Fast Poisson Disk Sampling in Arbitrary Dimensions

Robert Bridson*
University of British Columbia

Abstract

In many applications in graphics, particularly rendering, generating samples from a blue noise distribution is important. However, existing efficient techniques do not easily generalize beyond two dimensions. Here I demonstrate a simple modification to dart throwing which permits generation of Poisson disk samples in $O(N)$ time, easily implemented in arbitrary dimension.

CR Categories: I.3.0 [Computer Graphics]: General

Keywords: sampling, blue noise, Poisson disk

1 Introduction

Blue noise sample patterns—for example produced by Poisson disk distributions, where all samples are at least distance r apart for some user-supplied density parameter r—are generally considered ideal for many applications in rendering (see Cook's landmark paper for example [1986]). Unfortunately the naïve rejection-based approach for generating Poisson disk samples, dart throwing, is impractically inefficient.

Many papers have overcome this inefficiency in two dimensions; I focus on one approach in particular, due to Dunbar and Humphreys [2006]. Their algorithm maintains a "scalloped sector" data structure which efficiently encodes the geometry of the region of the plane at distance between r and $2r$ from existing samples, and permits efficient uniform sampling from this region. Since every point in this region is an allowable sample, and since every maximal Poisson disk sampling containing the existing samples must also contain a point from this region, their algorithm can very efficiently generate the desired distribution.

However, many sampling applications use three or more dimensions: rendering with motion blur or depth-of-field, many particle systems for animation, etc. Existing two-dimensional fast blue noise samplers do not easily generalize to higher dimensions, thus practitioners tend to use either uniform random distributions (despite undesirable clustering), jittered/stratified sampling (which reduces but doesn't eliminate clustering), or more structured distributions which induce anisotropy.

In this sketch I present a new algorithm, easily implemented in arbitrary dimensions, that is guaranteed to take $O(N)$ time to generate N Poisson disk samples. Similar to the approach of Dunbar and Humphreys, sample candidates are drawn only from a region near existing samples, but instead of exactly computing the allowed region, rejection sampling is used to discover it.

2 The Algorithm

The algorithm takes as input the extent of the sample domain in $\mathbf{R}^n$, the minimum distance r between samples, and a constant k as the limit of samples to choose before rejection in the algorithm (typically $k = 30$).

*email: rbridson@cs.ubc.ca

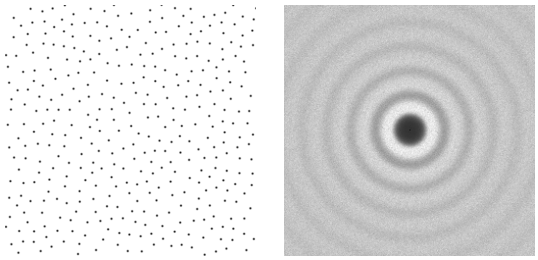

Figure 1: *Two-dimensional sample pattern from the algorithm and corresponding periodogram averaged over many runs.*

Step 0. Initialize an n-dimensional background grid for storing samples and accelerating spatial searches. We pick the cell size to be bounded by $r/\sqrt{n}$, so that each grid cell will contain at most one sample, and thus the grid can be implemented as a simple n-dimensional array of integers: the default -1 indicates no sample, a non-negative integer gives the index of the sample located in a cell.

Step 1. Select the initial sample, x_0, randomly chosen uniformly from the domain. Insert it into the background grid, and initialize the "active list" (an array of sample indices) with this index (zero).

Step 2. While the active list is not empty, choose a random index from it (say i). Generate up to k points chosen uniformly from the spherical annulus between radius r and $2r$ around x_i. For each point in turn, check if it is within distance r of existing samples (using the background grid to only test nearby samples). If a point is adequately far from existing samples, emit it as the next sample and add it to the active list. If after k attempts no such point is found, instead remove i from the active list.

3 Analysis

Step 2 is executed exactly $2N-1$ times to produce N samples: each iteration either produces a new sample and adds it to the active list, or removes an existing sample from the active list. Each iteration of step 2 takes $O(k)$ time, and since k is held constant (typically quite small) the algorithm is linear.

Figure 1 shows results in two dimensions; the accompanying material shows results in three dimensions. The accompanying prototype code illustrates a simple implementation which takes the dimension of the space as a parameter.

Acknowledgements

This work was in part supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

COOK, R. L. 1986. Stochastic sampling in computer graphics. *ACM Trans. Graph. 5*, 1.

DUNBAR, D., AND HUMPHREYS, G. 2006. A spatial data structure for fast poisson-disk sample generation. *ACM Trans. Graph. 25*, 3, 503–508.