Specialized Acceleration Structures for Ray-Tracing

Warren Hunt

Bill Mark

Forward: Flavor of Research

- Build is cheap (especially with scan, lazy and build from hierarchy)
  - Grid build and BVH refit are really cheap
- Many specialized acceleration structures are required for maximal rendering performance
  - Already true for game rendering algorithms

One System, Many (Layered) Structures

- Static structures for static objects
- Structures with static topology for dynamic objects with static topology
  - Sub-division surfaces
- Structures for collections of dynamic objects or particles
  - Also used in collision detection
- Structures for clusters of highly coherent rays
  - Primary, shadows, large flat reflectors
- Structures for GI (probably not ray-tracing)

Adaptive Acceleration Structures in Perspective Space

Warren Hunt

Bill Mark

Introduction

- Use the perspective transform to accelerate (near) common origin ray-tracing
  - Higher performance for DoF and Soft Shadows
- Use an adaptive acceleration structure
  - Avoids “Teapot in a stadium” problems
- Use a cost metric to improve the effectiveness of the acceleration structure
  - Derived from first principles

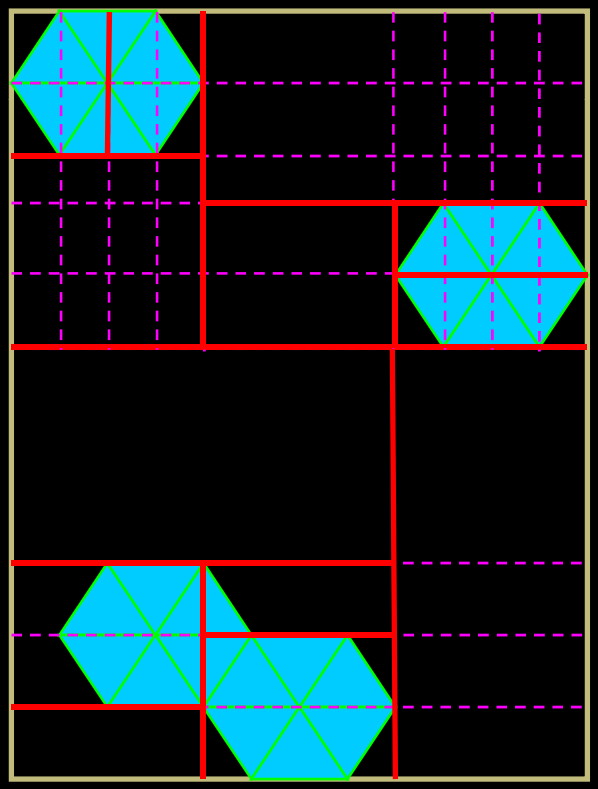
Motivation

- Perspective space holds promise for improving visibility performance for near-common origin rays
- Uniform grid isn’t adaptive enough to trace off-axis rays efficiently

Background: Adaptive Acceleration Structures

- Adaptive acceleration structure builders use cost metrics to improve quality
  - Present many candidate acceleration structures and pick the best (according to the metric)
- Commonly used in a top-down greedy way
  - E.g. The Surface Area Heuristic

kD-Tree Build With a Metric



The Surface Area Metric

- Most commonly used cost metric for building acceleration structures

- $Cost(s) = P(s_{left})Cost(s_{left}) + P(s_{right})Cost(s_{right})$
  - P(x) is the probability of a ray hitting x
  - Assumes uniform ray distribution
  - Assumes rays don't interact with objects

The SAM in Perspective Space

- Uniform distribution assumption for the SAM makes even less sense in perspective space

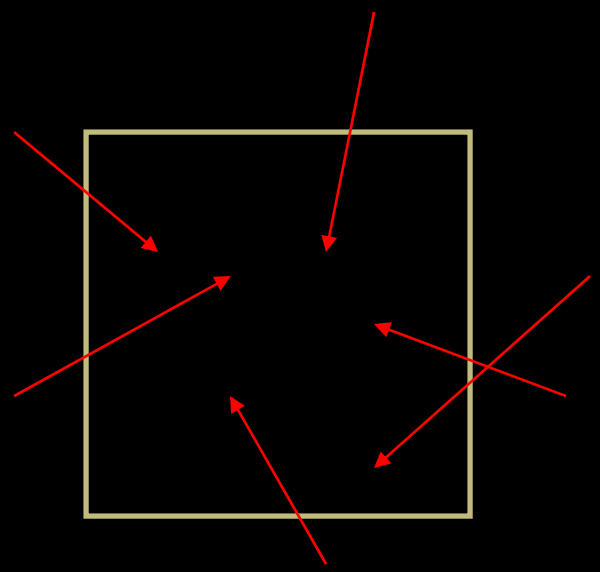
- Response: Start over from scratch
  - Derive a new cost metric in perspective space
  - We call the new metric the PSAM

The PSAM: New Assumptions

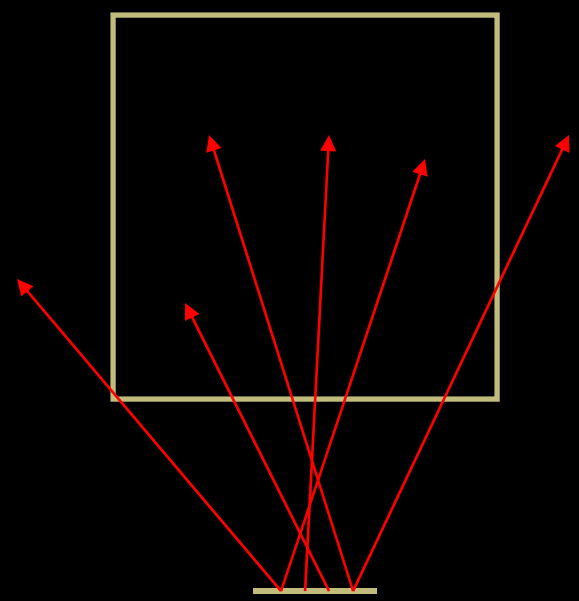
- Assume rays are originating from a rectangular source
  - Area camera or light
  - Dimensions $2A_x$ by $2A_y$

Ray Distribution Assumptions

SAM:



PSAM:

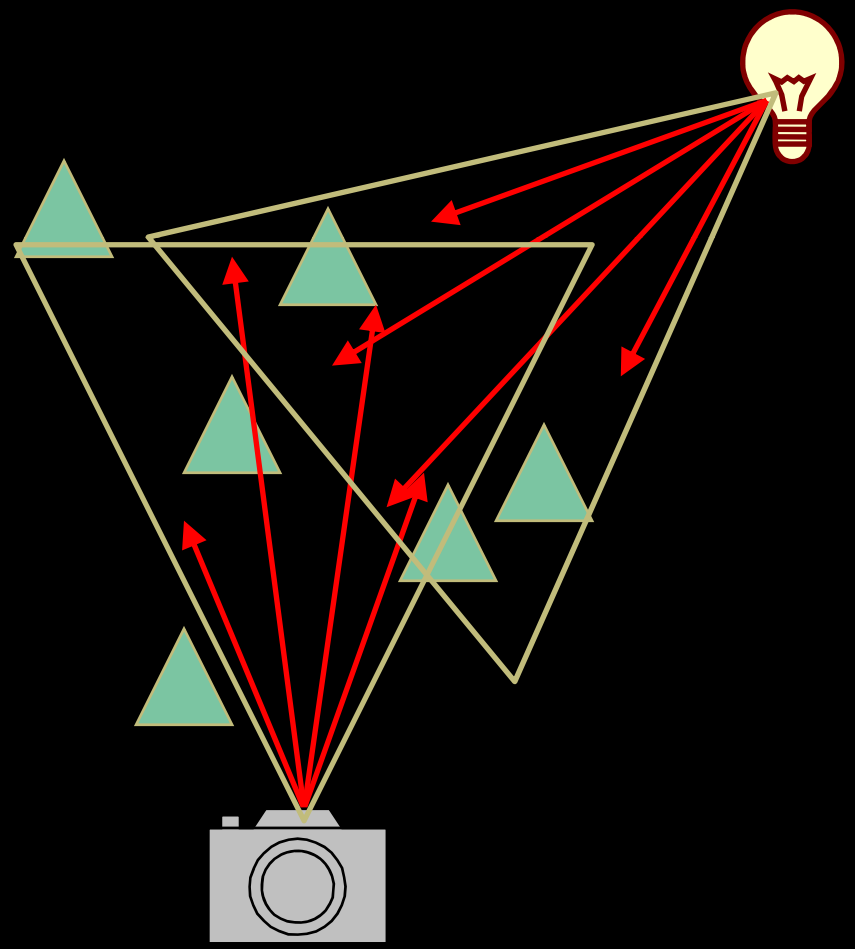


The PSAM: New Assumptions

- Assume rays are originating from a rectangular source
  - Area camera or light
  - Dimensions $2A_x$ by $2A_y$
- Use one perspective space structure per area camera or light

Many Structures:

- One for each Camera or Light
- Each structure makes unique assumptions about ray distribution



Deriving the PSAM: Setup

- Goal: Compute the probability of a ray intersecting an axis aligned voxel
  - Given a voxel and ray distribution
- Measure the number of rays entering a voxel
  - By integrating the intersection function over all rays
- The child/parent ratio of these measures determines the probability of ray intersection

Measure of Rays:

- Given area light with dimensions $2A_x$ by $2A_y$
- Given box with dimensions $\Delta x'$ by $\Delta y'$ by $\Delta z'$

- The measure (G) of rays intersecting the box:
- $G(box,A) = \Delta x'\Delta y' + (A_y/2)(\Delta x'\Delta z') + (A_z/2)(\Delta y'\Delta z')$
  - Derivation in the paper

Measure of Rays:

- Given area light with dimensions $2A_x$ by $2A_y$
- Given box with dimensions $\Delta x'$ by $\Delta y'$ by $\Delta z'$

- The measure (G) of rays intersecting the box:
- $G(box,A) = \Delta x'\Delta y' + (A_y/2)(\Delta x'\Delta z') + (A_z/2)(\Delta y'\Delta z')$
  - Derivation in the paper

The PSAM:

- $Cost(s,A) = P(s_{left})Cost(s_{left})+P(s_{right})Cost(s_{right})$

- Given G(box, A):
- $Cost(s,A) = (G(s_{left},A)/G(s,A))Cost(s_{left})$
  $+ (G(s_{right},A)/G(s,A))Cost(s_{right})$

  – Still assumes that rays don't interact with objects

The PSAM: Use

- Use the traditional SAH build algorithm with the PSAM in place of the SAM
- Clip the world to avoid the perspective singularity (cube map for point lights)
- Use back-face culling in all cases

Results

- Perspective space and the PSAH improves performance by 20%-40%
  - For soft shadows and DoF
- In many cases build and render in perspective space costs less than just render in world space
  - Back-face culling reduces build cost by ~50%

Results Subset Table (1 core)

Scene	Type	ISect	Traverse	Render	Total
FForest	Eye/SAH	9.369 M	23.15 M	1.16 s	8.37 s

Results Subset Table (1 core)

Scene	Type	ISect	Traverse	Render	Total
FForest	Eye/SAH	9.369 M	23.15 M	1.16 s	8.37 s
FForest	Eye/PSAH	5.482 M	14.49 M	0.71 s	1.29 s

Results Subset Table (1 core)

Scene	Type	ISect	Traverse	Render	Total
FForest	Eye/SAH	9.369 M	23.15 M	1.16 s	8.37 s
FForest	Eye/PSAH	5.482 M	14.49 M	0.71 s	1.29 s
Bunny69	Eye/SAH	2.838 M	10.43 M	0.492 s	2.62 s
Bunny69	Eye/PSAH	1.515 M	5.981 M	0.272 s	0.43 s

Results Subset Table (1 core)

Scene	Type	ISect	Traverse	Render	Total
FForest	Eye/SAH	9.369 M	23.15 M	1.16 s	8.37 s
FForest	Eye/PSAH	5.482 M	14.49 M	0.71 s	1.29 s
Bunny69	Eye/SAH	2.838 M	10.43 M	0.492 s	2.62 s
Bunny69	Eye/PSAH	1.515 M	5.981 M	0.272 s	0.43 s
FForest	DoF/SAH	148.0 M	369.6 M	18.5 s	25.8 s
FForest	DoF/PSAH	92.60 M	314.7 M	14.5 s	15.1 s
Bunny69	DoF/SAH	81.19 M	255.1 M	12.7 s	14.8 s
Bunny69	DoF/PSAH	46.29 M	141.1 M	7.8 s	8.0 s

Notes:

- Build Only Gets Cheaper!
  - Build times are not using lazy build
  - Build times are not using build from hierarchy
- Hard/Soft shadows behave much like primary/DoF rays (elided for time/space)

Heuristic Compare on FForest

Heuristic	Build	Isects	Traverse	Render	Total
Eye/Median	0.209 s	63.85 M	48.76 M	4.16 s	4.38 s
Eye/SAH	0.449 s	11.46 M	31.64 M	1.43 s	1.88 s
Eye/PSAH	0.581 s	5.482 M	14.49 M	0.71 s	1.29 s

Heuristic Compare on FForest

Heuristic	Build	Isects	Traverse	Render	Total
Eye/Median	0.209 s	63.85 M	48.76 M	4.16 s	4.38 s
Eye/SAH	0.449 s	11.46 M	31.64 M	1.43 s	1.88 s
Eye/PSAH	0.581 s	5.482 M	14.49 M	0.71 s	1.29 s
DoF/Median	0.208 s	999.1 M	794.4 M	65.8 s	66.0 s
DoF/SAH	0.472 s	177.2 M	519.5 M	23.3 s	23.8 s
DoF/PSAH	0.580 s	92.60 M	314.7 M	14.5 s	15.1 s

Conclusions

- The perspective transform can produce more effective (ray-specialized) acceleration structures for off-axis rays
- A new cost metric produces better results in perspective space

Section Conclusions

Section Conclusions

- Many specialized acceleration structures are required for maximal rendering performance
  - Already true for game rendering algorithms
- Build is cheap (especially with scan and lazy build from hierarchy)
  - Grid build and BVH refit are really cheap