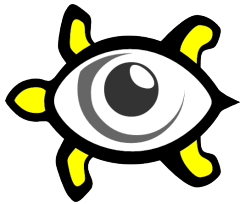
PixelPie: Maximal Poisson-disk Sampling with Rasterization

C. Y. Ip M. A. Yalçin D. Luebke A. Varshney

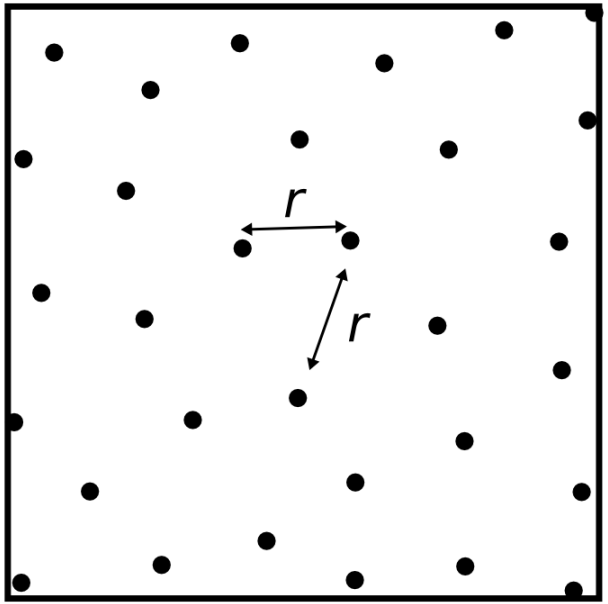
UNIVERSITY OF MARYLAND



NVIDIA

Poisson-Disk Distribution

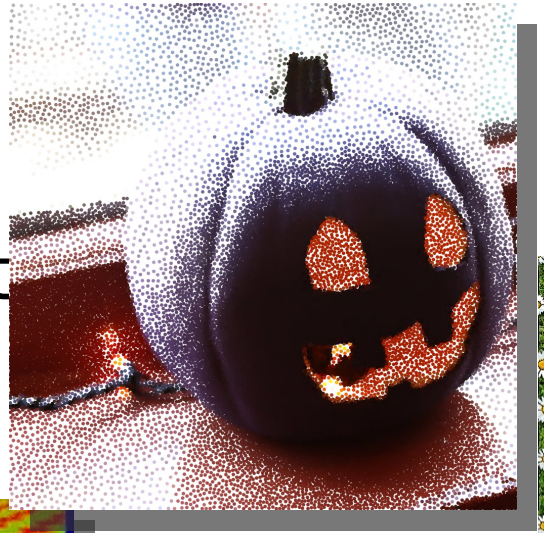
Poisson-Disk Distribution



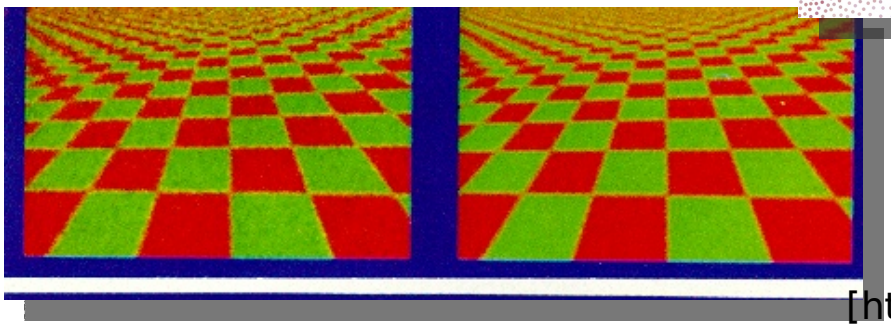
Random samples that are at least r apart

Applications

- Antialiasing
- Dithering
- Object Placement



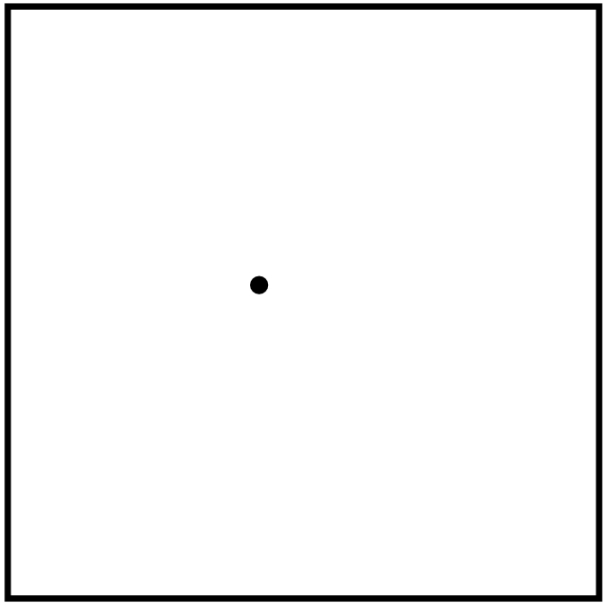
[Dippe & Wald SIGGRAPH 1985]



[http://devmag.org.za/2009/05/03/poisson-disk-samplin

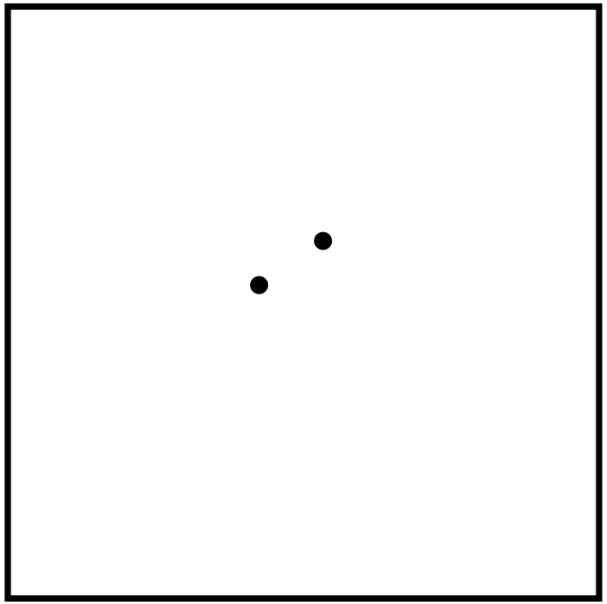
Dart Throwing

[Cook SIGGRAPH 1986]



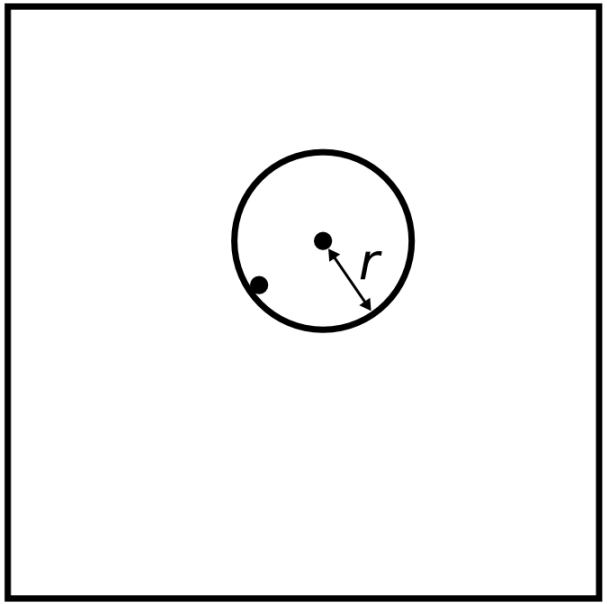
Dart Throwing

[Cook SIGGRAPH 1986]



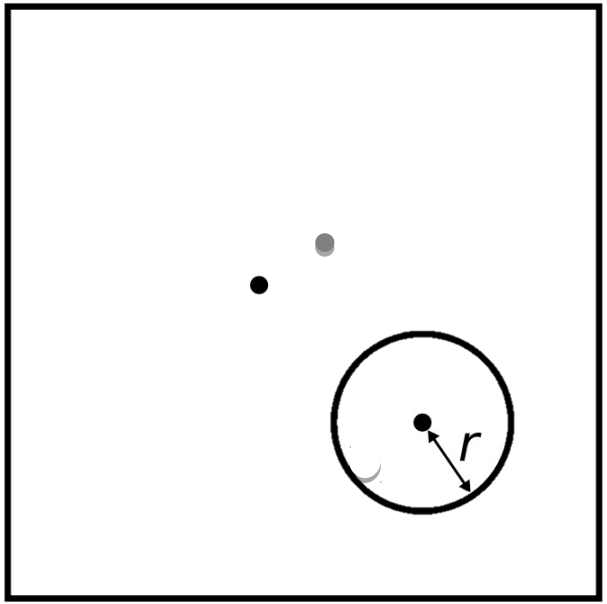
Dart Throwing

[Cook SIGGRAPH 1986]



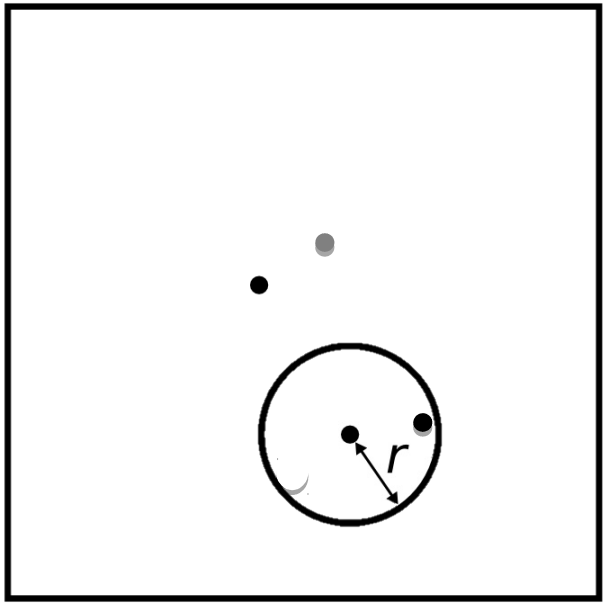
Dart Throwing

[Cook SIGGRAPH 1986]



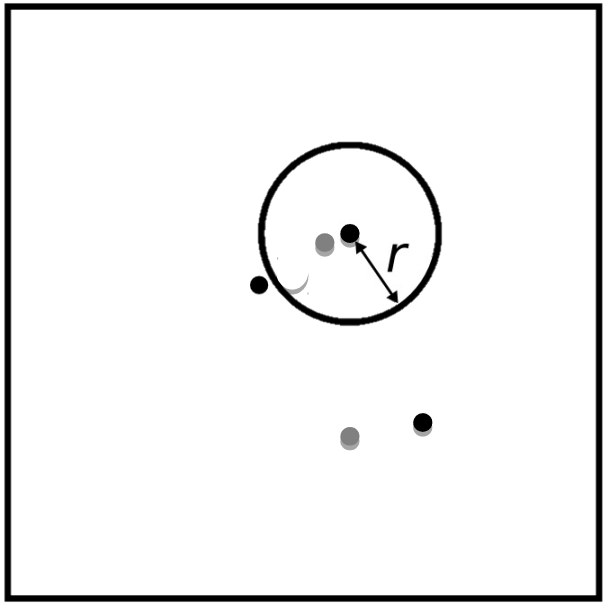
Dart Throwing

[Cook SIGGRAPH 1986]



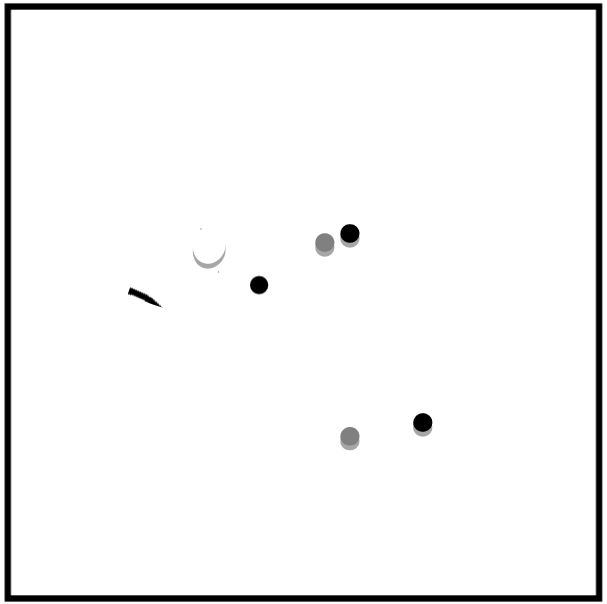
Dart Throwing

[Cook SIGGRAPH 1986]



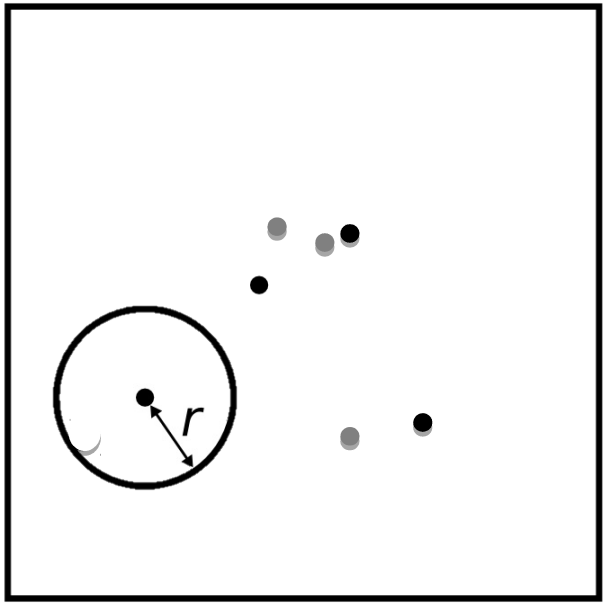
Dart Throwing

[Cook SIGGRAPH 1986]



Dart Throwing

[Cook SIGGRAPH 1986]

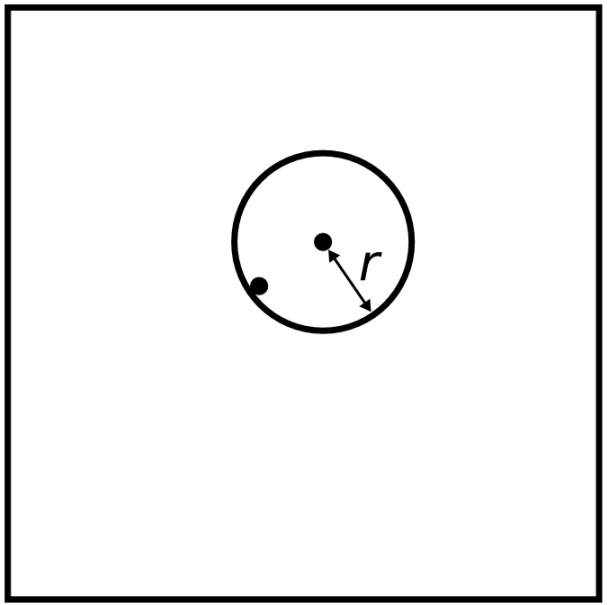


Related Work

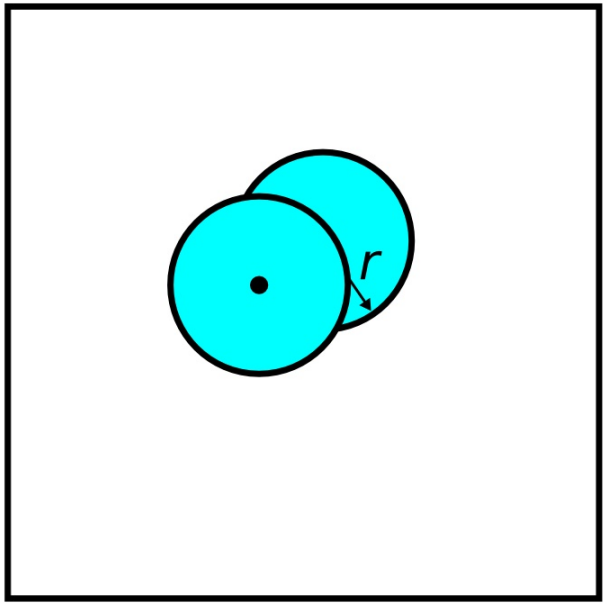
- Spatial Data Structure Approaches:
  - Trees [Dunbar & Humphreys ToG 2006, Gamito & Maddock ToG 2009]
  - Grid [Jones et al. JGT 2006, 2011]
- Approximate Approaches:
  - Tiles [Lagae & Dutre CGF 2005, 2006, Kopf et al. ToG 2006, Ostromoukhov et al. ToG 2004, 2007]
  - Farthest Point Optimization [Balzer ToG 2009, Chen & Gotsman CGF 2012, de Goes et al. ToG 2012]
- Parallel Approaches:
  - GPGPU [Wei ToG 2008]
  - CUDA [Bower et al. ToG 2010, Xiang et al. SIGGRAPH 2011, Ebeida et al. ToG 2011, CGF 2012]
- Survey [Lagae & Dutre CGF 2006]

Concept

When There is a Conflict…



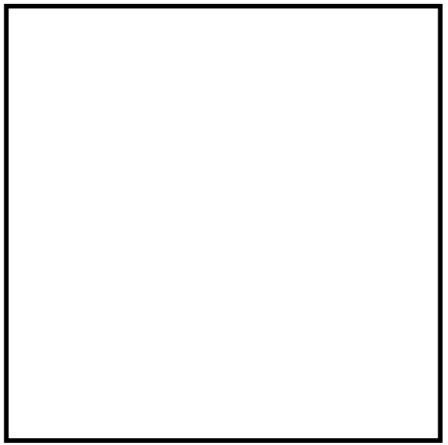
Draw Solid Disks



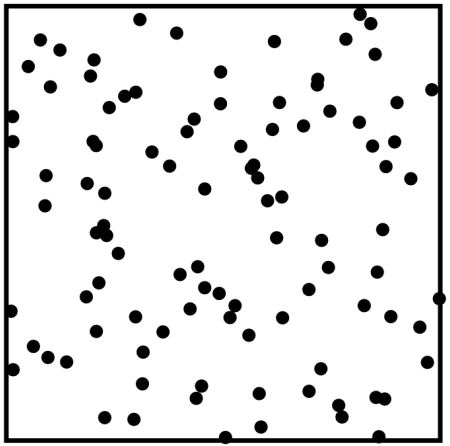
The conflicted dart center is occluded

In Parallel

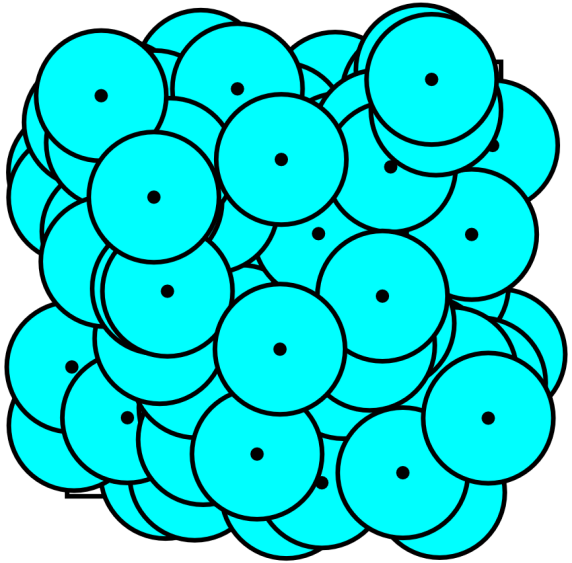
Empty Domain



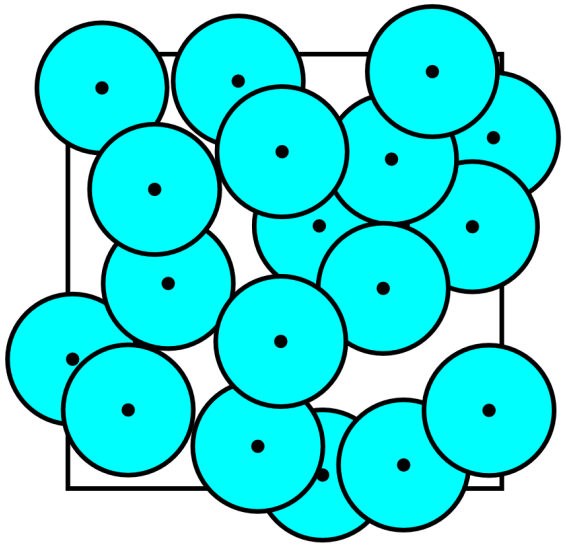
Random Darts



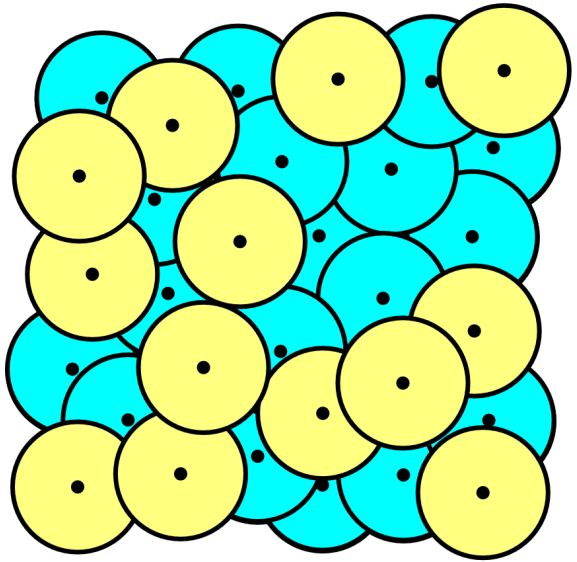
Draw Solid Disks around the Darts



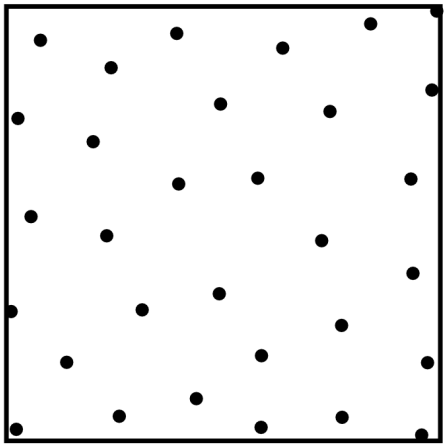
Remove Darts with Occluded Center



Iteratively Fill in the Empty Regions



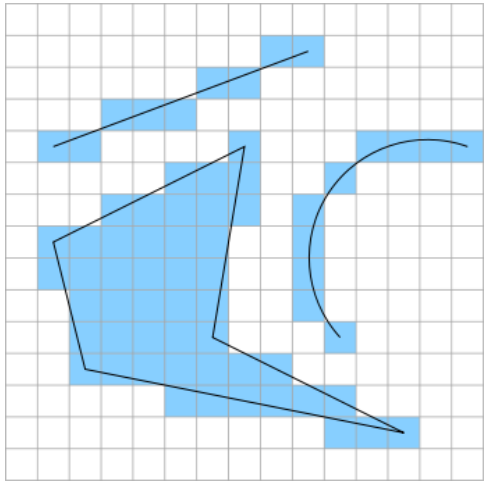
Poisson-disk Distribution



Rasterization Implementation

Rasterize Disks on Textures

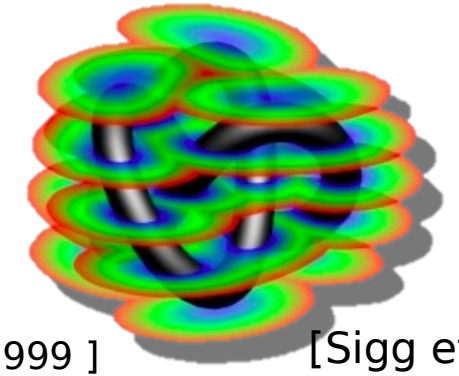
- Render solid disks by **Rasterization**
- Target Domain: **Discrete depth texture**
- Reject conflicts by **Occlusion Queries**



Related Work

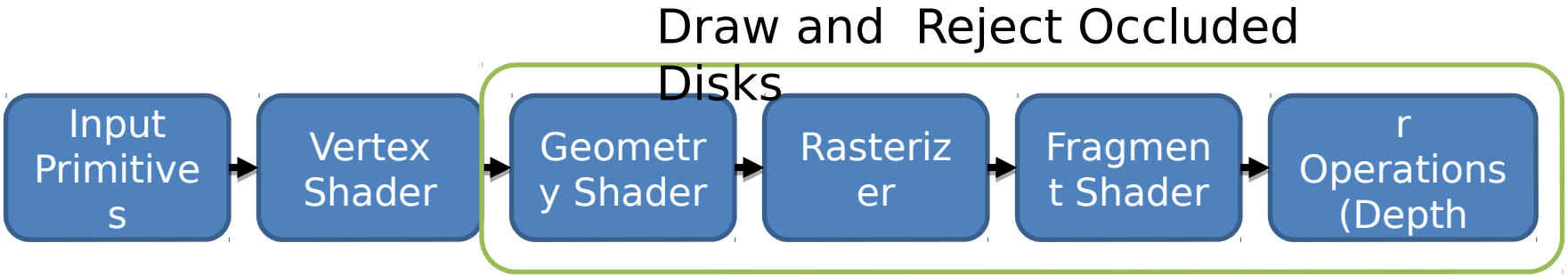
- Voronoi Diagrams and Distance Transform by Rasterization
  - GPU Rasterization [Hoff et al. SIGGRAPH 1999]
  - 3D Distance transform [Sigg et al. Vis 2003, Sud et al. I3D 2004]
  - High order Voronoi Diagrams [Fischer & Gostman JGT 2006]
  - Jump Flooding Approach [Rong et al. I3D 2006, TVCG 2011]

[Hoff et al. SIGGRAPH 1999]



[Sigg et al. Vis 2003]

Graphics Pipeline: 2-Step Iterative Procedure



Step 1: Throw Random Darts

Emit Triangles

Trim Triangles to Disks on a Depth Map

Step 2: Re-throw Darts

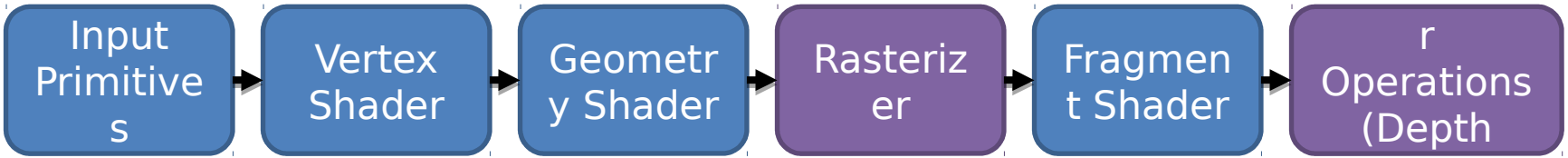
Reject Conflicts with Occlusion Queries

Trim Triangles to Disks on a Coverage Map

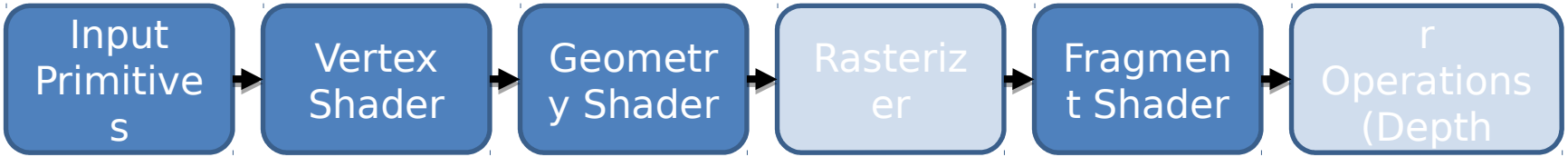
Post-processing: Find empty regions (CUDA stream compaction)

Iteratively Fill the empty regions

How about using CUDA?

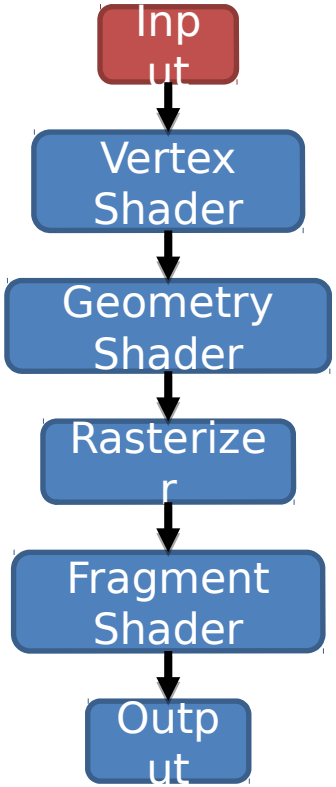


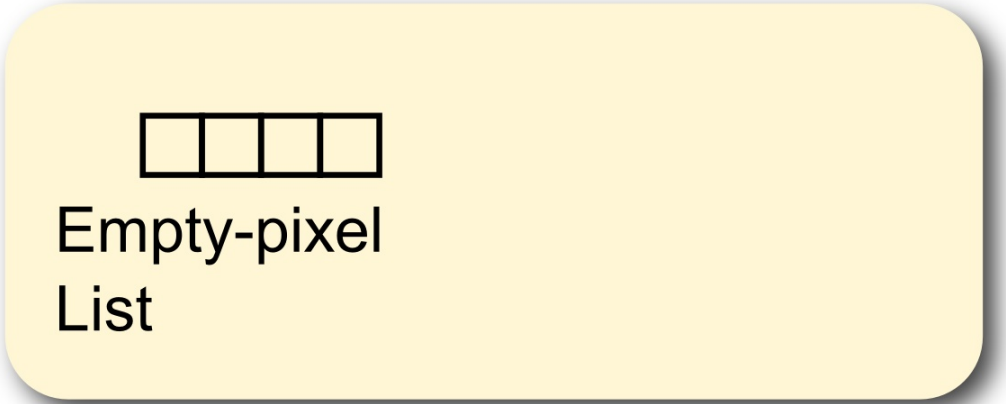
How about using CUDA?



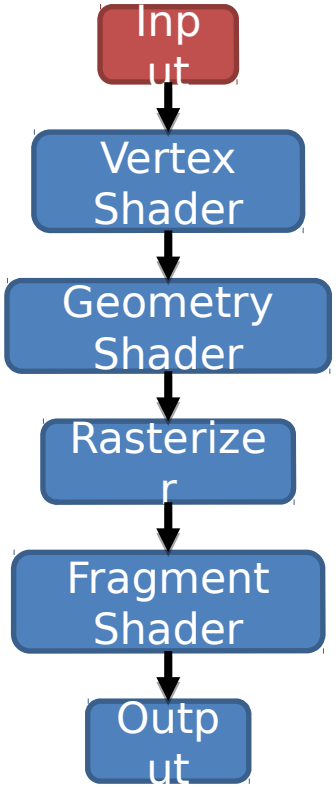
- **Hardware rasterizer and depth test not currently available in CUDA**
- **CUDA software rasterization is slower than the hardware [Laine & Karras HPG 2011]**
- **Use GPU hardware to accelerate geometry computing**

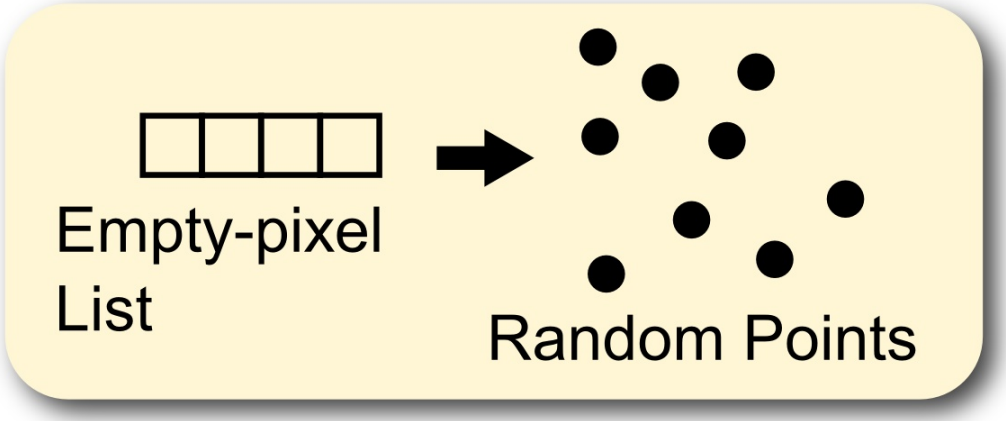
Step 1: Input


Input
Vertex Shader
Geometry Shader
Rasterizer
Fragment Shader
Output

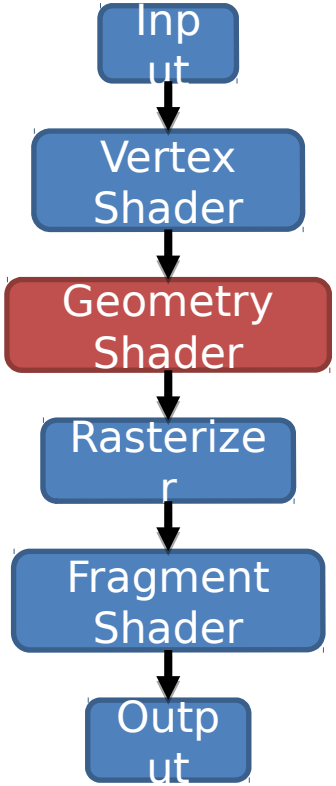

Empty-pixel List

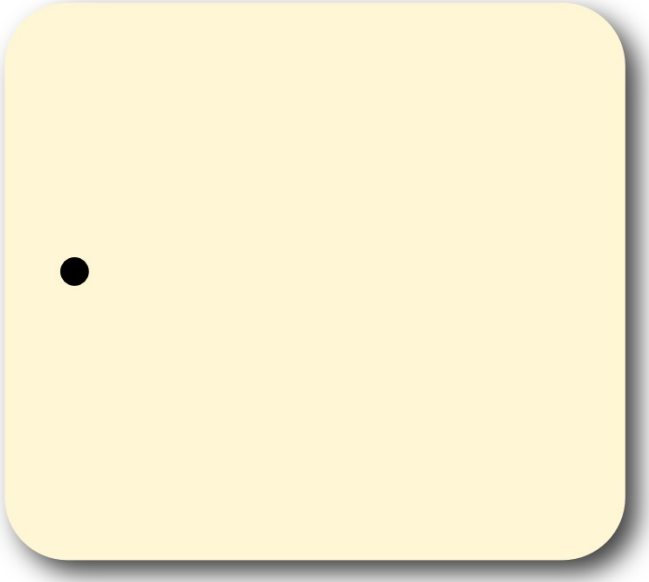
Step 1: Input


Input
Vertex Shader
Geometry Shader
Rasterizer
Fragment Shader
Output

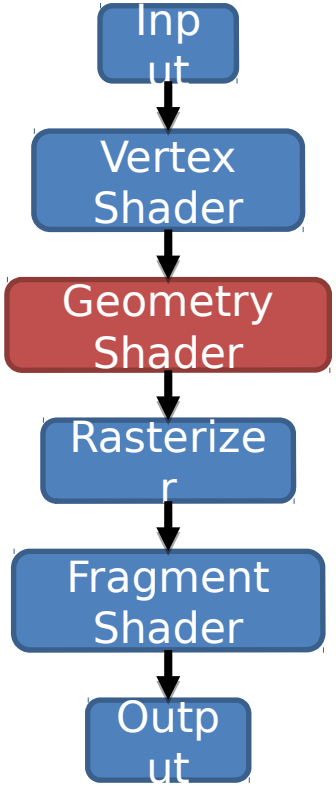

Empty-pixel List
Random Points

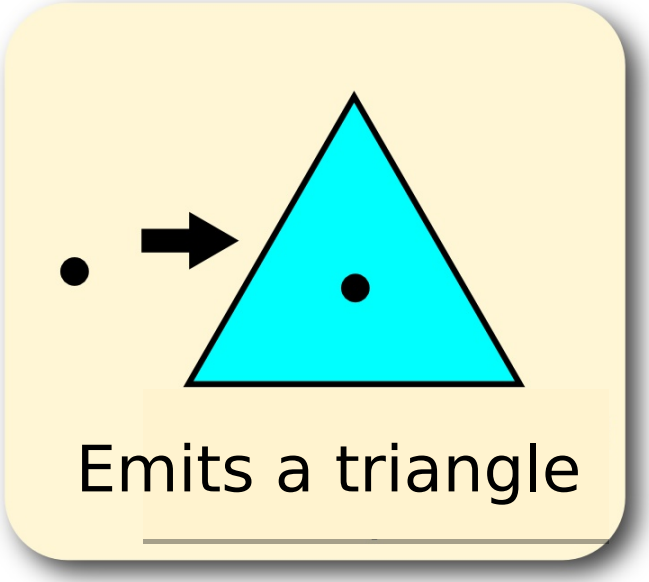
Step 1: Geometry Shader





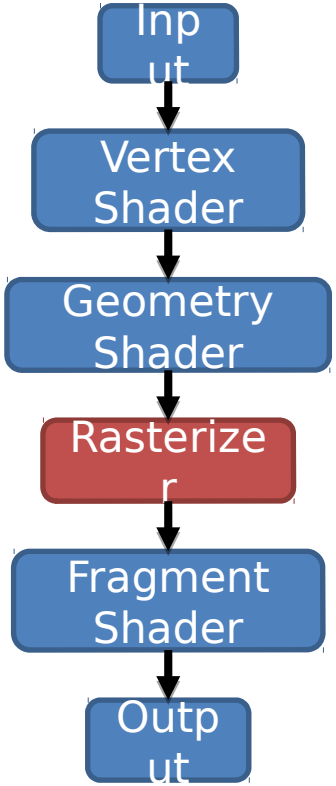
Step 1: Geometry Shader

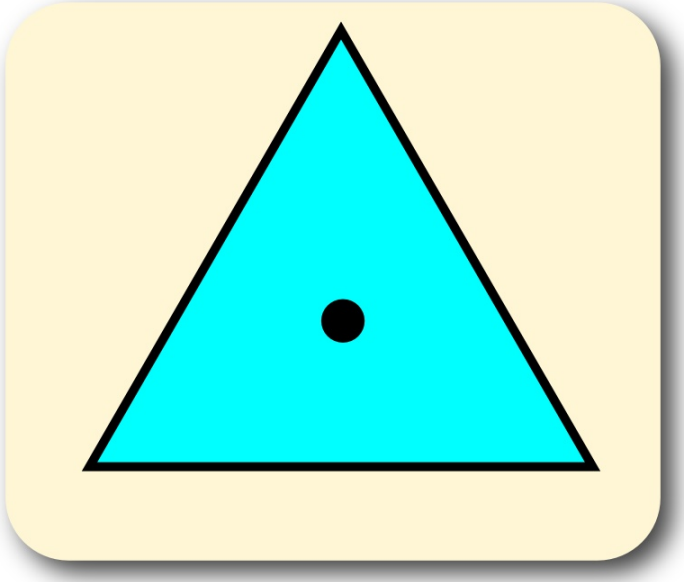




Unique depth by with

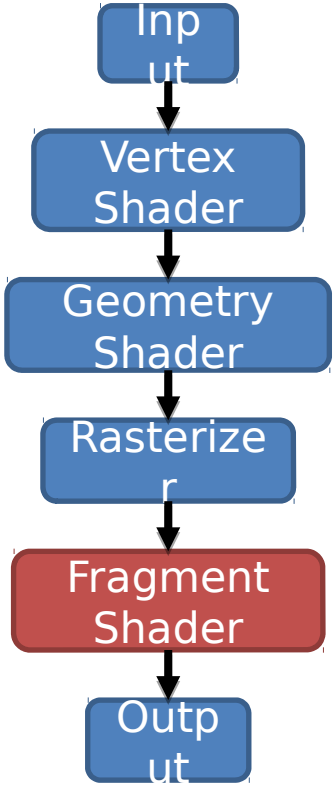
Step 1: Rasterizer

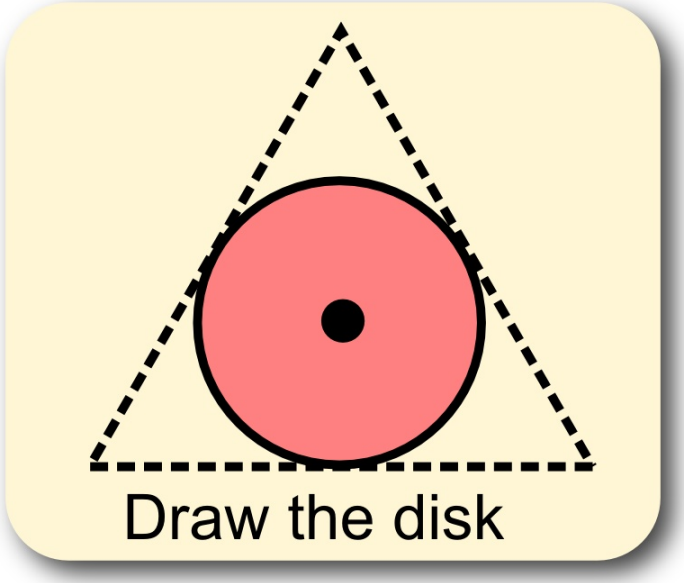




The Rasterizer rasterizes

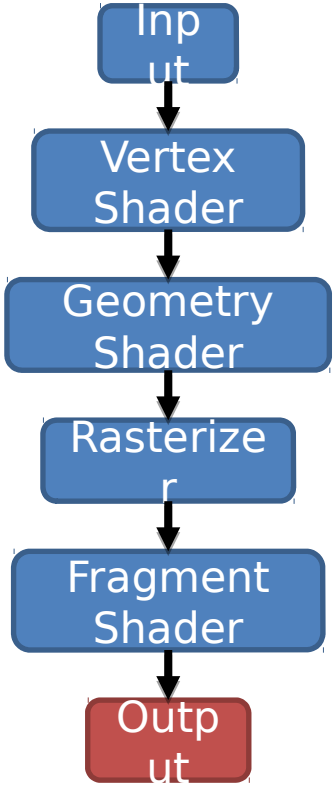
Step 1: Fragment Shader

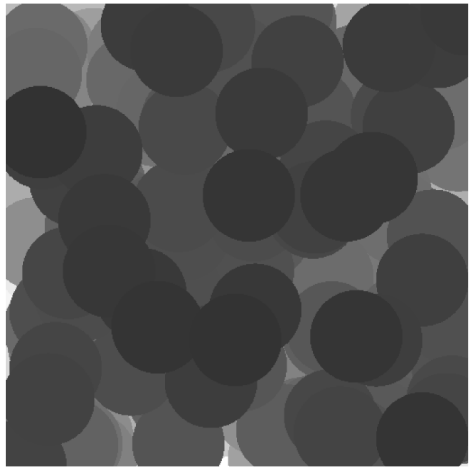




Trims the triangle to

Step 1: Output

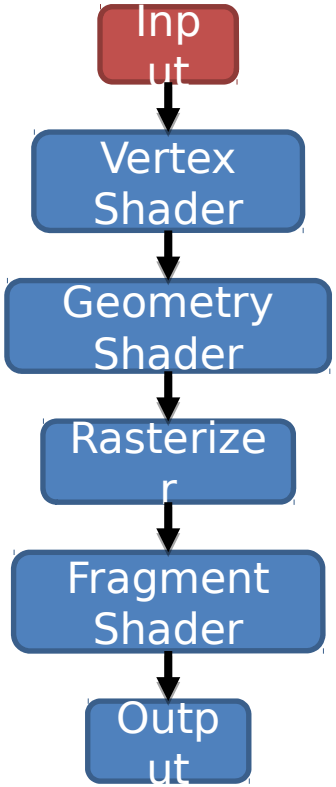


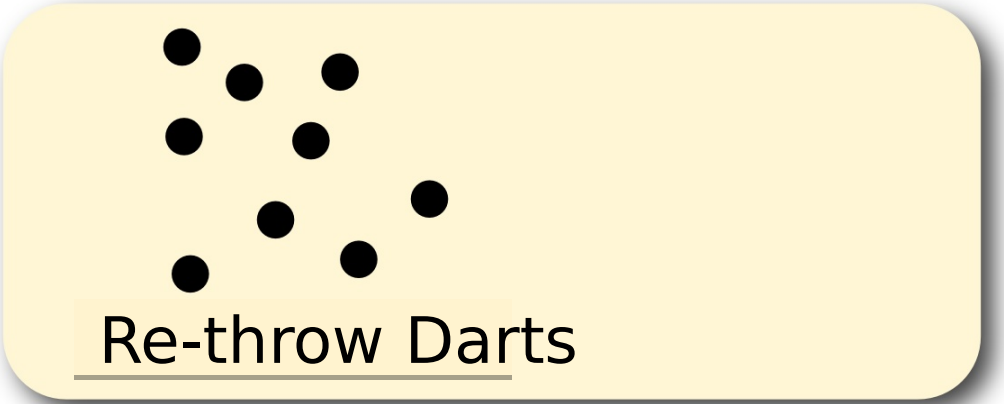


Depth Map

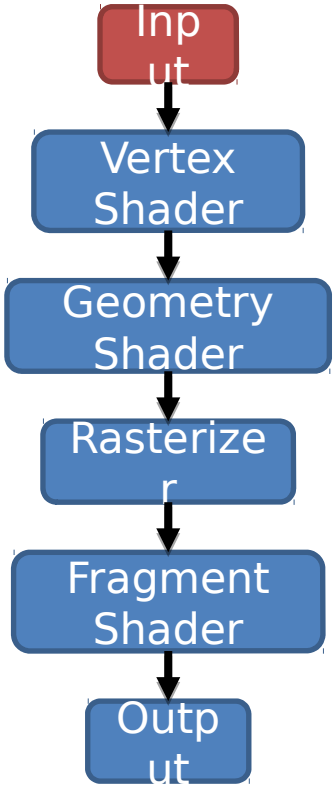
Depth Test ensures proper occlusions

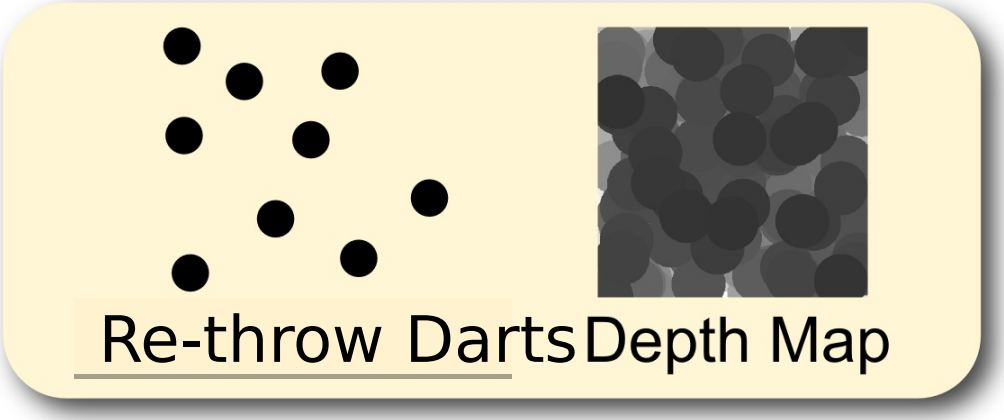
Step 2: Input


Inp
ut
Vertex
Shader
Geometry
Shader
Rasterize
r
Fragment
Shader
Outp
ut

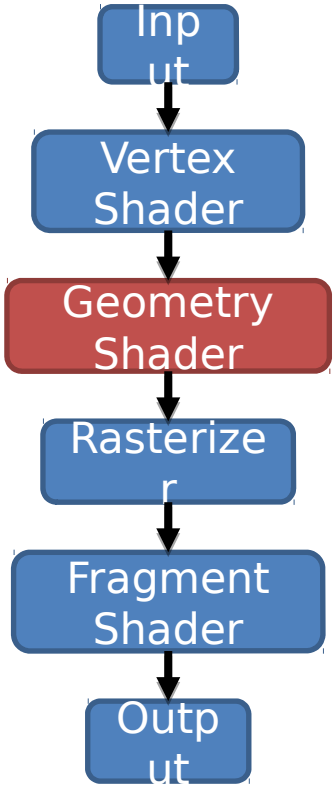

Re-throw Darts

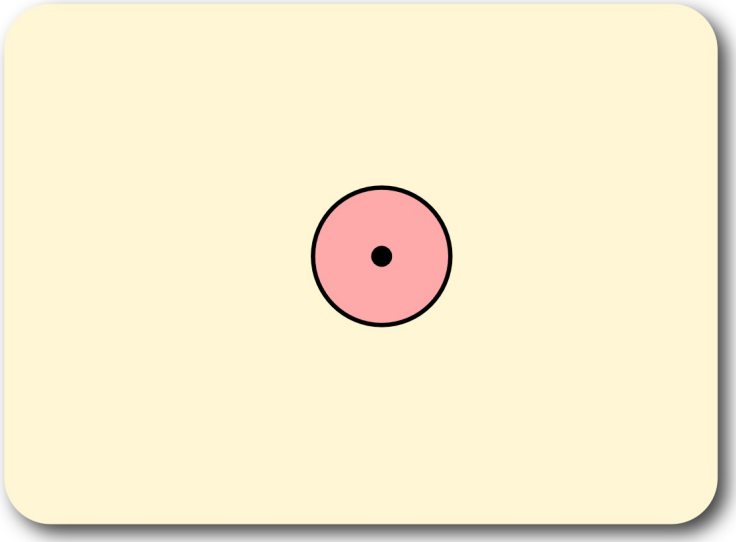
Step 2: Input

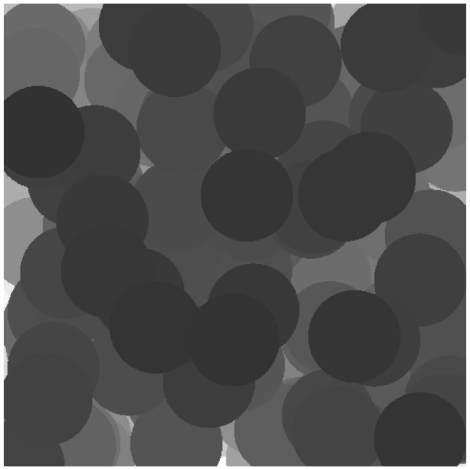

Inp
ut
Vertex
Shader
Geometry
Shader
Rasterize
r
Fragment
Shader
Outp
ut


Re-throw Darts
Depth Map

Step 2: Geometry Shader

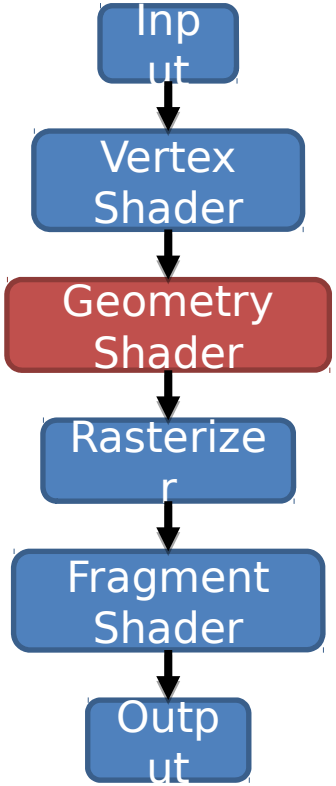


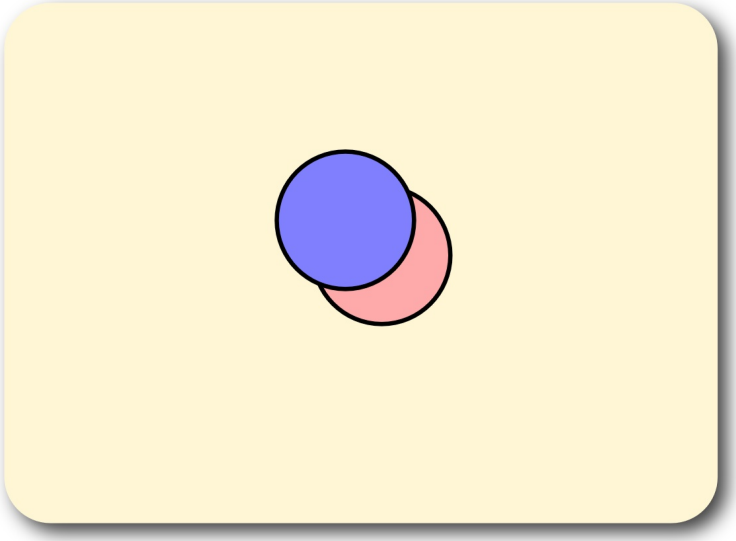




Query the depth map for

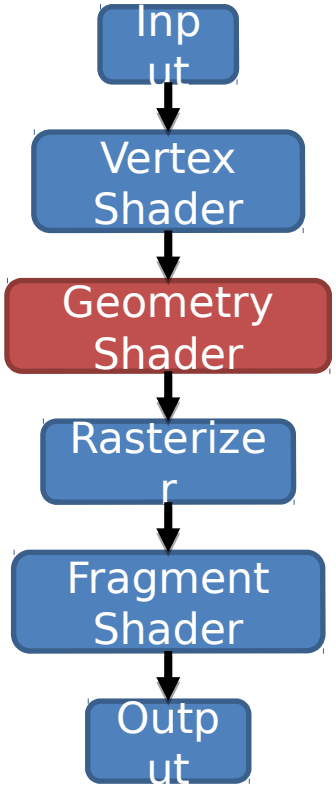
Step 2: Geometry Shader

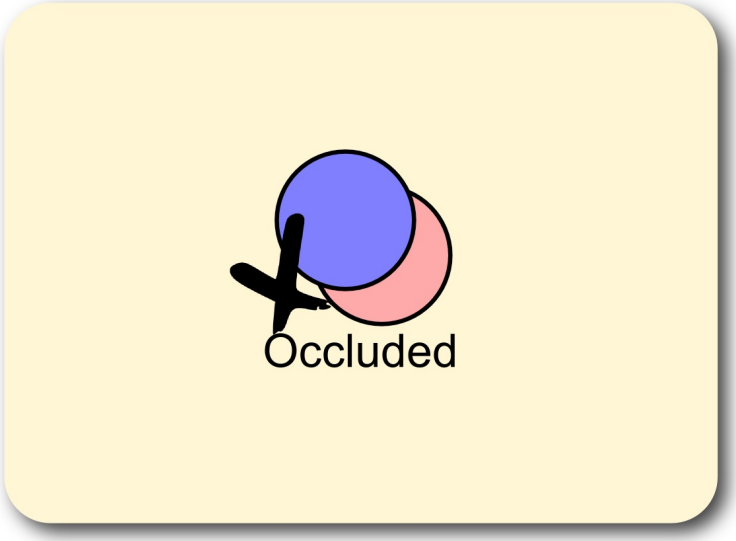




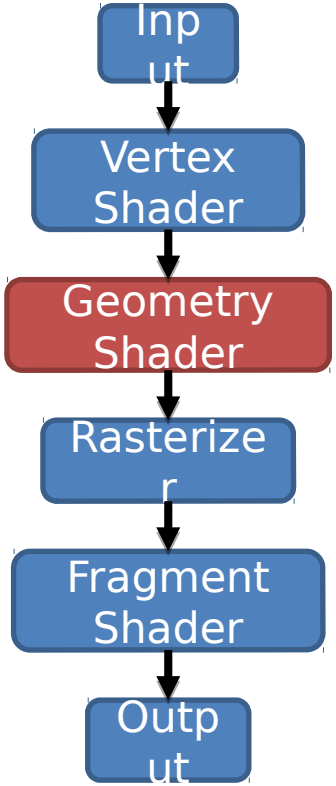
Occluded Dart Center:
Depth-map depth <

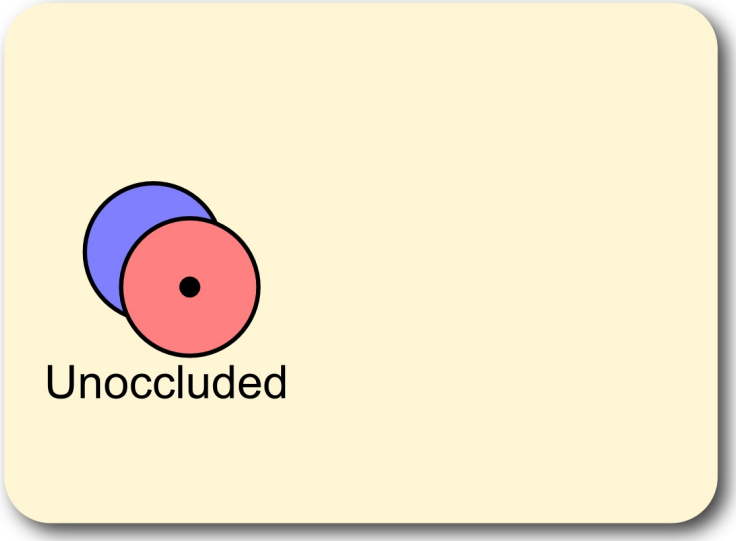
Step 2: Geometry Shader





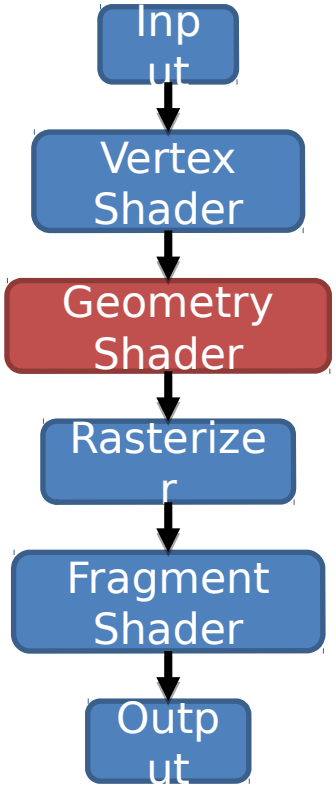
Step 2: Geometry Shader

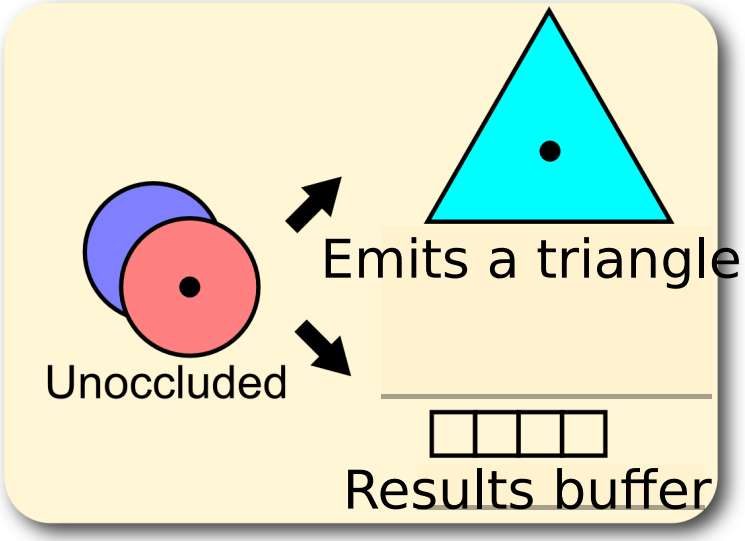




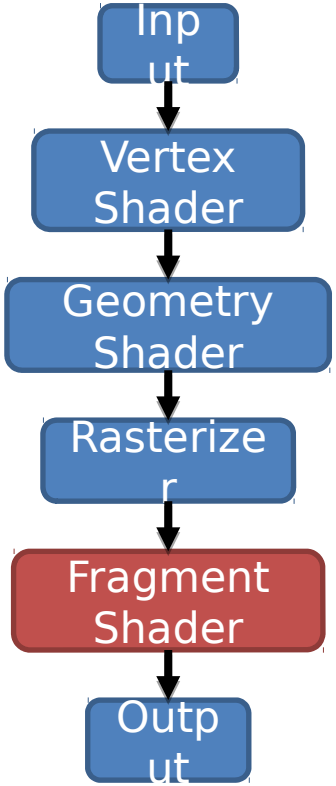
Unoccluded Dart Center:
Depth-map depth =

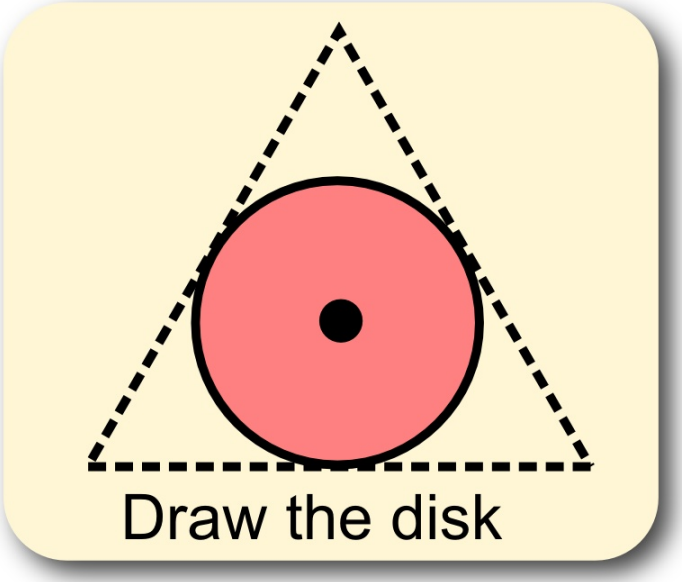
Step 2: Geometry Shader





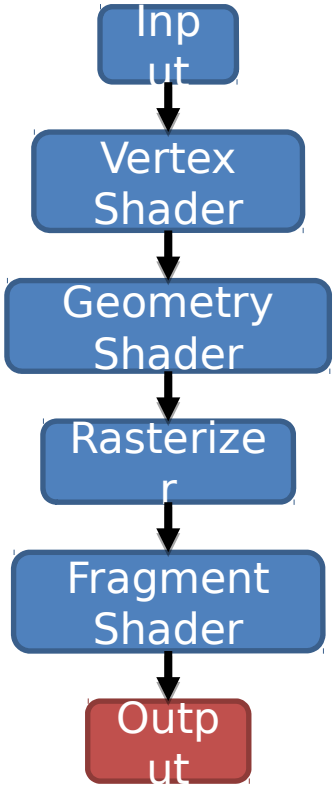
Step 2: Fragment Shader

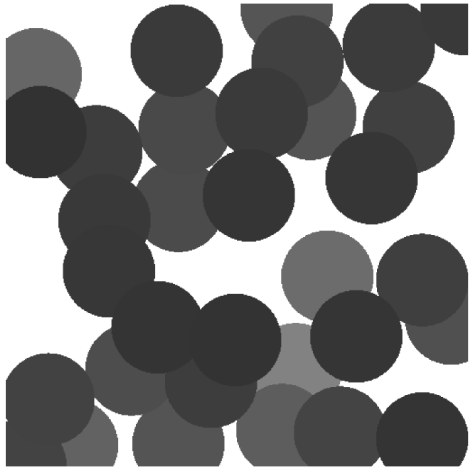




Trims the triangle to

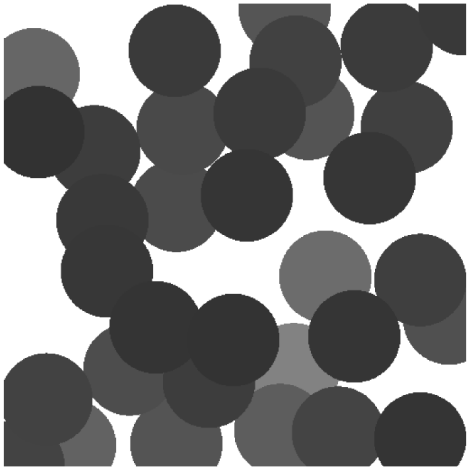
Step 2: Output





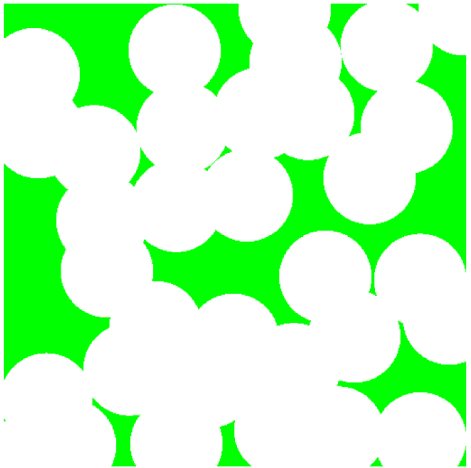
Coverage Map

Post Processing



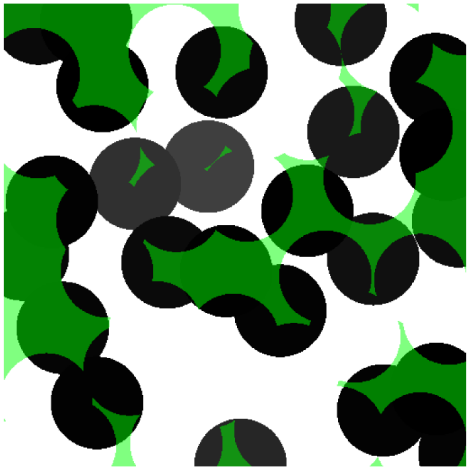
Update the empty pixel list

Stream Compaction (CUDA Thrust)



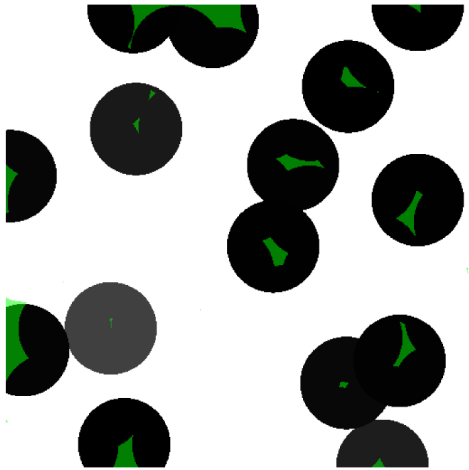
Update the empty pixel list

Keep Iterating…



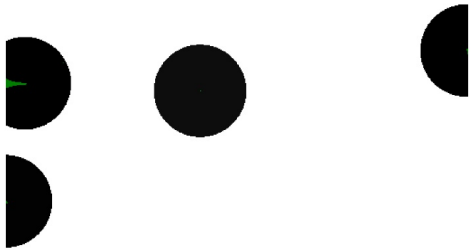
Iteration 2

Keep Iterating…



Iteration 3

No Empty Pixel: Complete



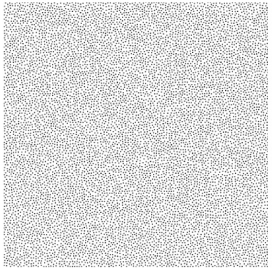
Iteration 4

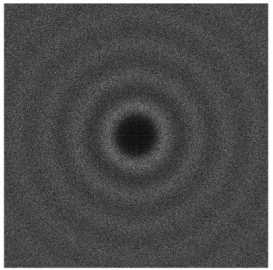
Guarantee Maximal !

Quality

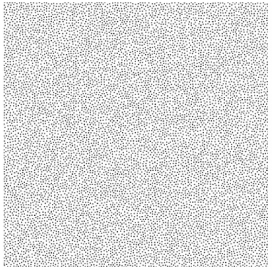
Spectrum Quality Comparison

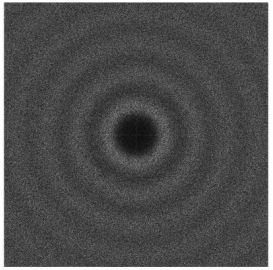
4096^2





[Gamito and Maddock ToG 2009]



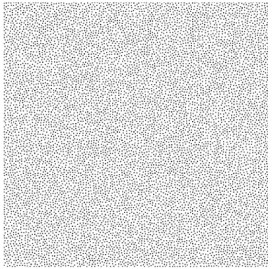


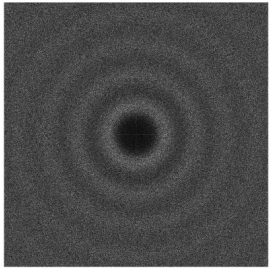
Samples

Power Spectrum
Periodogram

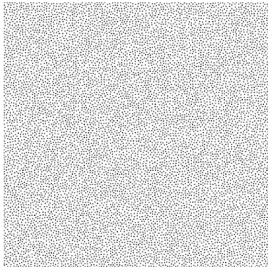
Spectrum Quality Comparison

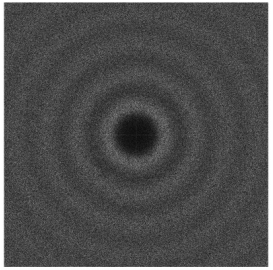
2048^2





[Gamito and Maddock ToG 2009]



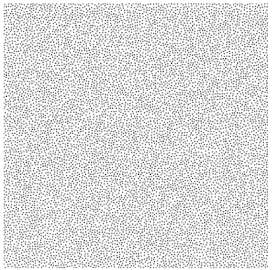


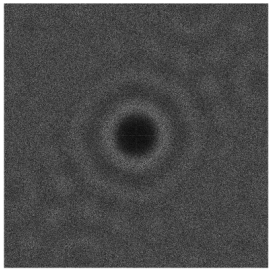
Samples

Power Spectrum
Periodogram

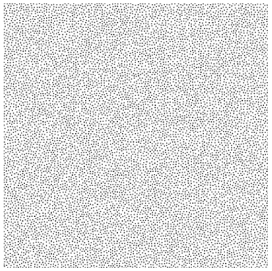
Spectrum Quality Comparison

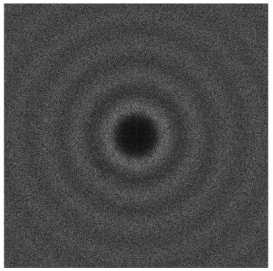
1024^2





[Gamito and Maddock ToG 2009]

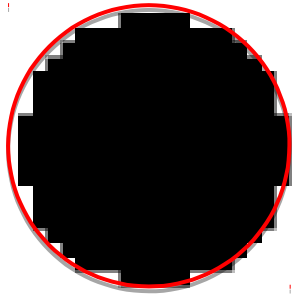


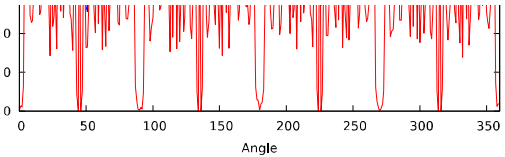


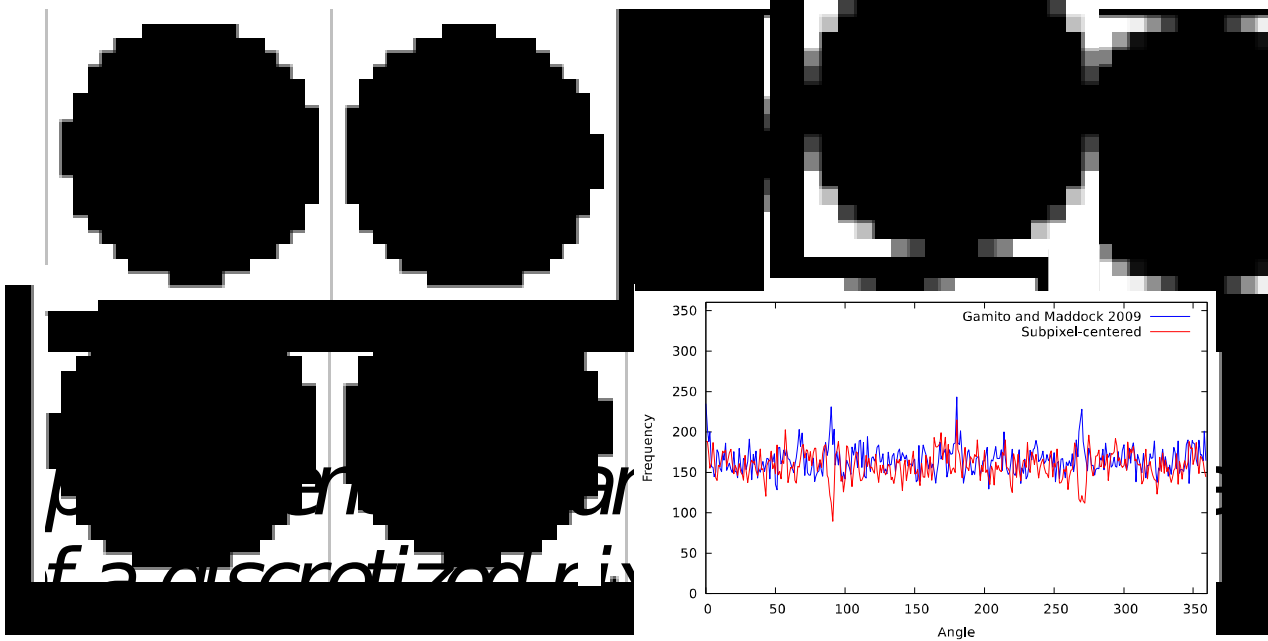
Samples

Power Spectrum
Periodogram

Angular Bias of Rasterized Disks



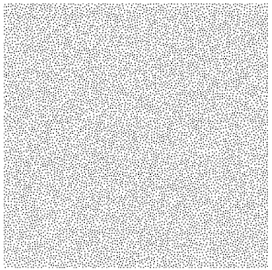


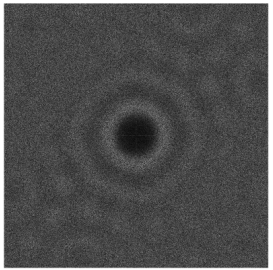


Pixel-centered **Subpixel-centered variation** **Anti-aliased average**

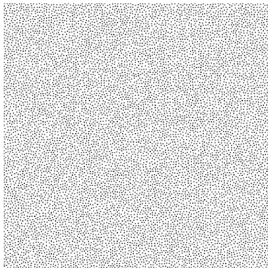
Spectrum Quality Comparison

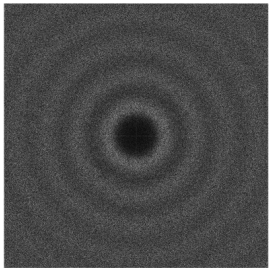
1024^2





[Gamito and Maddock ToG 2009]



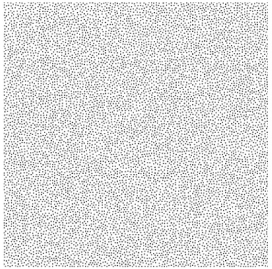


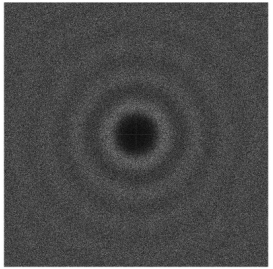
Samples

Power Spectrum
Periodogram

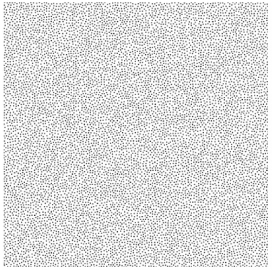
Spectrum Quality Comparison

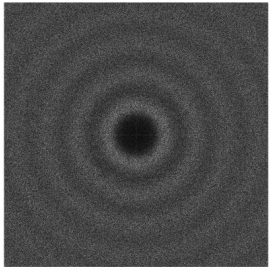
1024^2
w/ Subpixel centers





[Gamito and Maddock ToG 2009]



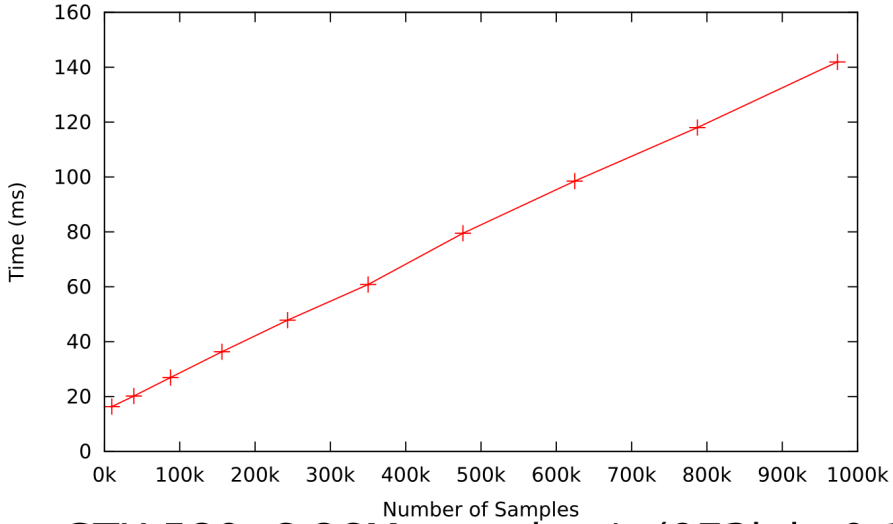


Samples

Power Spectrum
Periodogram

Performance

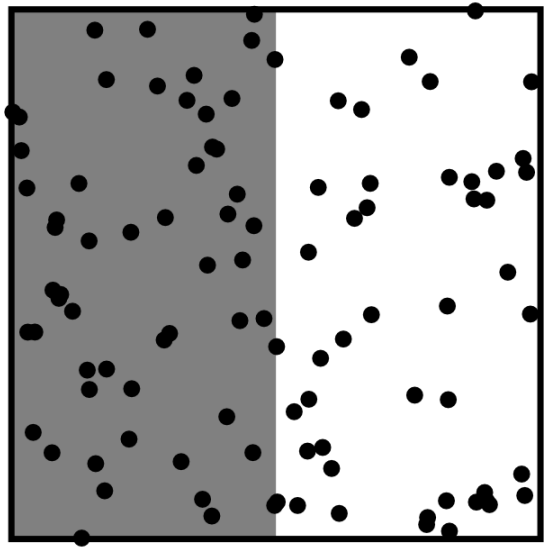
Uniform Sampling



Ours on GTX 580: 6.86M samples /s (973k in 0.14s)
Ours on GTX 460: 4M samples /s (787k in 0.19s)
[Ebeida et al. CGF 2012] on GTX 460: 1M samples /s

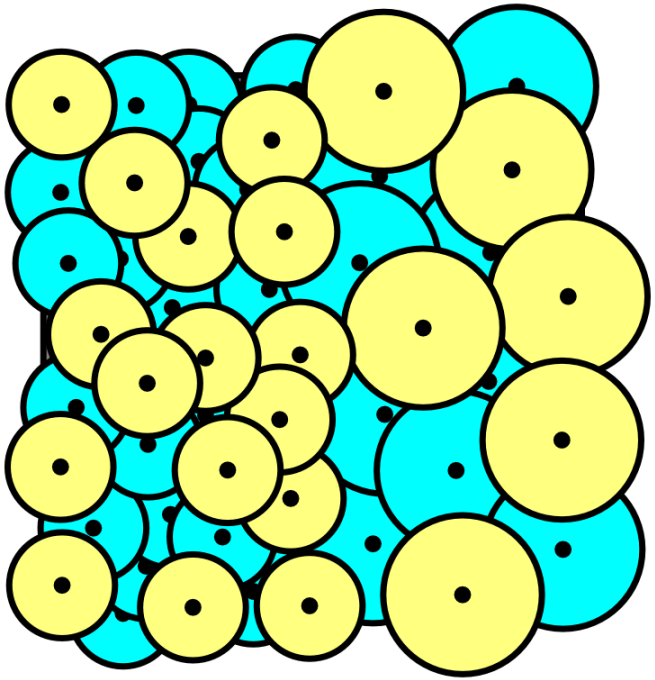
Importance Sampling

Random Darts

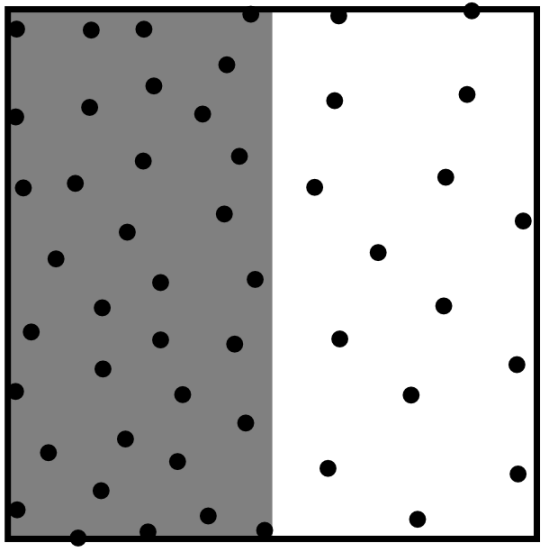


Sampling according to an importance

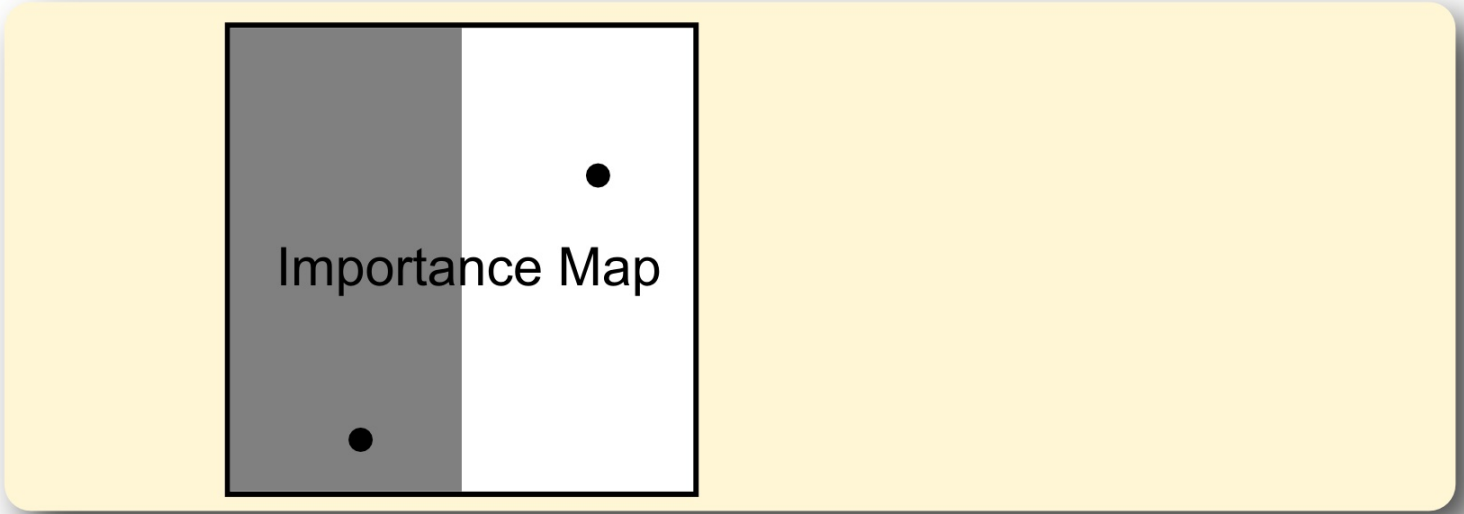
Vary the Disk Radii



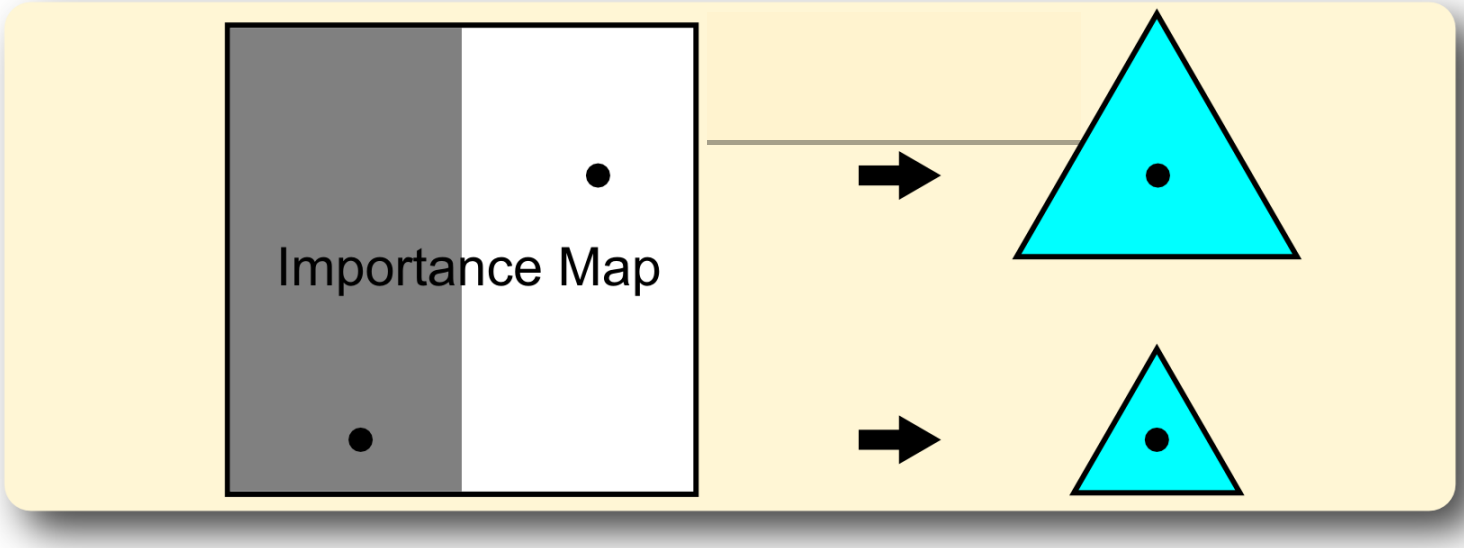
Importance Sampling



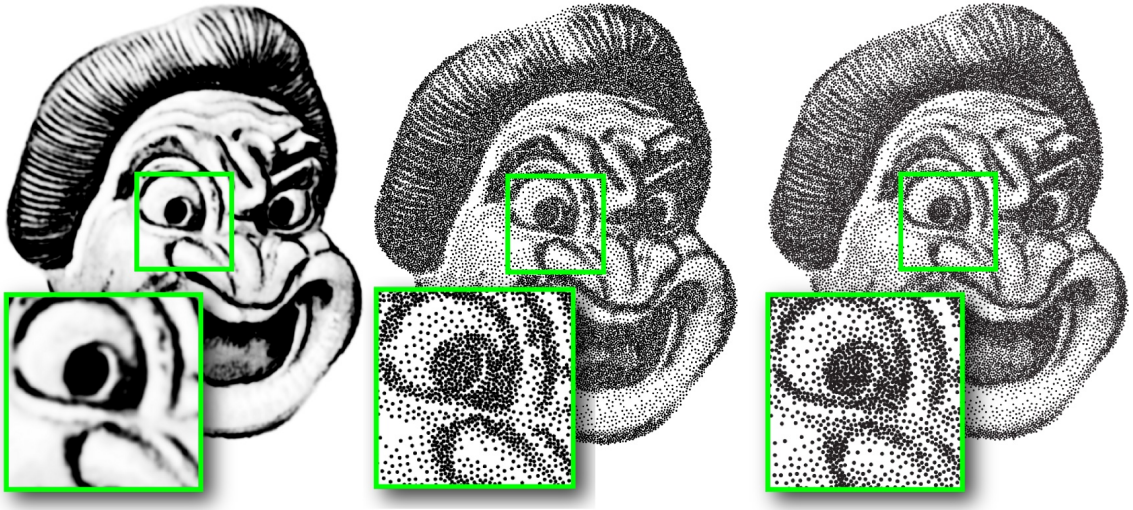
Importance Sampling: Geometry Shader



Importance Sampling: Geometry Shader



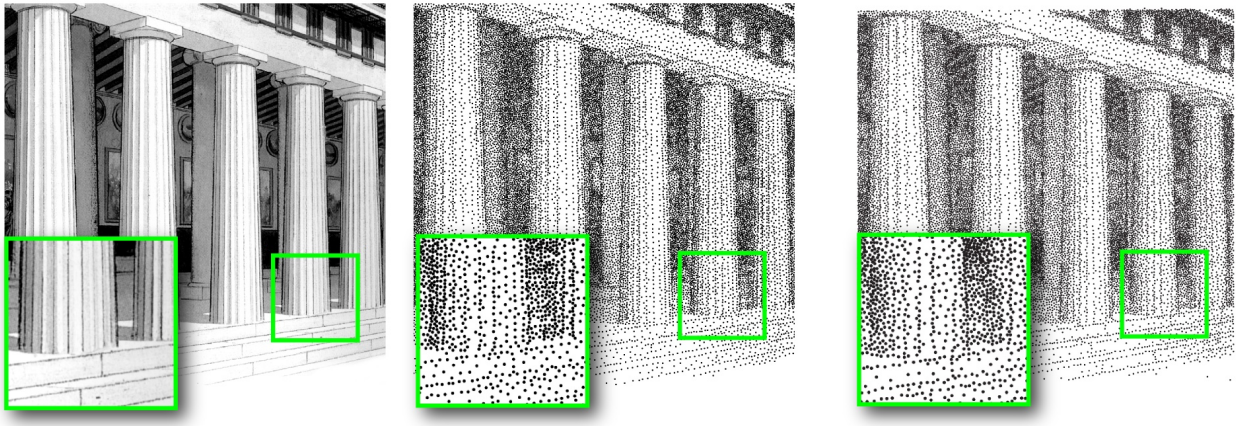
Results



0.053s on GTX580 24s on Core i7

[Kalantari & Sen CGF 2011]

Results



0.031s on GTX58025s on Core i7

[Kalantari & Sen CGF 2011]

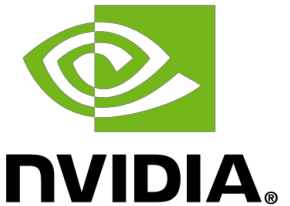
Conclusions

- Poisson-disk sampling with programmable graphics functions and hardware
- **Rasterization** and **Occlusion Queries** to reject conflicts
- **Subpixel-centered** disks to reduce bias
- Importance sampling by varying disk radii
- 6.8 M samples / sec on GTX 580

Acknowledgements

- Anonymous reviewers for the constructive comments
- National Science Foundation: CCF 05-41120, CMMI 08-35572, CNS 09-59979
- NVIDIA CUDA Center of Excellence Program

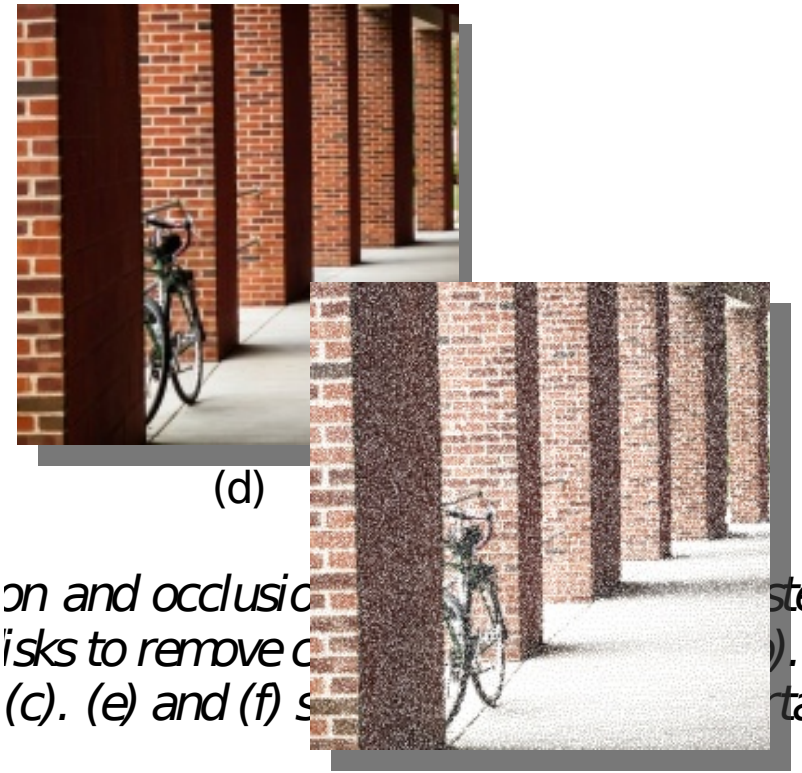
Thank you!



Questions ?

Please see our websites for the paper and software:

- Software
  - sf.net/p/pixelpie
- Cheuk Yiu Ip
  - www.cs.umd.edu/~ipcy
- GVIL Research Highlights
  - www.cs.umd.edu/gvil



(d)

on and occlusio … steriz
isks to remove c …). Thi
(c). (e) and (f) s … tance

graphics problem, the Poisson disk distri