceedings.

[7] Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods*. John Wiley and Sons, New York, N.Y., 1986.

[8] Donald Knuth. *The Art of Computer Programming, Volume 3*. Addison-Wesley, New York, N.Y., 1981.

[9] Mark E. Lee, Richard A. Redner, and Samuel P. Uselton. Statistically optimized sampling for distributed ray tracing. *Computer Graphics*, 19(3):61–68, July 1985. ACM Siggraph '85 Conference Proceedings.

[10] Don P. Mitchell. Generating antialiased images at low sampling densities. *Computer Graphics*, 21(4):65–72, July 1987. ACM Siggraph '87 Conference Proceedings.

[11] James Painter and Kenneth Sloan. Antialiased ray tracing by adaptive progressive refinement. *Computer Graphics*, 23(3):281–288, July 1989. ACM Siggraph '89 Conference Proceedings.

[12] Y. A. Screider. *The Monte Carlo Method*. Pergamon Press, New York, N.Y., 1966.

[13] Peter Shirley. *Physically Based Lighting Calculations for Computer Graphics*. PhD thesis, University of Illinois at Urbana-Champaign, November 1990.

[14] A. H. Stroud. *Approximate Calculation of Multiple Integrals*. Prentics-Hall, Englewood Cliffs, N. J., 1971.

[15] S. K. Zeremba. The mathematical basis of monte carlo and quasi-monte carlo methods. *SIAM Review*, 10(3):303–314, July 1968.