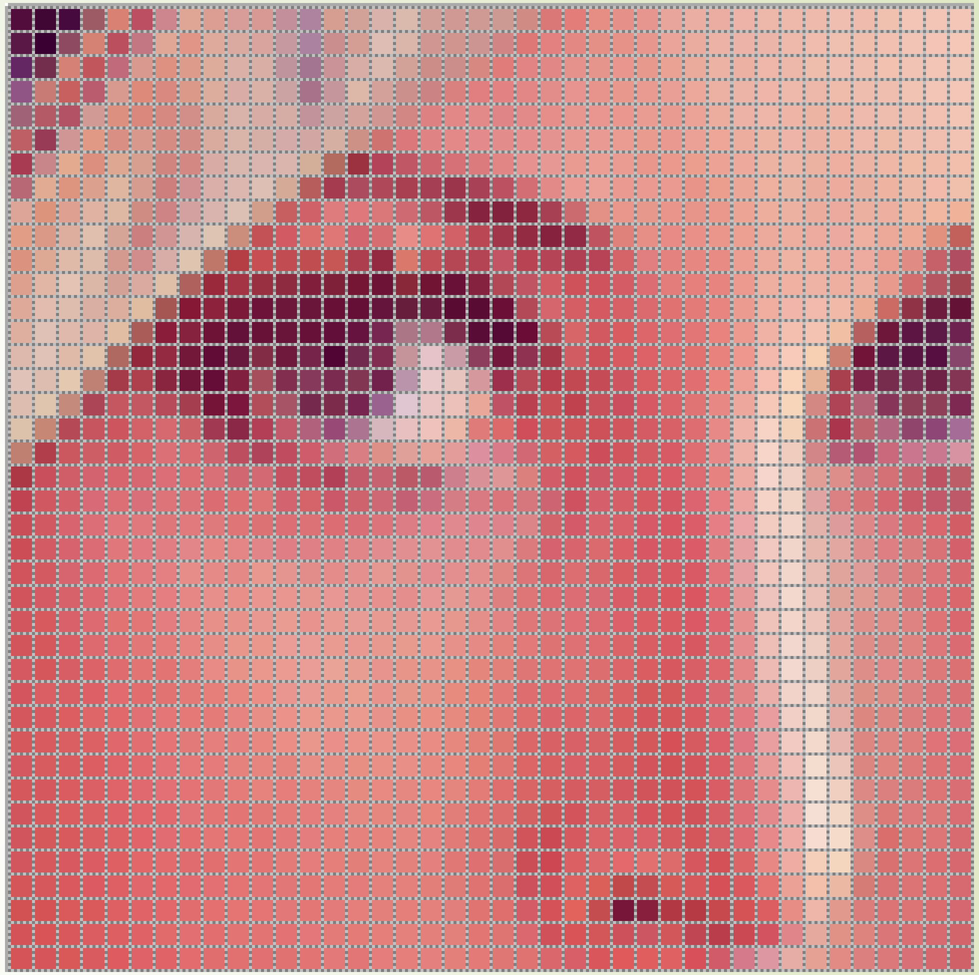
CS451
Ray Tracing
BVH and sampling

Jyh-Ming Lien
Department of Computer SCience
George Mason University

Most materials are from “Reaistic Ray Tracing” by Shirley and Morley

Review

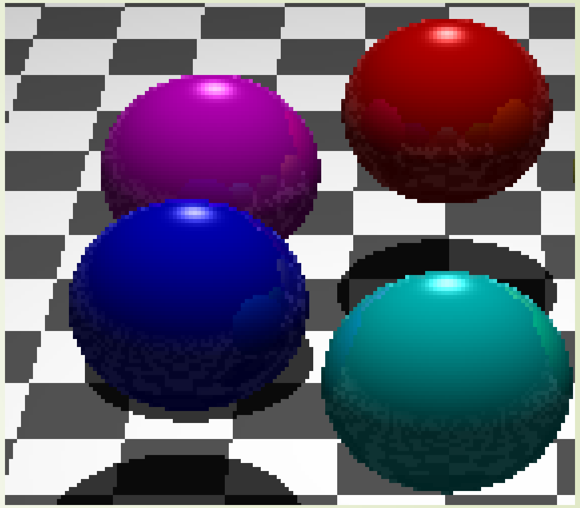
- What we learned last week
  - Ray tracing framework
  - Reflect
  - Refraction
  - Soft shadow, reflection, refraction
- Remember that a pixel defines an area, not a point, in an image



Today

- More on ray tracing
  - Bounding volume hierarchy
  - Sampling strategy
  - Monte Carlo ray tracing
- Why do we need so many different versions of ray tracers?

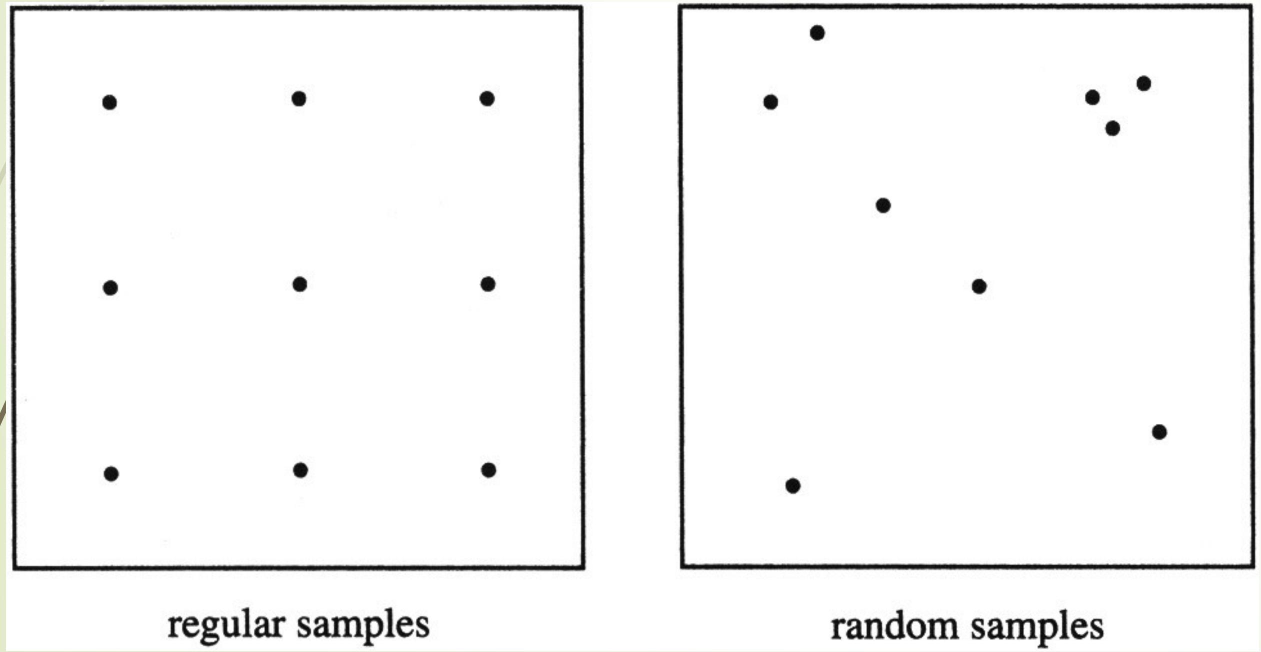
Sampling

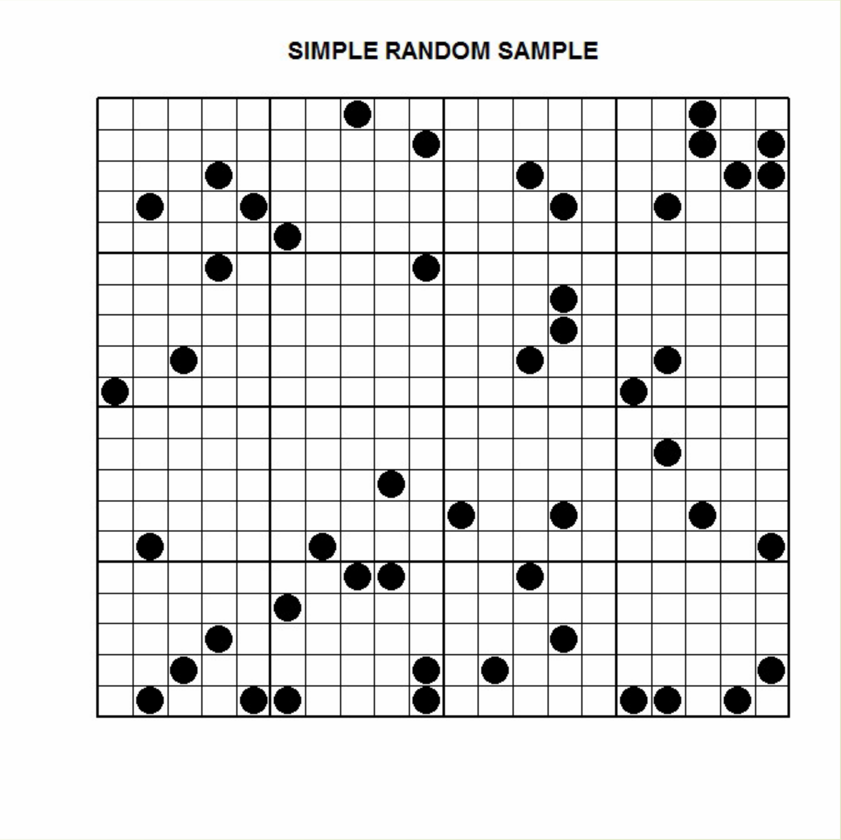


- Uniform distribution is good, BUT
- Correlation between samples is bad (jaggy rendering)
  - So, using points on a grid is bad (called Regular Sampling)
- Random Sampling (with uniform random number generator)

$$
\begin{aligned}
&\textbf{for } i = 0 \text{ to } N-1 \\
&\quad x_i = randfrom(0,1) \\
&\quad y_i = randfrom(0,1)
\end{aligned}
$$

Simple Random Sample



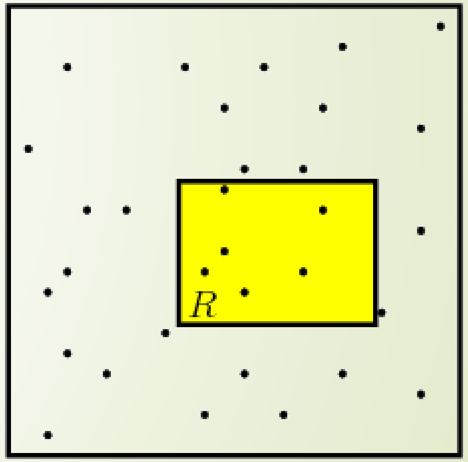


Good Samples

- A good set of samples has the following properties
  - uniformly distributed over the unit square without regular spacing
  - Some minimum distance is maintained between samples points
  - 1D x and y projections of unit square are also uniformly distributed
- In general, a very large sample size satisfy all these properties
  - BUT, it will take you a lot of time to render

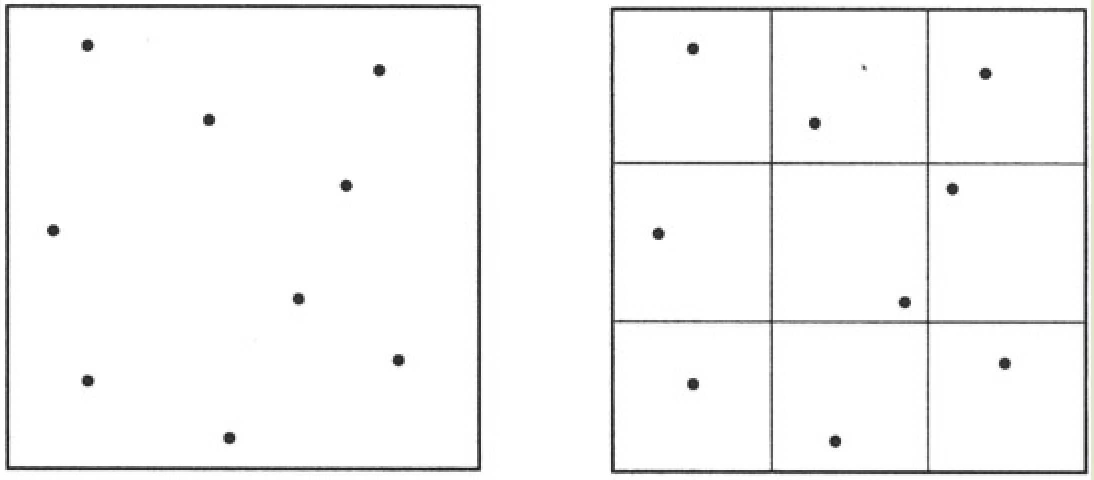
Low-Discrepancy Sampling

- Given a sample P of size k create from a measure space X
- The discrepancy of P w.r.t a range space R is
  - $D(P) = \max_R \left| \frac{\|P \cap R\|}{k} - \frac{\mu(R)}{\mu(X)} \right|$



Jittered Sampling

- Jittered (stratified) sampling
  - Prevent clumping of samples

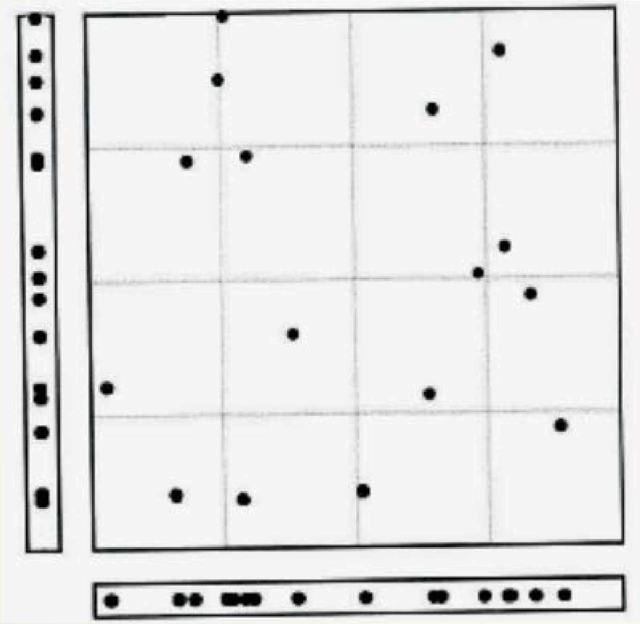


jittered sampling

$$
\begin{array}{l}
\textbf{for } i = 0 \text{ to } n_x - 1 \\
\quad \textbf{for } j = 0 \text{ to } n_y - 1 \\
\qquad k = i n_x + j \\
\qquad x_k = randfrom(i/n_x, (i+1)/n_x) \\
\qquad y_k = randfrom(j/n_y, (j+1)/n_y)
\end{array}
$$

Problem of Jittered Sampling

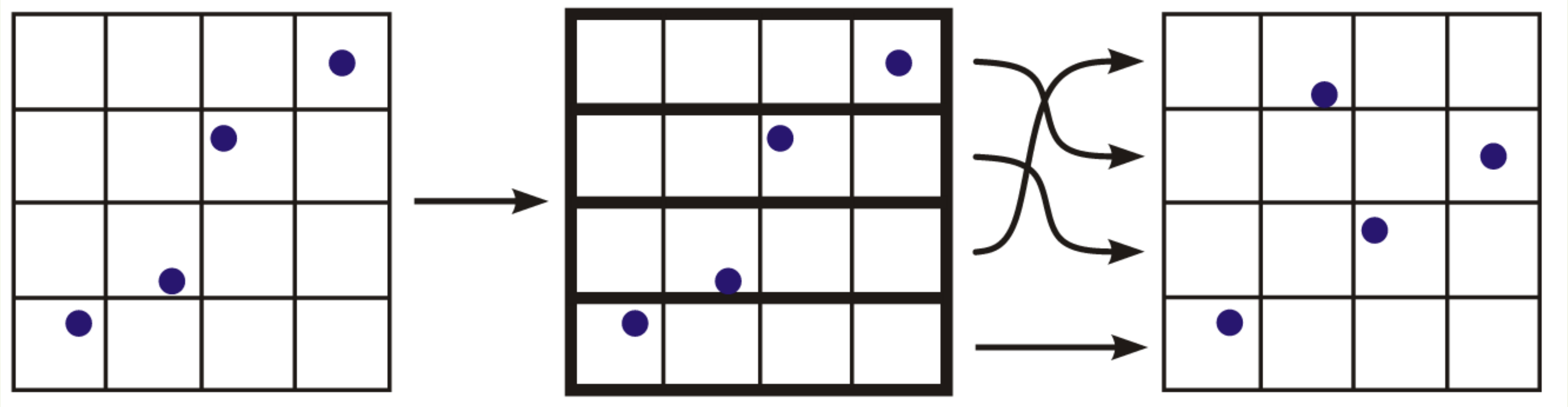
- Jittering suffers when these samples are projected onto the X- or Y-axis



Projections on the *X*- or *Y*-axis are not uniformly distributed

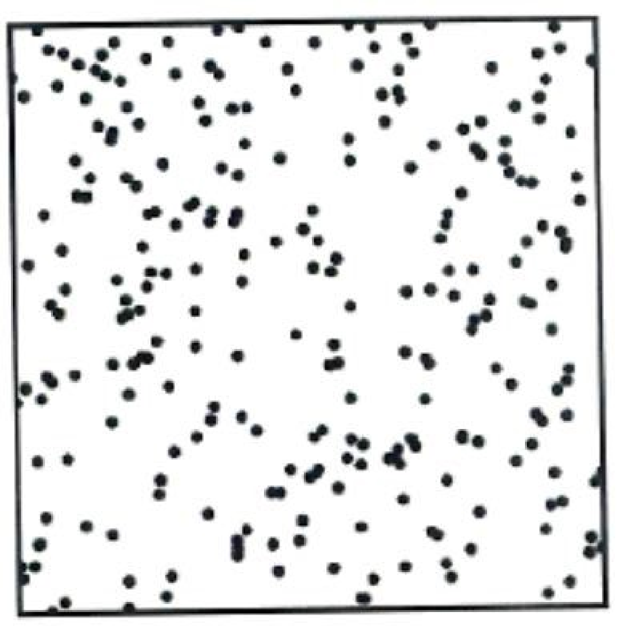
n-rook Sampling

1. Place n rooks on a n by n chessboard
2. Shuffling the y(or x) coordinates of the rooks

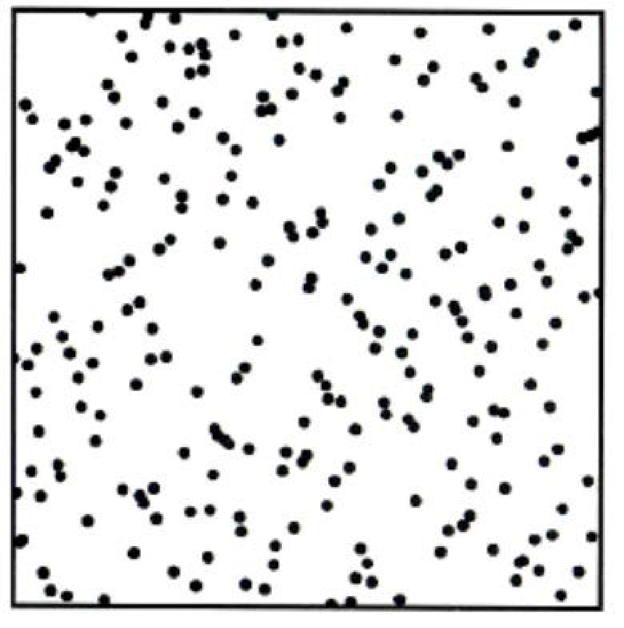


n-rook Sampling

- Problem: 1D projections is fixed, but at the cost of ruining the 2D distribution!

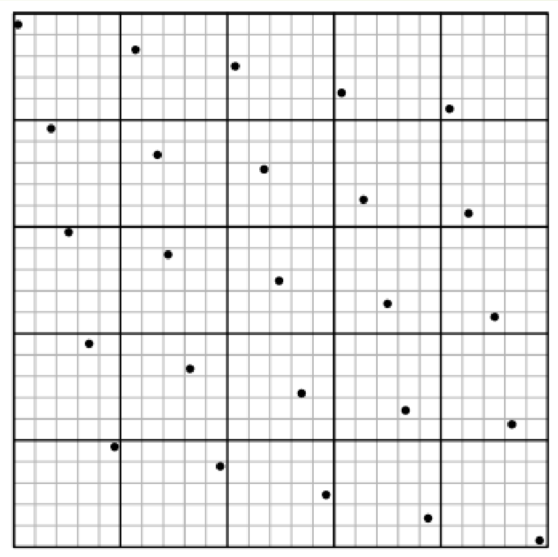


random



n-rook

Multi-Jittered Sampling



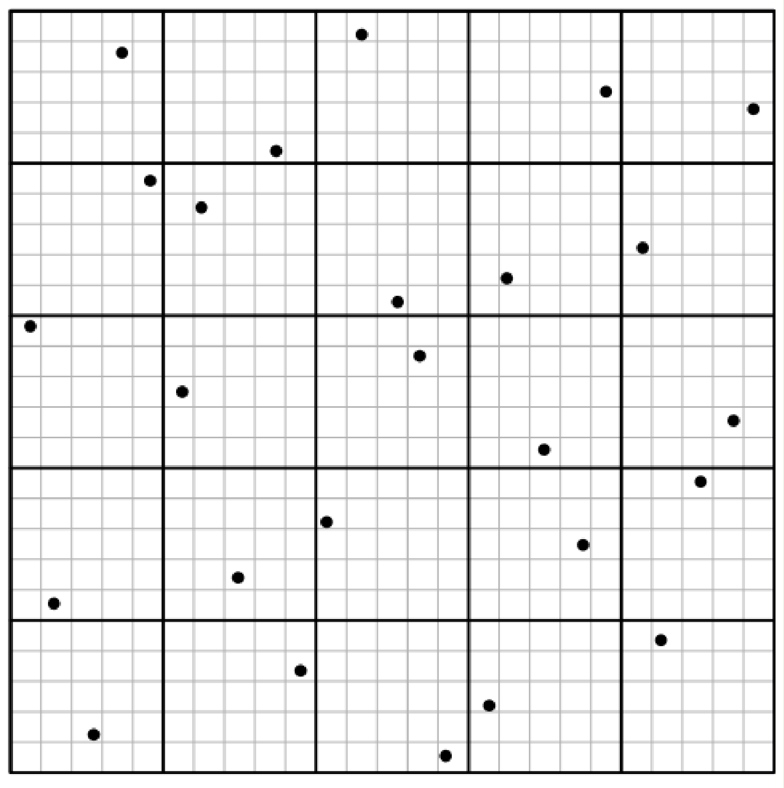
- Multijittered sampling
  - Combine jittering and rook sampling
  - Create a canonical arrangement

```
for (int j = 0; j < n; ++j) {
    for (int i = 0; i < m; ++i) {
        p[j * m + i].x = (i + (j + drand48()) / n) / m;
        p[j * m + i].y = (j + (i + drand48()) / m) / n;
    }
}
```

Multi-Jittered Sampling

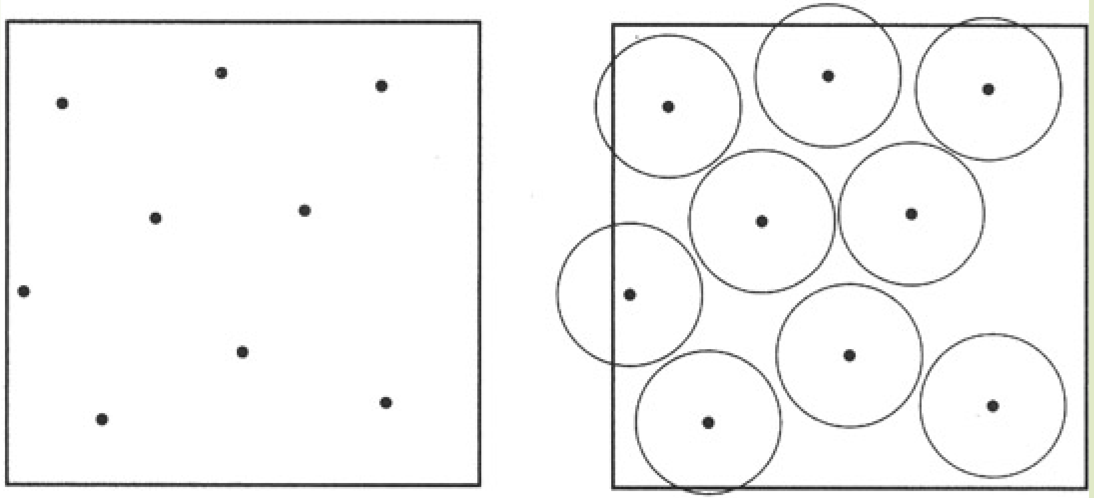
- Multijittered sampling
  - Create a canonical arrangement
  - Shuffle the X coordinates
  - Shuffle the Y coordinates

```
for (int j = 0; j < n; ++j) {
    for (int i = 0; i < m; ++i) {
        int k = j + drand48() * (n - j);
        std::swap(p[j * m + i].x,
                  p[k * m + i].x);
    }
}
for (int i = 0; i < m; ++i) {
    for (int j = 0; j < n; ++j) {
        int k = i + drand48() * (m - i);
        std::swap(p[j * m + i].y,
                  p[j * m + k].y);
    }
}
```



Poisson-Sampling

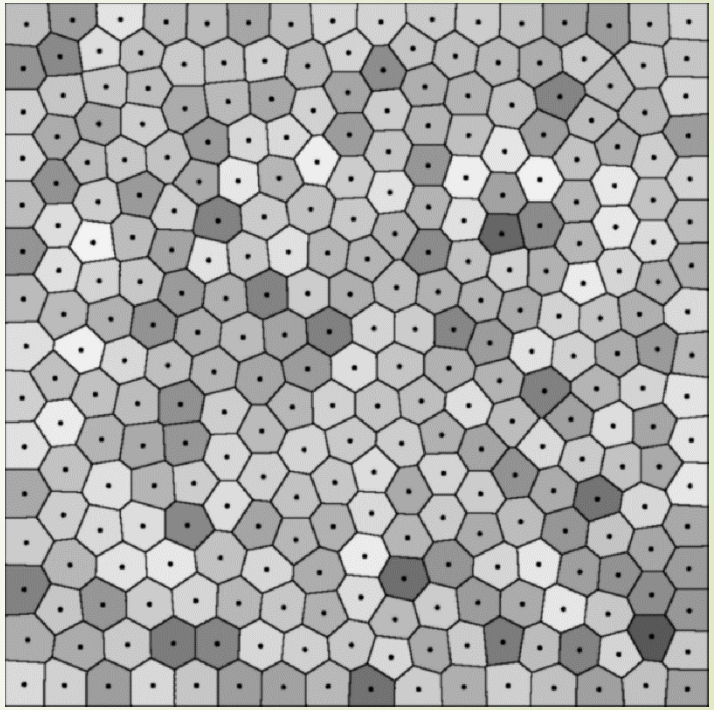
```
i = 0
while i < N
    x_i = randfrom(0, 1)
    y_i = randfrom(0, 1)
    reject = false
    for j = 0 to i − 1
        if [(x_i − x_j)^2 + (y_i − y_j)^2] < d^2
            reject = true
            break
    if not reject
        i = i + 1
```



Poisson-disk sampling

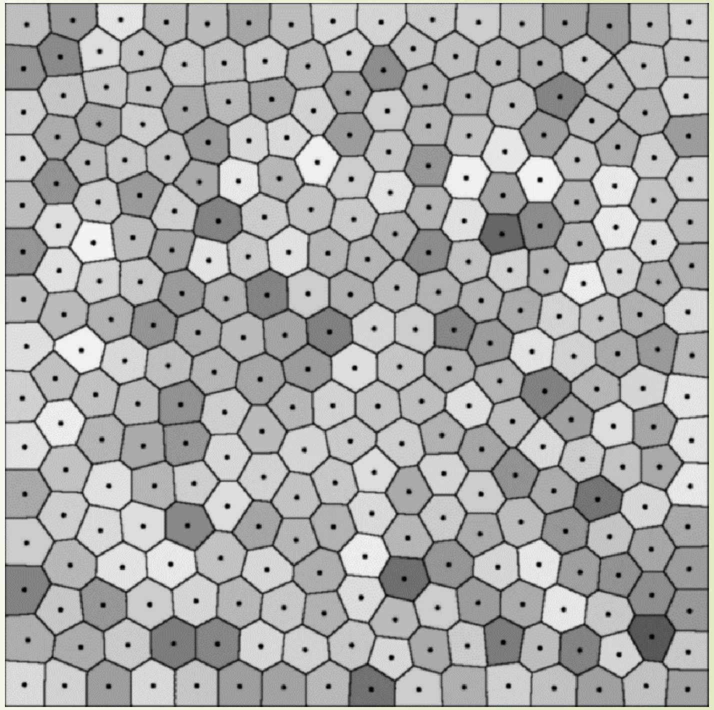
Centroidal Voronoi Diagram (CVD)

- Definition: Voronoi diagram
- Definition: centroidal Voronoi diagram
- CVD provides sampling with low discrepancy



Centroidal Voronoi Diagram (CVD)

- Definition: Voronoi diagram
- Definition: Centroidal Voronoi diagram
- CVD provides sampling with low discrepancy



CVD is very useful in computer graphics, for example in generating stippling drawing

Blue Noise Sampling

- The term “blue-noise” refers to any noise with minimal low-frequency components and no concentrated spikes in energy.
- Blue-noise sampling generates randomized uniform distributions.
- https://www.youtube.com/watch?v=mJOvenBySGs

Sampling and Filtering

- Sample from this 2D function
- $L(x,y) = \frac{1}{2}(1 + \sin\frac{x^2+y^2}{100})$

- http://www.cc.utah.edu/~ndn1/pete-rt/ray1_old.html