Fast Generation of Poisson-Disk Samples on Mesh Surfaces by Progressive Sample Projection

Tong WANG, Reiji SUDA
University of Tokyo
• A practical way to generate Poisson-disk samples on mesh surfaces effectively.
  • Simple inspiration
  • A new parallelization pattern
Poisson-disk Samples

- Randomly place samples, checks that any two are not too close.
  - Distance be larger than specified radius

- Computation time, often very high.
Sampling of nature

Image from Wikipedia
Blue noise feature

• One reason that Poisson-disk sampling is good for many applications is because of its blue noise spectral properties.
Sampling mesh surfaces

- Throwing darts in a clever way
  - Often unbiased, well defined.
  - Not so easy to parallelize.
  - Not very fast.

- Relaxation
  - Can generate more favorable pattern.
  - Need some iterations, often slow.

- Mass Elimination
  - Small bias.
  - Not maximal.
  - Fastest so far.
Elimination

• Process each sample in serial (Yuksel 2015)
Mass Elimination

• Eliminate from a massive amount of samples
  • Sometimes called *Sample pool* or *pre-sampling samples*.

• Eliminate samples in parallel.
  • [Bowers et al. 2010 ]
  • [Ying et al. 2013]
Problem in Mass Elimination

- Which to keep, which to eliminate.
  - Without causing chain reaction.
  - Without racing.
Parallelize elimination process

- Use phase groups. ([Bowers et al. 2010])
  - Divide space to phase groups.
  - Threads processing in the same phase group is isolated and have no interaction with other threads.
  - Lower parallelism, some unfavorable bias.
Parallelize elimination process

- Use priority ([Ying et al. 2013])
Sampling mesh surfaces

- Throwing darts in a clever way
  - Often unbiased, well defined.
  - Not so easy to parallelize.
  - Not very fast.

- Relaxation
  - Can generate more favorable pattern.
  - Iterations, often slow.

- Mass Elimination
  - Small bias.
  - Fast.
  - Low parallelism or irregular parallelization

- Sample Projection
Another parallel pattern

• Draw inspiration form phase group:
  • Samples added should have no intra-batch conflicts

• Draw inspiration from priority-based method:
  • Samples can have different priority.

• Also form food baking:
  • Rotating food to get better coverage of heat.
Sample Plane (projector)
Sample Projection

• Prepare some 2D Poisson-disk sample patterns (projectors)
• Project samples on the projectors towards mesh surfaces (using ray tracing).
• Eliminate conflicts.
• Rotate projectors.
• Repeat.
What is needed

• A pre-generated 2D Poisson-disk sample pattern of a specified radius to start with.
  • Quality of the 2D pattern will not affect too much on samples of projection space.

• A fast ray tracer to do the projection.
  • Embree or
  • OptiX on NVIDIA GPUs.
Overview of the algorithm

1. Build Orthogonal Ray Batch Group (ORBG).
2. Project samples onto mesh by tracing rays.
3. Eliminate conflicts.
4. If meets demand, break and output.
5. Else, rotate the ORBG, go to 2.
Orthogonal Ray Batch Group (ORBG)

- 2D Sample plane is called a projector or view plane.
- Each plane will generate ray batch for projection.
- Three orthogonal planes form a group called ORBG.
Project samples onto mesh surfaces

- Record every intersections through each ray.
- Deal with restart conflicts.
Conflict elimination
Rotate ORBG

- Rotate projectors that generate the ORBG.
- Rotation destinations should be low-discrepancy.
Conflict elimination

• Samples generated from the same projector have no conflicts.
  • Except restart conflicts.

• Resolve conflicts from different projectors using priority.
  • Very short priority chain.
What’s good about this pattern

• No sample-pool needed (Sample-pool free)
  • Generating Poisson-disk sample from scratch.

• No intra-batch conflicts (intra-batch conflict free)
  • No nested priority problem during elimination phase.
  • Regular and GPU friendly.
Results and analysis

- Performance (CPU: i7-7700K, GPU: NV GTX1080)

![Graph showing performance comparison](chart.png)
Iterations: 50
\( \rho / \rho_{\text{max}} = 92.5\% \)

Iterations: 50
\( \rho / \rho_{\text{max}} = 91.0\% \)

Iterations: 50
\( \rho / \rho_{\text{max}} = 93.5\% \)
Sample Rate:
- Bunny (1.54M Samples)
- Dragon (1.13M Samples)
- Buddha (0.93M Samples)
- XYZRGB Dragon (0.63M Samples)

Radius Statistics:
- Bunny
- Dragon
- Buddha
- XYZRGB Dragon
Iteration: 1
SR: 8.7M/s
Samples: 624618
$\rho/\rho_{\text{max}} = 79.1\%$

Iteration: 3
SR: 3.3M/s
Samples: 847880
$\rho/\rho_{\text{max}} = 85.9\%$

Iteration: 100
SR: 118K/s
Samples: 1114999
$\rho/\rho_{\text{max}} = 96.5\%$

Iteration: 30
SR: 413K/s
Samples: 1055810
$\rho/\rho_{\text{max}} = 95.2\%$
Some choice to explain

- Shape of projectors
  - Plane can ensure intra-conflict free criteria.
  - Non-Euclidean surface degenerate the merging step to a mass elimination problem.
Some choice to explain

• Size of each project set, project from an arbitrary polyhedron rather than hexahedron.
  • Complex priority setting.
  • The choice of cube is a balance between parallelism and regularity.
  • It is also possible to generate usable samples for only one or few iterations.
• Project one projector at a time.
  • Less parallelism
  • More kernel launch.

• Project X (X>3) projectors at a once.
  • More complex numbers, irregular execution, unbalanced workload.
Choice on ray tracing or rasterization

- Sample projection ➔ Sample rasterization
- Rasterization is basically a faster primary ray tracing.

A brief flow
  - Rasterize mesh to 2D sample plane.
  - Fetch points from the target (2D sample plane) only on sample positions.
  - Eliminate conflicts the same way.
  - Iterate this process by rotating the object (or plane)
Test on equivalent implementation

• The implementation of rasterization method can greatly affect the result.
More applications

- Few iterations can generate usable results.
Easy to sample adaptively

Dragon
Tris: 871K
Samples: 3.2M
Iterations: 250
SR: 60K/s
Image Stippling

Samples: 2.4 Million
Effective Sampling Rate: 100K/s
Samples: 2.3 Million
Effective Sampling Rate: 105K/s
Limitations

• Better for generation of massive amount of samples.
  • A little bit heavy on sparse samples.

• Convergence is non-linear.
  • Gap management will be a good future work.

• Radius based, hard to control the exact number of samples.
  • Combined with other methods to control the number.

• No theoretical explanation on the bias of this method.
Thank you!