SVGPU: Real Time 3D Rendering to Vector Graphics Formats

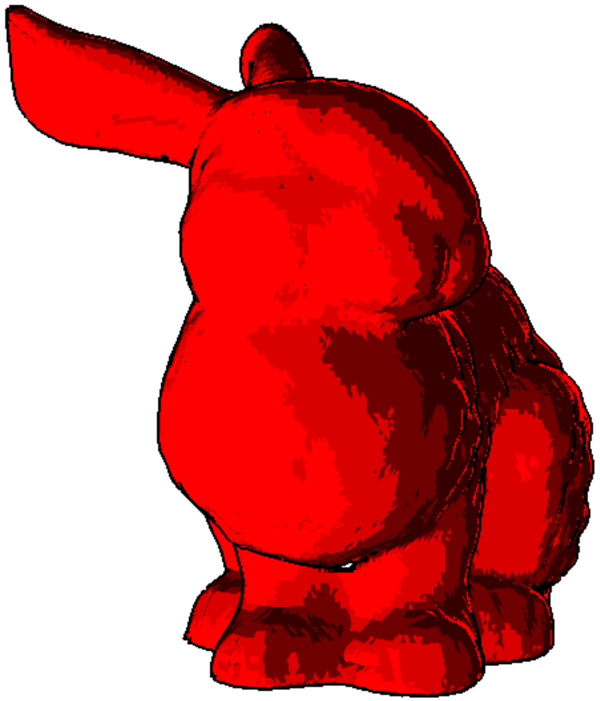
APOLLO I. ELLIS AND WARREN HUNT AND JOHN C. HART

UIUC *OCULUS* *UIUC*

AIELLIS2@ILLINOIS.EDU, WARREN.HUNT@GMAIL.COM, JCH@ILLINOIS.EDU

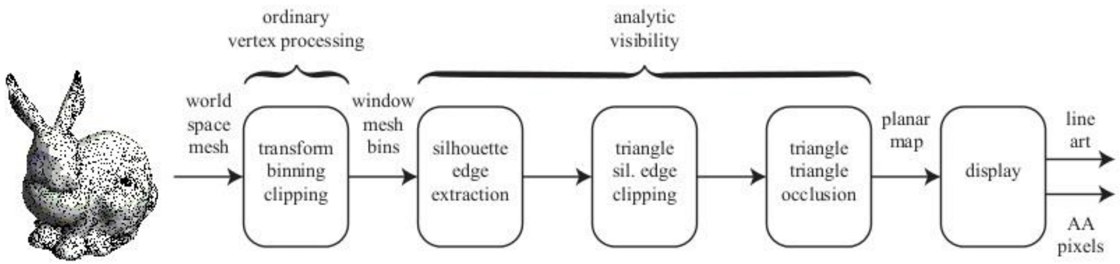
Intro: What if we could output triangles instead of pixels?

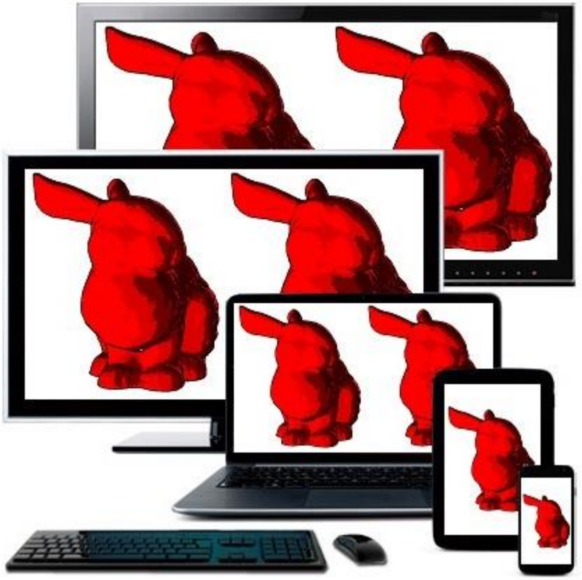
- Benefits of vector graphics representations
  - Vertex correspondences
  - NPR effects
  - Antialiasing
  - Resolution independent compression
- SVGPU (Scalable Vector GPU)
- Outputs a planar map in real time
- Currently built in Cuda



SVGPU is simple and fast

- Spatial hashing extracts silhouettes
- Clipping clips triangles to silhouettes
- Occlusion discards hidden surfaces





Previous approaches still short of real time

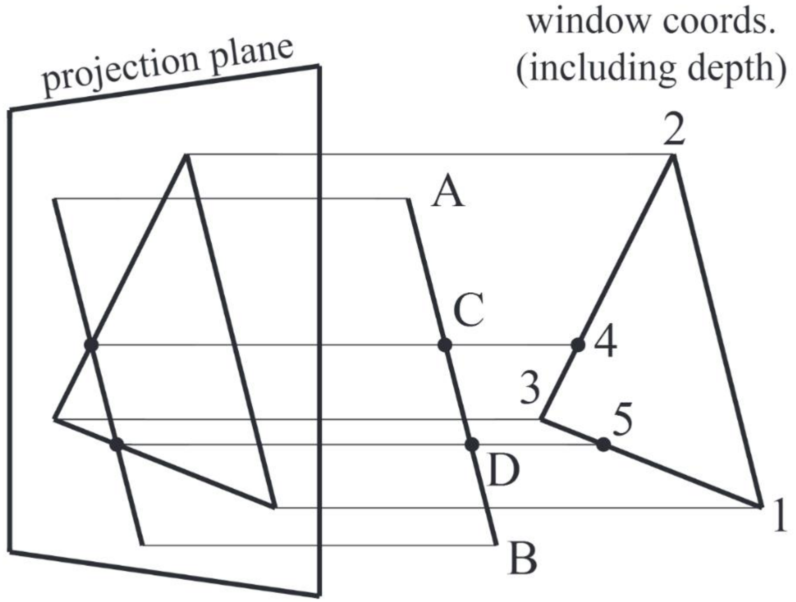
- Offline systems have been proposed
- Largely CPU programs focusing on quality
  [Winkenbach and Salesin][Stroila et al.][Karsch and Hart][Eisemann et al.]
- The GPU has been used for fast analytic visibility [Auzinger et al.]
- No one has achieved real time
- We build on Robert's Algorithm [Roberts, L. 1963]
- Yes, from the 60's!

First we need to extract silhouettes

- Hash all edges into a spatial hash table
- Minimal Collisions < 5
  - Rarely more than 2
- Second pass uses active keys checking for front-back facing pairs
- If a pair is found we bin the edge
- Reasonably quick, < 2ms

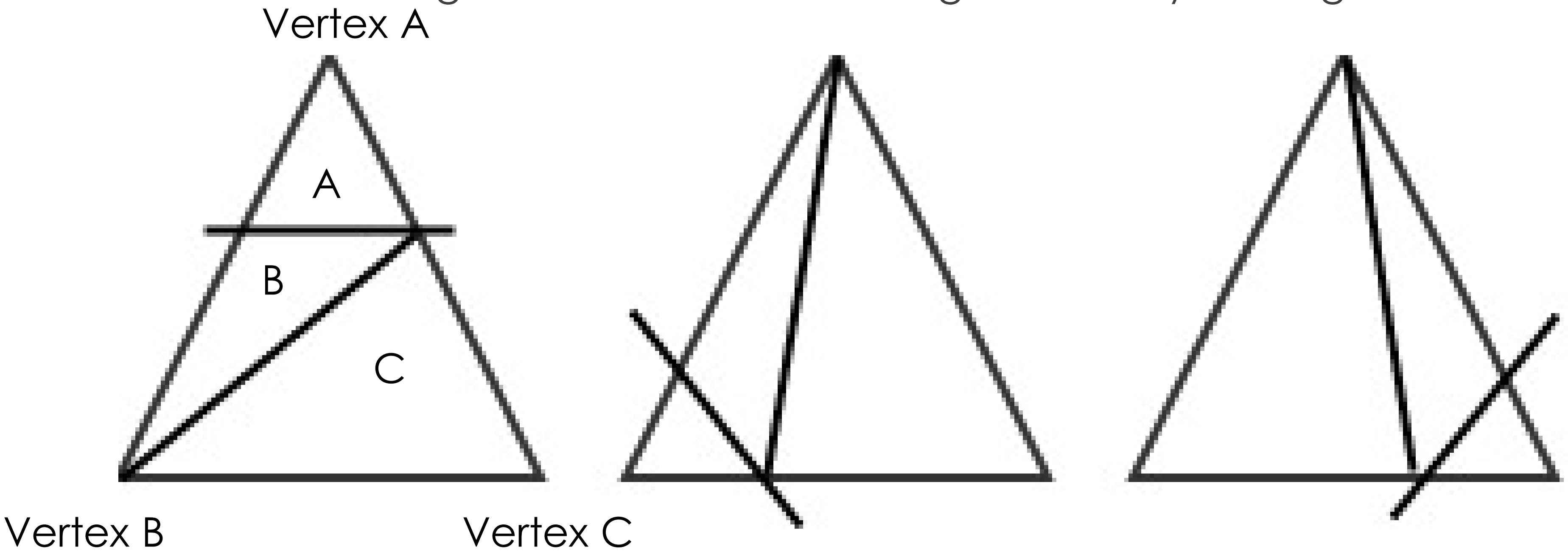
Clipping

- Insight: When triangles are clipped to silhouette edges, the complexity of clipping is reduced



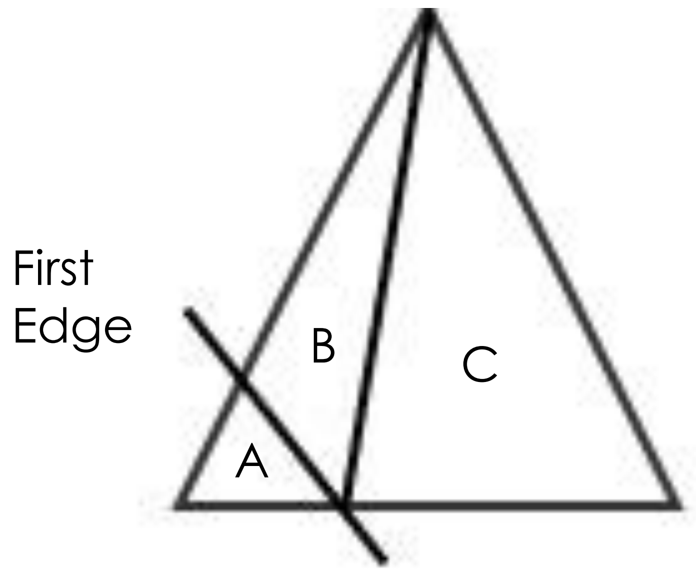
Clipping: There are basically 3 cases

- The side of the edge that is occluded changes, actually making 6 cases



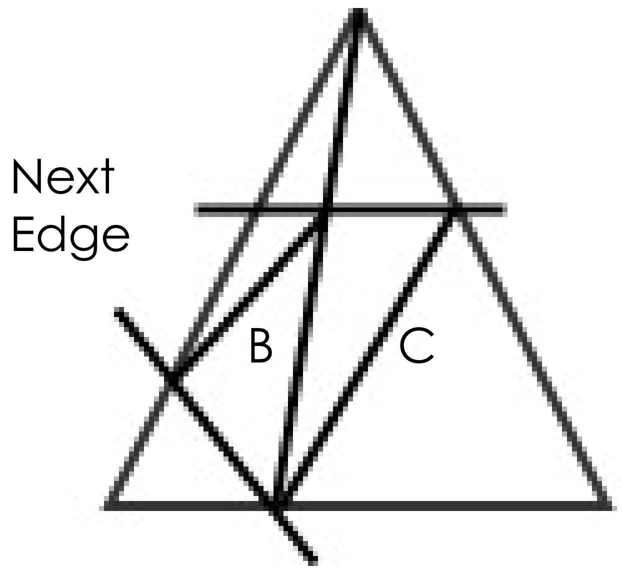
Clipping: After round 1 we have three triangles

- Clipping the original triangle creates A B and C



Clipping: In round 2 we generate 6 new triangles

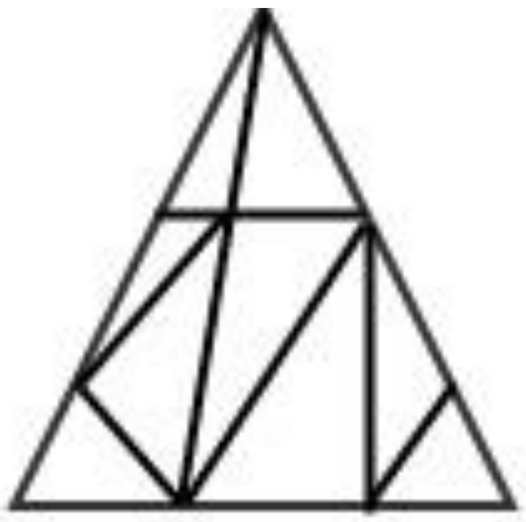
- B and C are clipped against next edge



Clipping: Round 3 generates 3 new triangles

- Only one triangle is clipped against the last edge

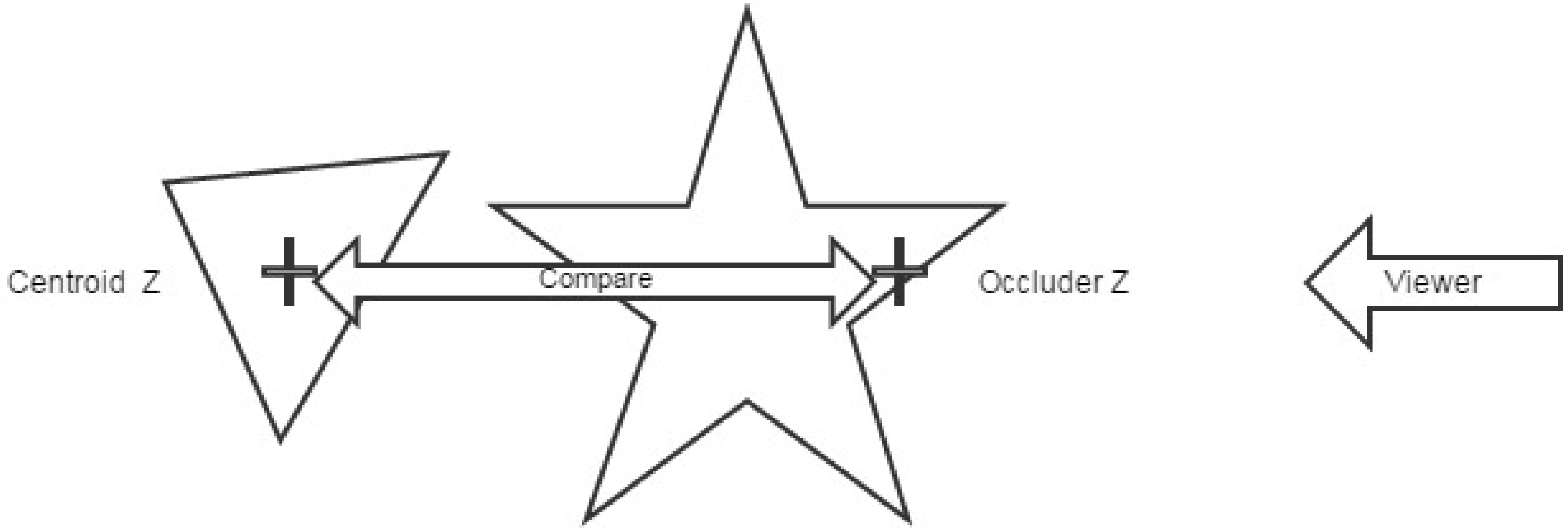
Clipping: Result, 9 triangles



Occlusion Phase Discards Rapidly

- Insight: Silhouette clipping completely resolves partial visibility cases.
- Triangles are either fully occluded or fully visible
- Suffices to check overlap in X Y and “is closer”
- Compute hyper-plane separation test for overlap
- Then Z test the occludee centroid against the occluder
- Do see why this Z test will holds?

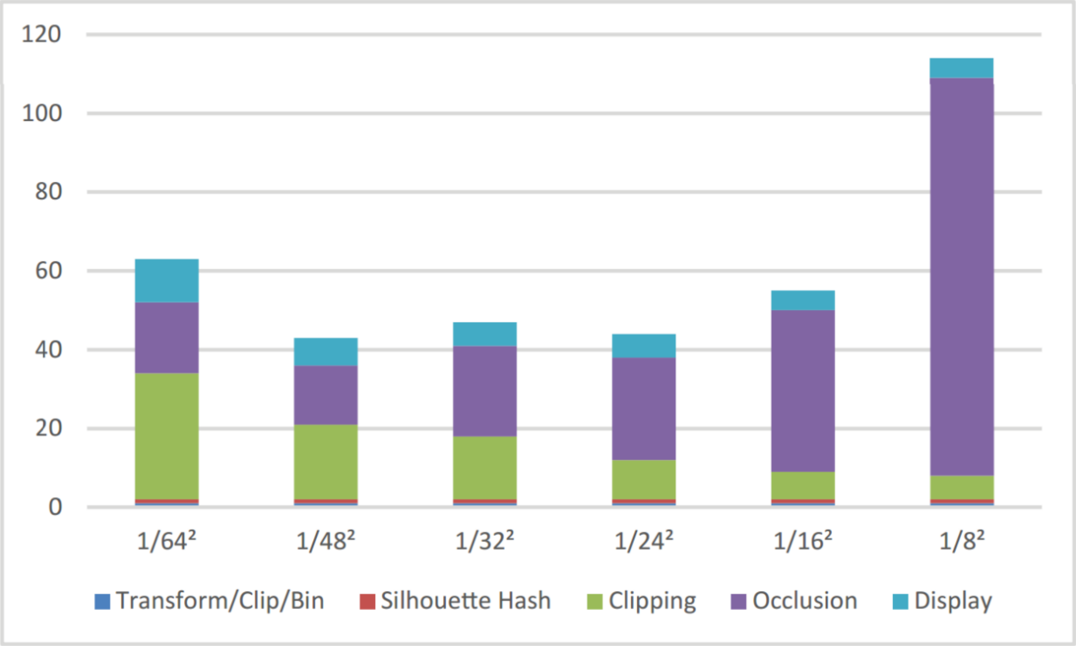
Z Centroid will eventually be occluded



Initial results were sub real time

Model	in → out	bins (non-∅)	max	ave. $\frac{\Delta}{\text{bin}}$	t
Bunny	69K → 280K	2,304 (33%)	698	139	41
Dragon	202K → 950K	2,304 (19%)	1,740	555	222
Armadillo	212K → 1.2M	1,152 (44%)	2,076	356	256
Buddha	293K → 1M	2,304 (43%)	712	200	140

Bin Tuning and Re-binning



- Bin sizes help and hurt different phases
- Course binning is good at first (64 bins)
- Occlusion does better with more (2048)
- The cost to change bin resolution (Rebinning) is small.
- ~ 2ms re-bin for a 14ms improvement.

We are still tuning and improving!

- Currently 75Hz for Bunny
- 30Hz for Armadillo (slowest scene)
- 9X performance over state of the art!
- Optimizing memory usage (e.g. empty bin storage)
- Reducing tessellation
- Applications (Compression, Antialiasing…)

Questions?

References

- ROBERTS, L. 1963. Machine perception of three-dimensional solids. Tech. Rep. TR 315, Lincoln Laboratory, MIT.
- WINKENBACH, G., AND SALESIN, D. H. 1994. Computer generated pen-and-ink illustration. Proc. SIGGRAPH, 91–100.
- STROILA, M., EISEMANN, E., AND HART, J. 2008. Clip art rendering of smooth isosurfaces. IEEE TVCG 14, 1, 135–145.
- KARSCH, K., AND HART, J. C. 2011. Snaxels on a plane. Proc. NPAR, 35–42.
- EISEMANN, E., PARIS, S., AND DURAND, F. 2009. A visibility algorithm for converting 3d meshes into editable 2d vector graphics. ACM TOG 28, 3, 83:1–83:8.
- AUZINGER, T., WIMMER, M., AND JESCHKE, S. 2013. Analytic visibility on the gpu. Computer Graphics Forum (Proc. Eurographics) 32, 2 (May), 409–418.