CS 563 Advanced Topics in Computer Graphics
QSplat

by Matt Maziarz

Outline

- Previous work in area
- Background
- Overview
- In-depth look
- File structure
- Performance
- Future

Before QSplat

- Point Rendering
  - To save on setup and rasterization costs compared to triangles
  - For large amount of geometric data
  - Particle systems
- Works best if each voxel is the size of a pixel
  - Cline has *dividing cubes* algorithm for us when voxels bigger than pixels
  - Sawn proposed an algorithm for voxels smaller than a pixel
  - Both are used to make results correctly antialiased

Before QSplat cont.

- Visibility culling
  - Frustrum
  - Backface
  - Occlusion

Before QSplat cont.

- Level of Detail Control
  - Static / pre computed
  - Multi-resolution analysis
    - Uses a base mesh and wavelets
  - Progressive mesh
    - Base mesh with series of vertex split operation
    - ROAM

Background

- 3D scanning
  - New advances mean objects can create a mesh with hundreds of millions of polygons
- Current computing
  - Not fast enough to display meshes of this size in real-time
  - Traditional simplification methods and progressive display algorithms take to much time and storage space

Background cont.

- Traditional methods
  - Focus on individual edges and vertices, high cost per vertex for these methods
- Scanned data
  - Large number of vertices and locations often imperfect due to noise
- Since tradition methods spend lots of effort on vertices so need new methods with a low cost per vertex.
- QSplat created for 3D scanning
  - Does not use any connectivity information, which is only useful for depth testing

Overview

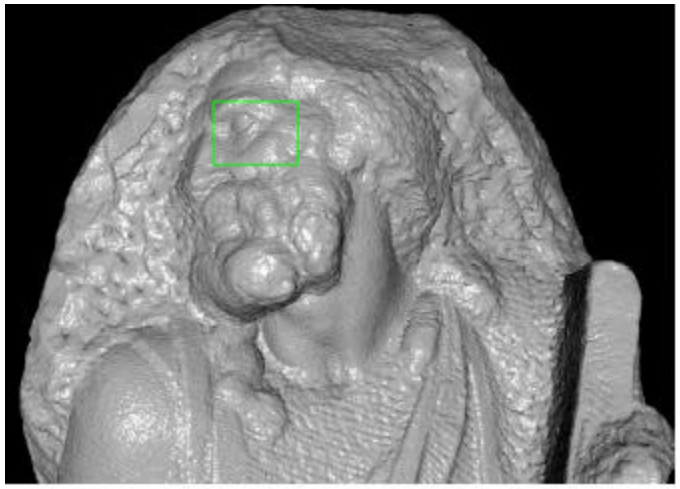
- QSplat
  - Uses hierarchy of bounding spheres
    - For visibility culling
    - Level of detail control
    - Rendering
  - Each node contains
    - Sphere’s center
    - Radius
    - Normal
    - Width of the normal cone
  - Hierarchy is preprocessed and saved on disk

Overview cont.

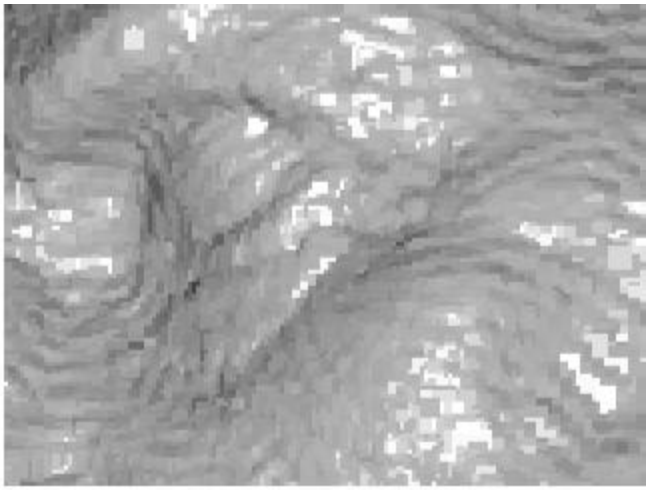
- Traverse hierarchy algorithm

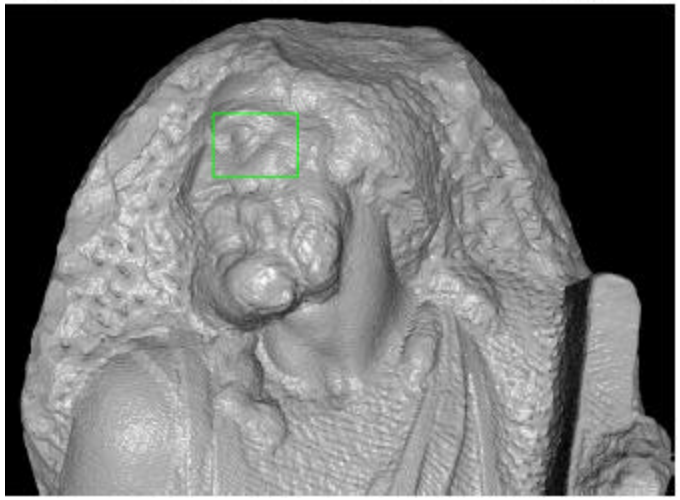
```
TraverseHierarchy(node)
{
  if(node not visibile)
    skip this branch of tree
  else if (node is a leaf node)
    draw splat
  else if (benefit of recusing further is too low)
    draw splat
Else
  for each child of node
    TraverseHierarchy(child)
```

Overview cont.



5-pixel cutoff
1,017,149 points
722 ms





1-pixel cutoff
14,835,967 points
8308 ms

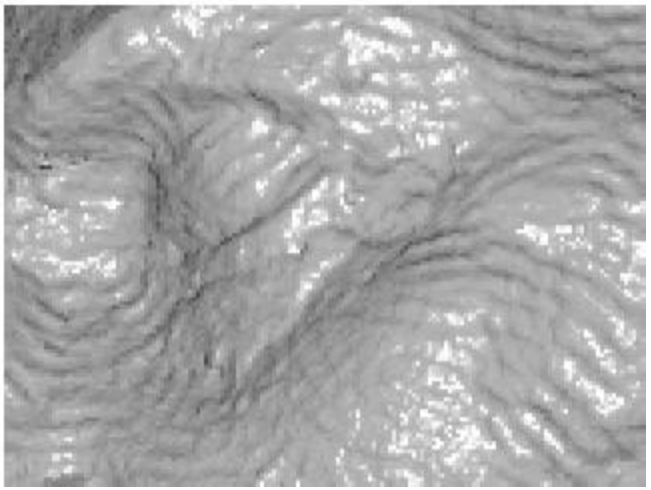
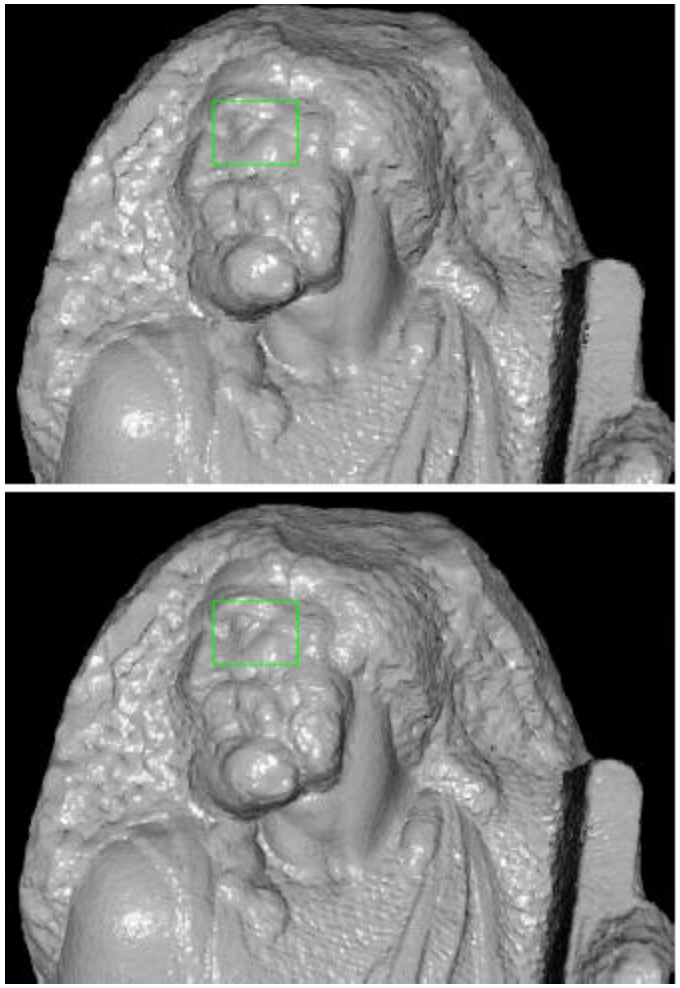
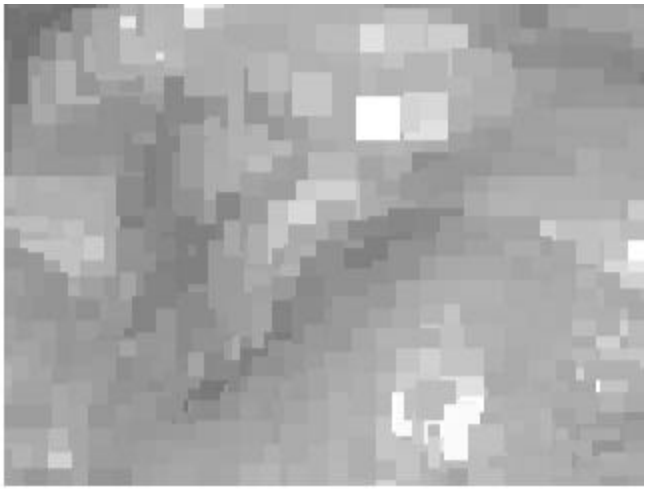


Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Overview Cont.



15-pixel cutoff
130,712 points
132 ms



10-pixel cutoff
259,975 points
215 ms

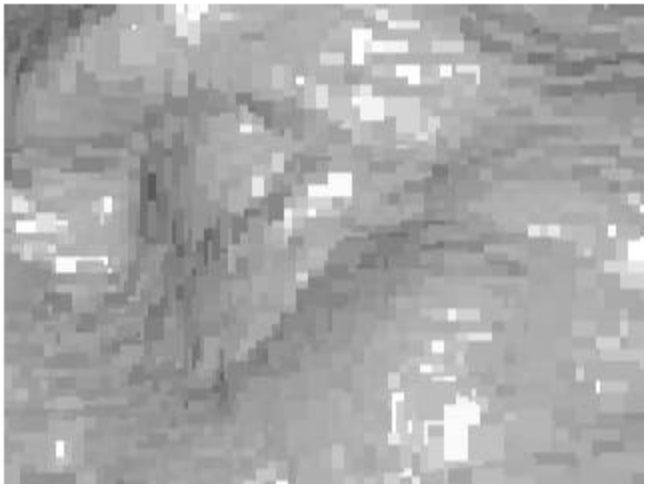


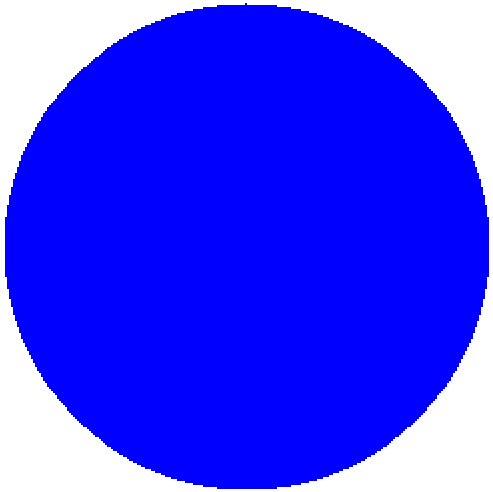
Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Overview cont.

- Preprocessing

```
BuildTree(vertices[begin..end])
{
  if (begin == end)
    return Sphere(vertices[begin])
  else
   midpoint = PartitionAlongLongestAxis(verts)
    leftsubtree = BuildTree(verts[begin..midpoint)
    rightsubtree = BuildTree(verts[midpoint..end])
    return BoudingSphere(leftsubtree, rightsubtree)
}
```

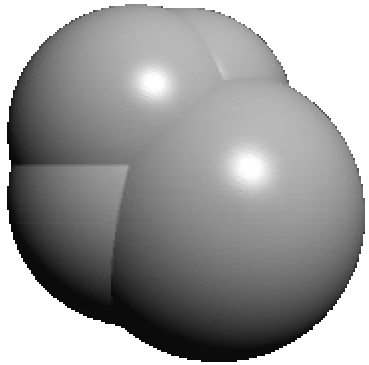
Overview cont.

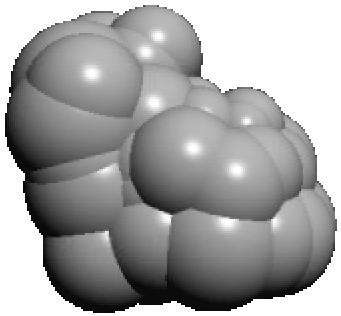


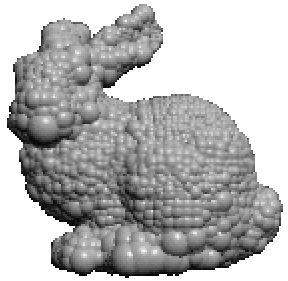
Overview cont.

- Pre processing
  - Better with mesh, can be done with Cloud
  - Mesh
    - Easy to computer normals
    - Makes sure no holes
      - Each vertex in the end is a leaf node
      - If two vertices connected by an edge the spheres for each vertices will be made big enough to touch
      - Current algorithm makes sure the size of sphere is equal to the max size of a bounding sphere of all triangles that touch that vertex
  - When recursion reaches single node it creates a sphere that is at the center position of the vertex

Bounding spheres







- Image courtesy of Prof. Ioannis Stamos

Overview cont.

- Nodes
  - Each node has 0, 2, 3, 4 children
    - Never will have one child, if it does, then it is same as parent and therefore redundant
  - 4 Children used so average branching factor would be about 3.5
  - The higher the branching factor the less internal nodes which saves space

In-Depth Visibility Culling

- Frustum Culling
- Backface culling
  - Test if each normal or normal cone is facing away
  - If entire normal cone facing towards view marks children to not perform this test
- No occlusion culling
  - No real benefit for most scanned in structures
  - Useful if scenes more complex with multiple objects

In-Depth When to Recurse

- Threshold – when the area of the sphere is projected if it exceeds this threshold
- FPS – The cutoff or threshold is adjusted from frame to frame depending on the users selected frame rate
  - A predictive algorithm could be used to make the difference of rendering times from frame to frame smaller but it has not yet been implemented
- Normals
  - Close to the silhoutte recurse more
  - Normal cone is very wide recurse more
  - Above could be used but right now are not
- No need to smooth between LODs because the changes are not very significant

In-Depth Drawing Splats

- Leaf node
  - Draw Splat
- Threshold met
  - Draw Splat

- Splat
  - Size – based on diameter
  - Color – Lighting cacluations plus stored color
  - Occlusion – Done via the Z-buffer

File Structure

- Position and Radius
  - 13 bits
  - The radius is with respect to the parent
  - Range from 1/13 to 13/13 for
  - Position (X Y Z) offset is multiple of 1/13 of parent's diameter
  - Not all 13^4 offsets are valid, only 7621 are possible so can be stored in 13 bits using a lookup table
  - Average error for position is 0.04
  - Average error for radius is .15 because it must include the 0.04 and also takes the ceiling resulting radius to make sure no holes

File Structure cont.

- Tree Structure
  - 3 bits
  - Number of children 0,2,3,4 – 2 bits
  - All children are leaf nodes – 1 bit
- Overall Structure is Breadth first
  - This allows an image to be displayed on screen before all data is read in

File Structure cont.

- Normals
  - 14 bits
  - Normals corrospond to a 52 x 52 grid each with 6 faces
  - 52 x 52 = 16224 possible normals
  - A lookup table is used
  - Average error of 0.01 radian
  - Banding artifacts possible at specular highlights near areas with little curve
    - To fix make normals like radius and position being ralative to the normal of the parent spehe. This adds a good amount of time which could not be sacraficed.

File Structure cont.

- Norlam cones
  - 2 bits
  - Four possible values sin of half angle equals
    - 1/16
    - 4/16
    - 9/16
    - 16/16
  - This set discards over 90 percent of nodes that would be culled from true backface culling
  - Again could also be with respect to parent but increases computation time

File Structure cont.

- Colors (Optional)
  - 16 bits
  - 5 bits Red
  - 6 bits Green
  - 5 bits blue

Performance

- Rendering
  - Majority of time in inner loop
    - Calculating each position and radius
    - Visibility culling
    - Whether to traverse or draw
  - At lower levels of tree does not perform exact division
  - 1.5 to 2.5 million points per second on SGI Onyx2
    - This is after reading everything from disk
    - Depends on caching, and how much data culled

Performance cont.

- QSplats display rate is equivalent
  - 480 thousand polygons a second for progressive meshes
  - 180 thousand polygons a second for ROAM
- 200 – 300 thousand points a second for interactive display
  - No frame to frame coherence, so no caching of likely points for next frame
- When not interactive it recurses further to make image better

Performance cont.

- Low-end Hardware
- 366 mhz Pentium II w/ 128 memeory
  - No 3D acceleration
- QSplat
  - Window size = 500 x 500
  - Traverse 250 – 400 nodes a frame
  - 50 to 70 splats a frame
  - Fill rate of 40 million pixels a frame
  - Frame rate of 5hz | 5 Frames a second
  - Fully usable

Performance cont.

- Pre-processing
  - Progressive mesh with 200 thousand vertices
    - 10 hours
  - Hierarchical dynamic simplification 281 thousand vertices
    - 121 seconds
  - QSplats 2000 thousand vertices
    - Less than 5 seconds

Performance cont.

David's head, 1mm, color

David, 2mm

St. Matthew, 0.25mm

Typical performance

	Interactive	Static	Interactive	Static	Interactive	Static
Traverse tree	22 ms	448 ms	30 ms	392 ms	27 ms	951 ms
Compute position and size	19 ms	126 ms	30 ms	307 ms	31 ms	879 ms
Frustum culling	1 ms	4 ms	1 ms	3 ms	1 ms	3 ms
Backface culling	1 ms	22 ms	2 ms	25 ms	1 ms	35 ms
Draw splats	77 ms	364 ms	46 ms	324 ms	50 ms	1281 ms
Total rendering time	**120 ms**	**838 ms**	**109 ms**	**1051 ms**	**110 ms**	**3149 ms**
Points rendered	125,183	931,093	267,542	2,026,496	263,915	8,110,665

Preprocessing statistics

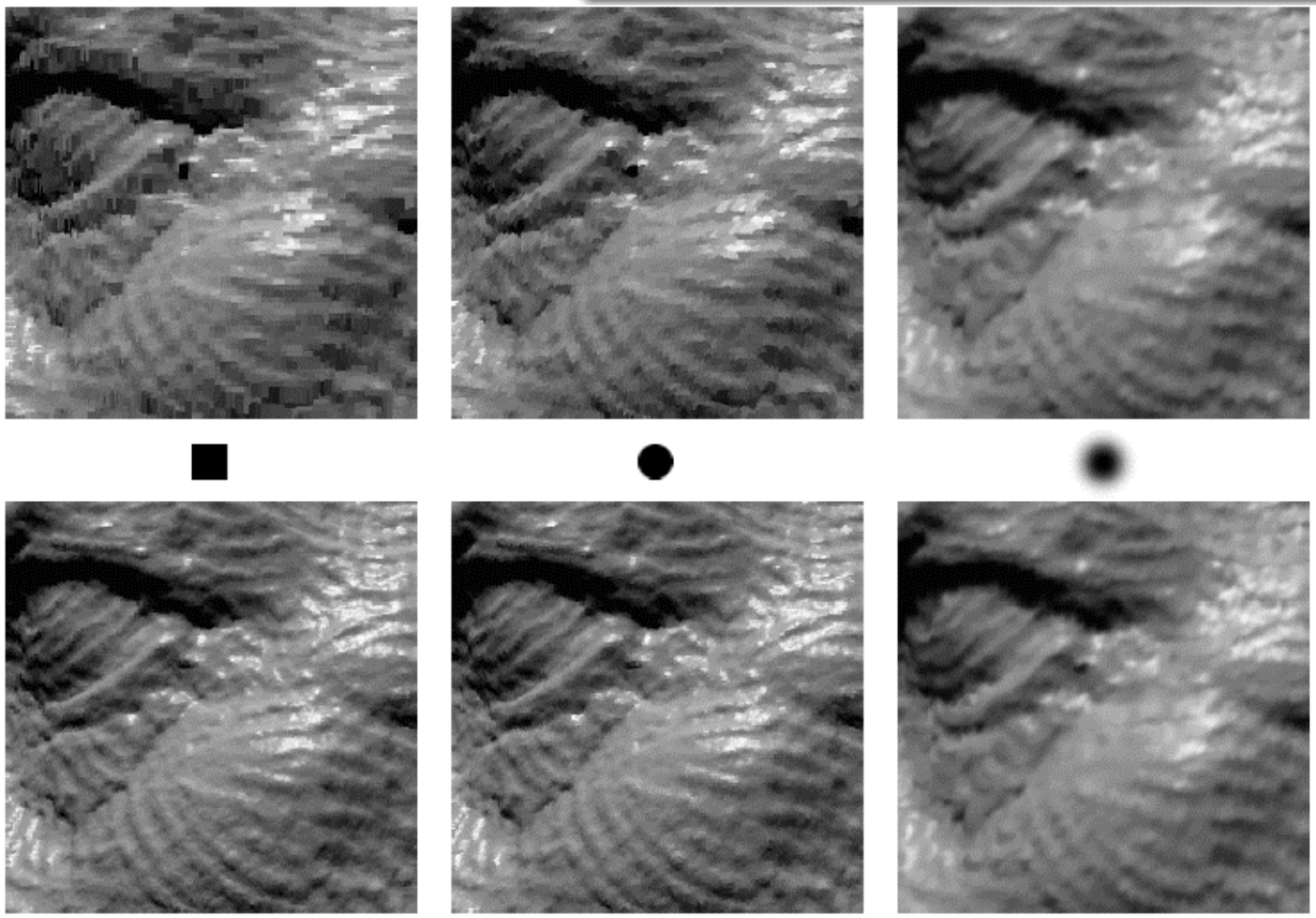
Input points (= leaf nodes)	2,000,651	4,251,890	127,072,827
Interior nodes	974,114	2,068,752	50,285,122
Bytes per node	6	4	4
Space taken by pointers	1.3 MB	2.7 MB	84 MB
Total file size	**18 MB**	**27 MB**	**761 MB**
Preprocessing time	**0.7 min**	**1.4 min**	**59 min**

Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Performance cont.

- Shapes of splats
  - Square / OpenGL point
    - It's the fastest, but a square and not AA
  - Circle / textured polygon
    - OpenGL is optimized for triangles
  - Gaussian splat / fuzzy splat
    - Color fades when farther from center, starts at ½ radius
    - Can be problematic with no Z-Buffer
  - Elliptic splats
    - Ratio of major minor axis which is N dot V (N and V normalized)
    - Ratio cut off at 10 to make sure hole free
    - Still possible holes on silhouette

Performance cont.



Top is same threshold(20 pixels), Bottom is same render time

Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Performance cont.

- Ellipse Splats

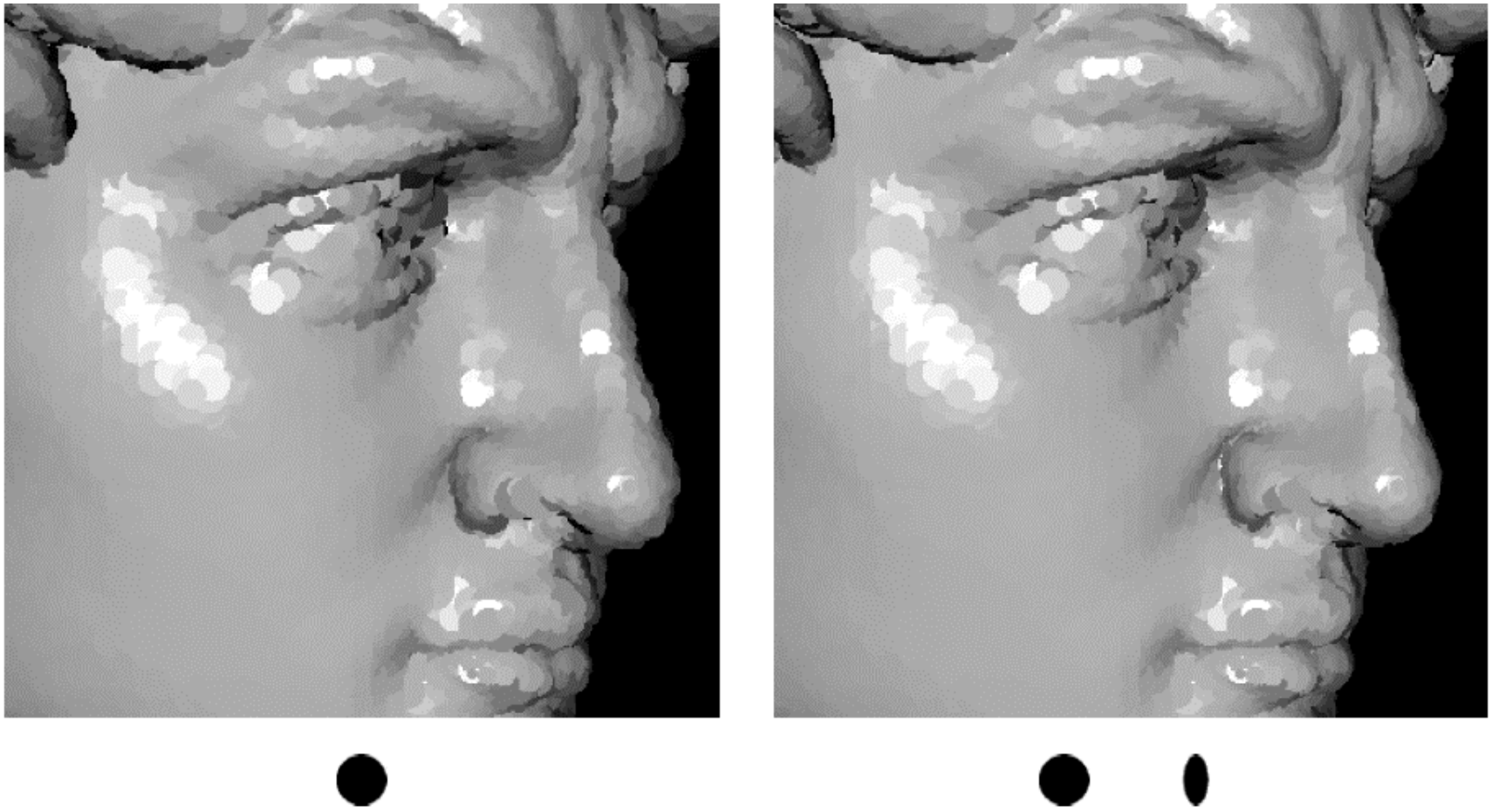


Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Performance cont.

- QSplat vs Polgyons

QSplat	Polygons
Good for models with detail everywhere	Good for models with large flat areas or subtle curves
High per pixel, but less slowdown when with no 3D hardware	Highly optimized with 3D hardware
Fast pre-processing	Decimation or LOD creation usually expensive

Performance cont.

(a)

Points

(b)

Polygons – same number of primitives as (a)
Same rendering time as (a)

(c)

Polygons – same number of vertices as (a)
Twice the rendering time of (a)

Image from *QSplat: A Multiresolution Point Rendering System for Large Meshes*

Future

- Set up in Museum
  - Simplified UI, only translate, rotate, and re-light
  - Users still sometimes got camera in weird positions
  - Generally well accepted by patrons

Future cont.

- If speed is more important
  - Remove the compression for position and radius storage
  - Store information as floats instead of doubles
  - You can parallelize rendering so portions of tree on multiple processors
  - Can even parallelize the pre-processing but doesn’t help for rendering

Future

- Huffman coding
  - Lossless, can use to encode file for small storage
  - Costs is same as frequency
  - More the character is used, the small codeword it has

Input	Alphabet	a	b	c	d	e	f	g	h	i	
	Costs	10	15	5	15	20	5	15	30	5	
Output	Codewords	0001	001	00001	100	11	000000	101	01	000001	
	Weighted path length	10*3	15*3	5*5	15*3	20*3	5*5	15*3	30*2	5*4	= 355

Image from http://en.wikipedia.org/wiki/Huffman_coding

Future

- Use Huffman coding for either storage or transmission over low bandwidth
- Network uses of QSplat
  - Transmit only parts of tree needed, can quickly render information it already has
  - Uses queues to make sure specific data needed gets sent
  - Chunks of data will not be sent if view is changed and data is now outside of view

Future cont.

a) Appearance of the model immediately after the start of streaming.

c) 10 seconds after (a).

b) 1 second after (a).

d) 60 seconds after (a).

Image from Streaming QSplat: A Viewer for Networked Visualization of Large, Dense Models

Future cont.

Image from Streaming QSplat: A Viewer for Networked Visualization of Large, Dense Models

Future cont.

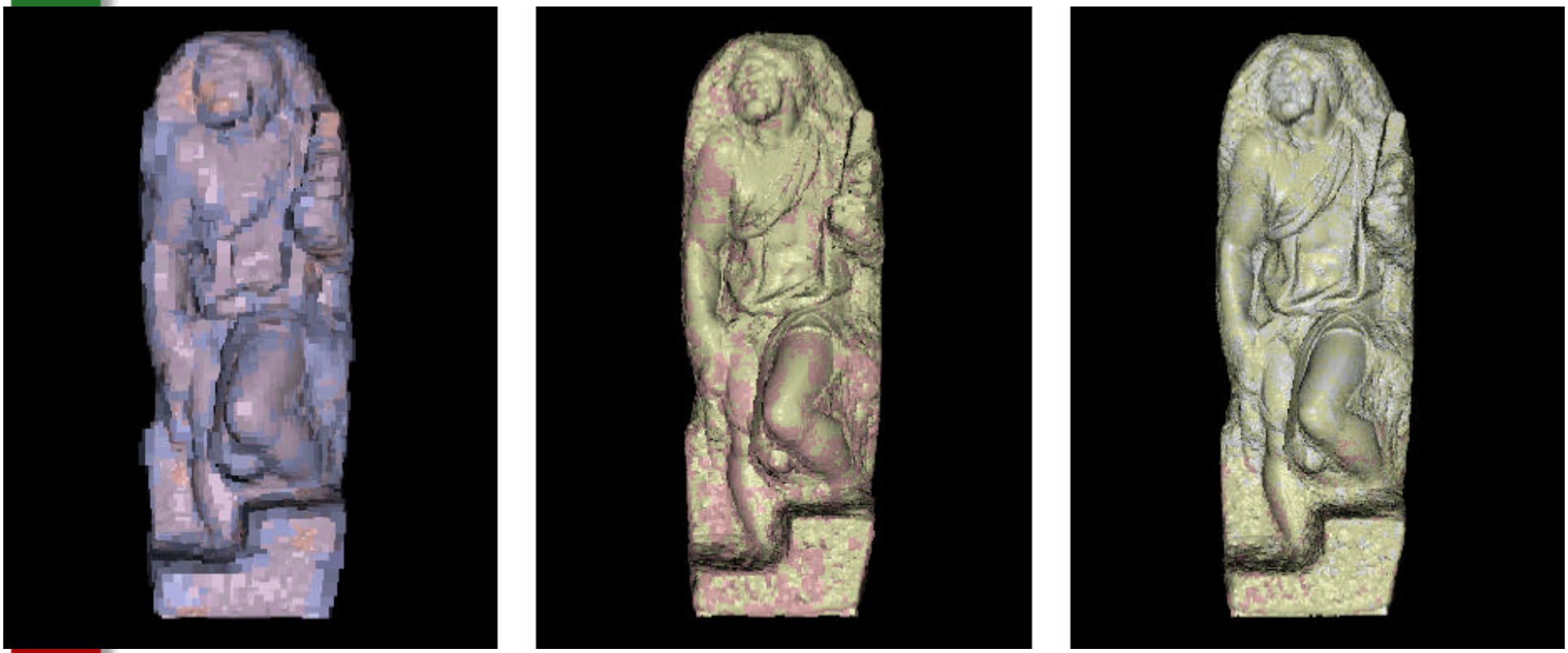


Image from Streaming QSplat: A Viewer for Networked Visualization of Large, Dense Models

Future cont.

- Handheld computers
  - No 3D hardware, which has little affect on QSplat
  - Smaller screen space / less pixels
  - Real-time because of the threshold heuristic

Future cont.

- Pre-fetching was tested but does not make a significant difference
- Sphere hierarchy is well suited data structure for ray tracing
- Transparency, BRDF’s could be added for better visual quality
- Stream of data, remove the temp file on client computer

References

- Rusinkiewicz Szymon, Levoy Mark, Streaming QSplat: A Viewer for Networked Visualization of Large, Dense Models http://graphics.stanford.edu/papers/sqsplat/ ACM 2001
- Rusinkiewicz Szymon, Levoy Mark, QSplat: A Multiresolution Point Rendering System for Large Meshes, SIGGRAPH 2000
- Stamos Ioannis, http://homepage.mac.com/ingart/3D/3q.html http://homepage.mac.com/ingart/3D/splat.ppt, April 5th 2005
- Wikipedia http://en.wikipedia.org/wiki/Huffman_coding, April 5th 2005
- Szymon Rusinkiewicz http://graphics.stanford.edu/software/qsplat/download.html, April 5th 2005
- http://www.csc.calpoly.edu/~zwood/teaching/csc570/final/kle/, April 6, 2005