MESH ENCODINGS FOR X3DOM: RECENT ADVANCES

Fraunhofer IGD

Max Limper

Fraunhofer IGD
Fraunhoferstraße 5
64283 Darmstadt, Germany

Tel.: +49 (0) 6151 155 – 554
Fax.: +49 (0) 6151 155 – 105
Email: max.limper@igd.fraunhofer.de

http://www.igd.fraunhofer.de/vcst

Outline

- Introduction
  - Declarative 3D Scalability Issues
  - Large Model Visualization with X3DOM
  - Declarative 3D Mesh Encoder: Key Aspects

- X3DOM Mesh Encodings
  - Sequential Image Geometry
  - Progressive Binary Encoder
  - POP (Progressively ordered Primitive) Buffer

Motivation:
X3DOM / Declarative 3D Scalability Issues

- X3DOM / Declarative 3D
  - Simplifies 3D WebApp development by extending HTML/DOM with new 3D elements
  - **Scalability issues**: mesh/anim data, encoded as text, leads to huge pages
  - **Solution**: external asset container, requested on demand. (Similar to <img>)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
                <title>Single Mesh Model with a flat hierarchy (small model)
</title>
        <link rel="stylesheet" type="text/css" href="x3dom.css" />
        </head>

<body>

    <h1>Single Mesh Model with a flat hierarchy (small model)</h1>

        <p class="case">
            <!-- Use the same "x", "y", "width" and "height" attributes as SVG.
            Optionally support the "viewBox" and "preserveAspectRatio" just
like SVG -->
                <X3D xmlns="http://www.web3d.org/specifications/x3d-namespace"
id="spheres" showStat="true" x="0px" y="0px" width="400px" height="400px">

  <Scene DEF='scene'>
    <Viewpoint position='0 0 25' orientation="0 0 0 1" />
     <Background skyColor='1 1 1'/>
   <Shape>
          <Appearance DEF='App'>
            <Material ambientIntensity='0.0243902' diffuseColor='0.41 0.39 0.03'
shininess='0.12' specularColor='0.94 0.72 0'/>
          </Appearance>
          <IndexedFaceSet DEF='_0' solid='false' creaseAngle='3.14'


coordIndex='658 660 689 -1 689 660 688 -1 658 659 660 -1 657 658 689 -1 657 689
690 -1 656 657 690 -1 656 690 691 -1 655 656 691 -1 655 691 692 -1 654 655 692
-1 654 692 693 -1 653 654 693 -1 653 693 694 -1 652 653 694 -1 652 694 695 -1
651 652 695 -1 651 695 696 -1 664 651 696 -1 664 696 697 -1 663 664 697 -1 663
697 698 -1 661 698 687 -1 662 663 698 -1 662 698 661 -1 628 629 646 -1 1277 1262
1261 -1 1507 1440 1441 -1 1506 1439 1440 -1 1506 1440 1507 -1 1505 1438 1439 -1
1505 1439 1506 -1 1504 1437 1438 -1 1504 1438 1505 -1 1503 1436 1437 -1 1503
1437 1504 -1 1502 1435 1436 -1 1502 1436 1503 -1 1434 1435 1502 -1 1501 1507
1441 -1 1500 1506 1507 -1 1500 1507 1501 -1 1499 1505 1506 -1 1499 1506 1500 -1
1498 1504 1505 -1 1498 1505 1499 -1 1497 1503 1504 -1 1497 1504 1498 -1 1496
1502 1503 -1 1496 1503 1497 -1 1434 1502 1496 -1 1495 1501 1441 -1 1494 1500
1501 -1 1494 1501 1495 -1 1493 1499 1500 -1 1493 1500 1494 -1 1492 1498 1499 -1
1492 1499 1493 -1 1491 1497 1498 -1 1491 1498 1492 -1 1490 1496 1497 -1 1490
1497 1491 -1 1434 1496 1490 -1 1489 1495 1441 -1 1488 1494 1495 -1 1488 1495
1489 -1 1487 1493 1494 -1 1487 1494 1488 -1 1486 1492 1493 -1 1486 1493 1487 -1
1485 1491 1492 -1 1485 1492 1486 -1 1484 1490 1491 -1 1484 1491 1485 -1 1434
1490 1484 -1 1483 1489 1441 -1 1482 1488 1489 -1 1482 1489 1483 -1 1481 1487
1488 -1 1481 1488 1482 -1 1480 1486 1487 -1 1480 1487 1481 -1 1479 1485 1486 -1
1479 1486 1480 -1 1478 1484 1485 -1 1478 1485 1479 -1 1434 1484 1478 -1 1477
1483 1441 -1 1476 1482 1483 -1 1476 1483 1477 -1 1475 1481 1482 -1 1475 1482
1476 -1 1474 1480 1481 -1 1474 1481 1475 -1 1473 1479 1480 -1 1473 1480 1474 -1
1472 1478 1479 -1 1472 1479 1473 -1 1434 1478 1472 -1 1471 1477 1441 -1 1470
1476 1477 -1 1470 1477 1471 -1 1469 1475 1476 -1 1469 1476 1470 -1 1468 1474
1475 -1 1468 1475 1469 -1 1467 1473 1474 -1 1467 1474 1468 -1 1466 1472 1473 -1
1466 1473 1467 -1 1434 1472 1466 -1 1465 1471 1441 -1 1464 1470 1471 -1 1464
1471 1465 -1 1463 1469 1470 -1 1463 1470 1464 -1 1462 1468 1469 -1 1462 1469
1463 -1 1461 1467 1468 -1 1461 1468 1462 -1 1460 1466 1467 -1 1460 1467 1461 -1
1434 1466 1460 -1 1459 1465 1441 -1 1458 1464 1465 -1 1458 1465 1459 -1 1457
```

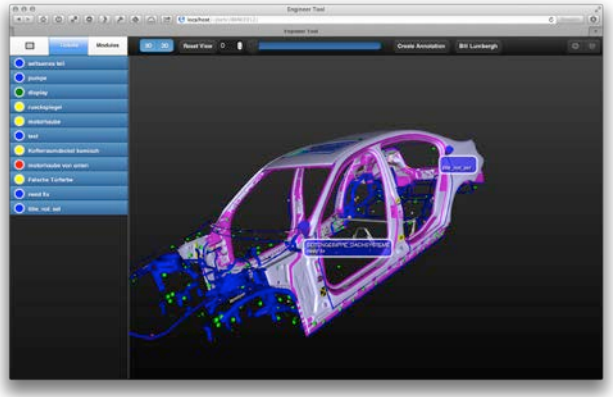
- Core Challenges:
  - How to efficiently encode meshes?
  - How to manage loading on demand?

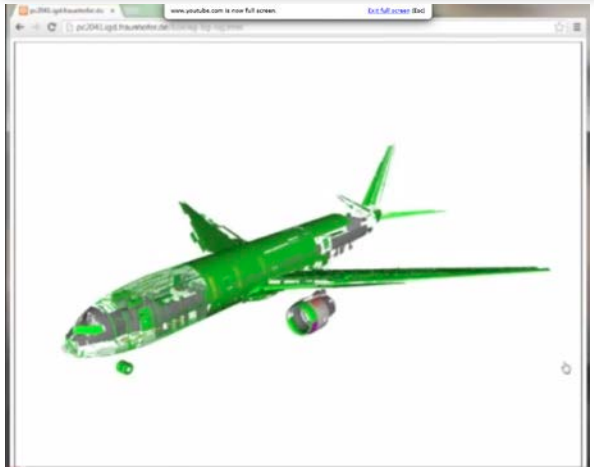
TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

Motivation:
Large Model visualization with X3DOM

- Out-Of-Browser Based Rendering (Out-Of-Core with W3C Technology)
  - **Goal**: Great user experience with massive datasets
  - **Solution**: Hybrid tree structure for client and server based visibility calculation which provides application controlled priorities





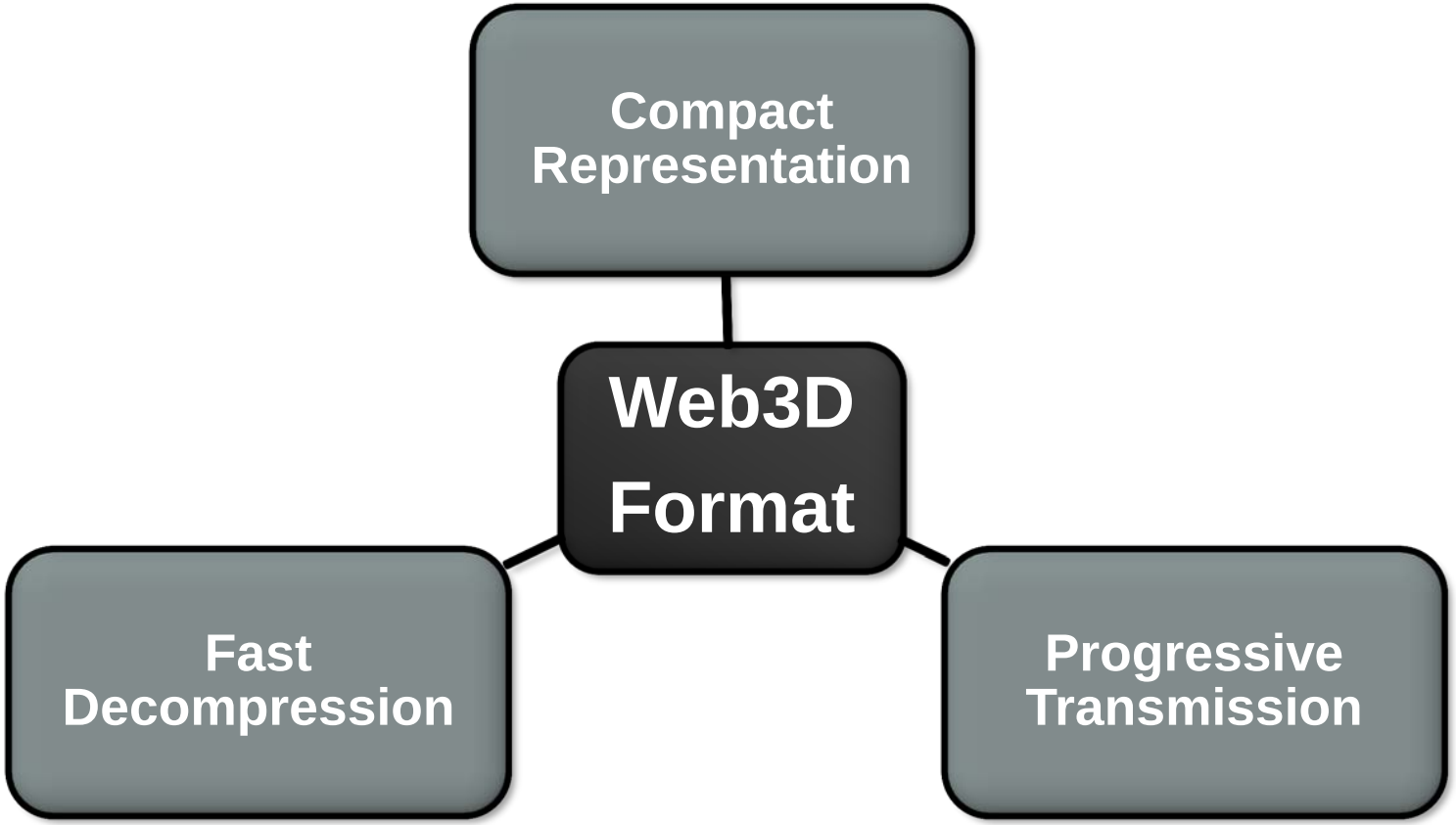
- Core Challenges:
  - How to efficiently page mesh data?
  - How to manage loading on demand?

based on 09-08-30-js4_PPT_Master_IGD

TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

Declarative 3D Mesh Encoder
Key Aspects



TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

Declarative 3D Mesh Encoder
Contradictory Constraints

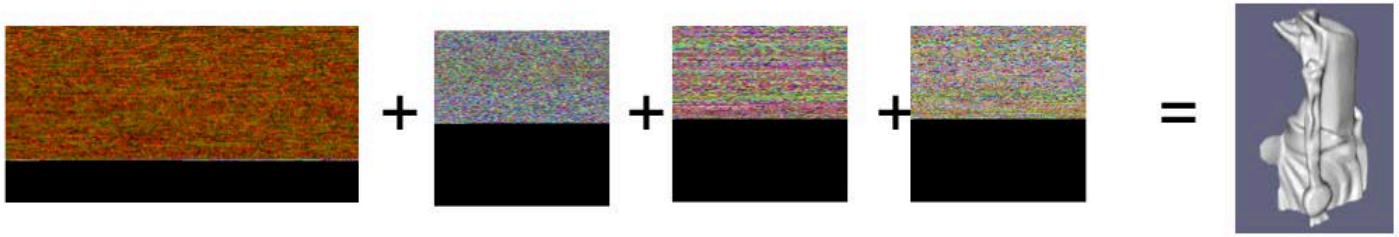
- **Compact Representation**
  - Externalize binary mesh data from HTML documents (*TypedArray* Spec.)
  - Integer quantization of floating-point mesh attributes (e.g., 16 bit)
  - Compression algorithms?

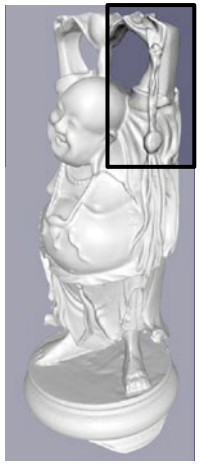
- **Progressive Transmission**
  - Progressive meshes (Hoppe) difficult for generalized 3D data
  - Progressive transmission ↔ fast decoding, fast GPU upload ?

- **Fast Decompression**
  - Ideal: Download mesh data, direct GPU upload
  - Still efficient: Exploit common HTTP / browser features (Images,GZIP, …)

Sequential Image Geometry

- 2010: No support for binary transmission with XHR/W3C
- **Solution:** Utilize images for meshes
  - Browser/Server are well-optimized for parallel fetching/decoding
  - Images can directly be uploaded to GPU buffers (textures)

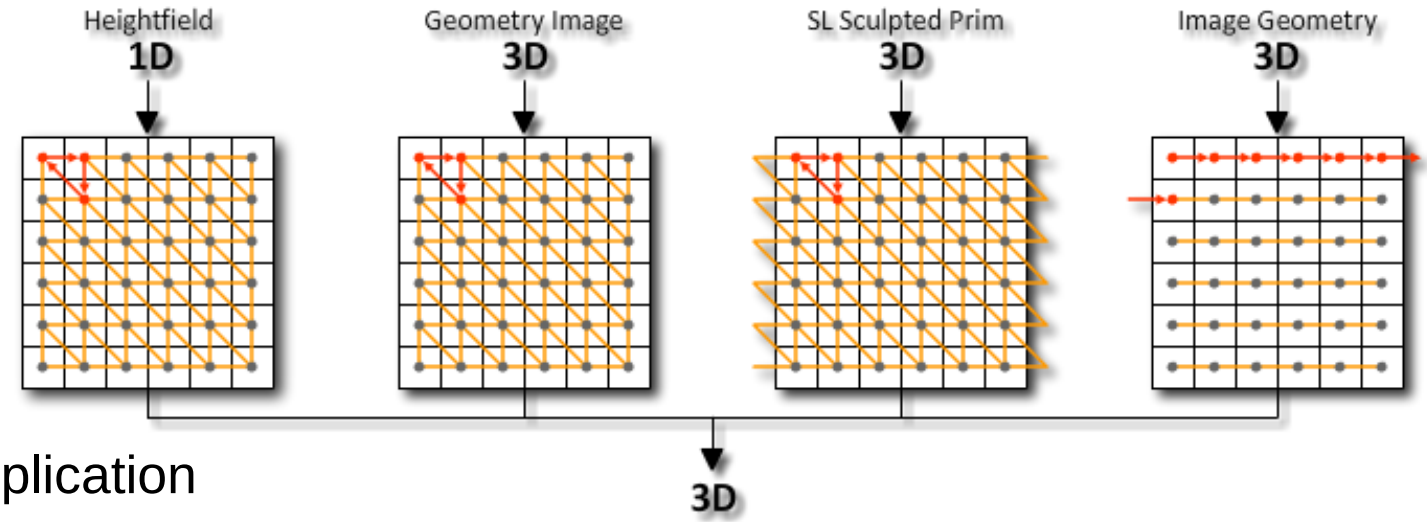




J. Behr et al., **"Using Images and Explicit Binary Container for Efficient and Incremental Delivery of Declarative 3D Scenes on the Web"**, *Proc. 17th Int'l Conf. Web3D Technology* (Web3D 12), ACM, 2012, pp. 17–25.

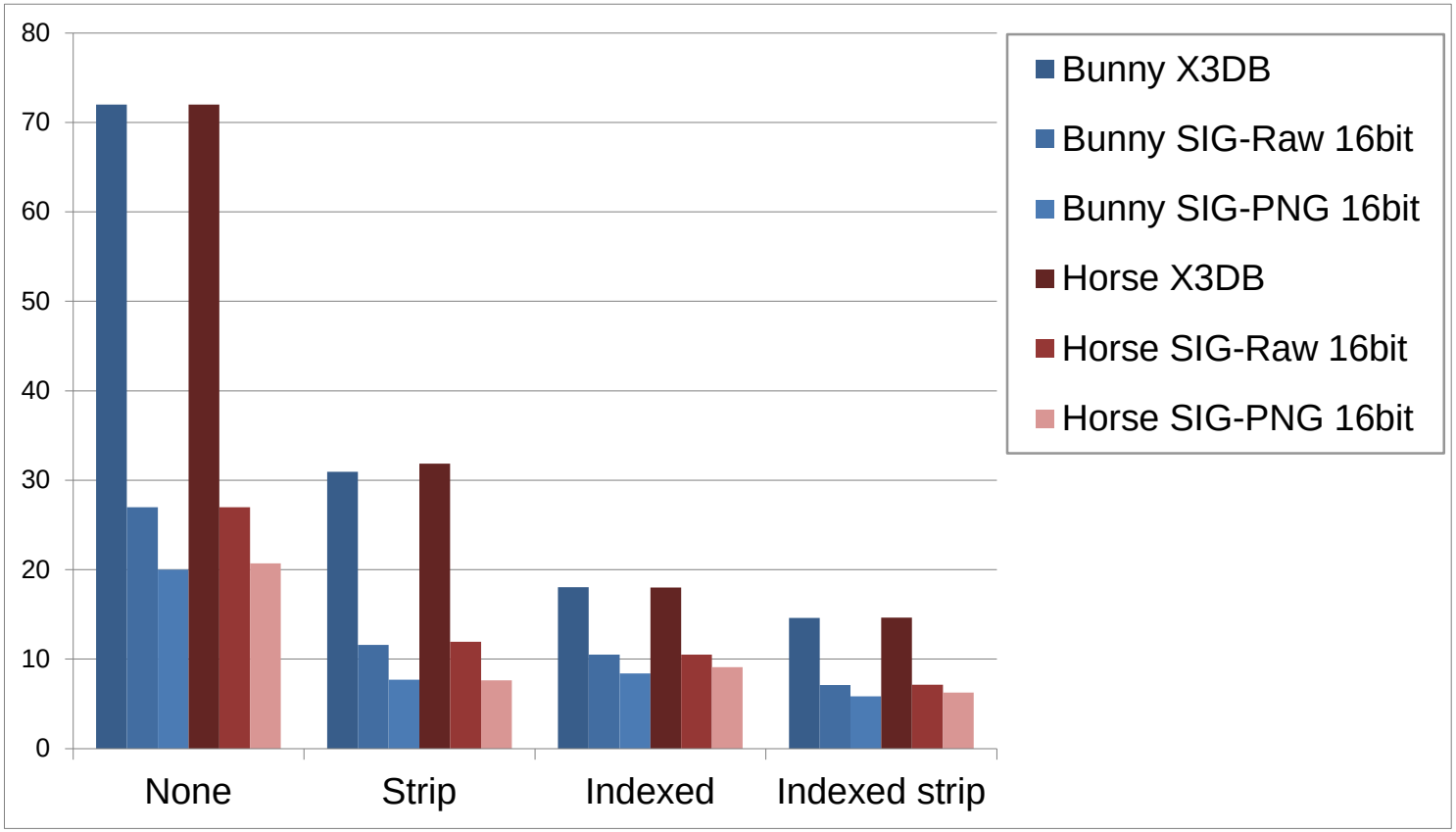
Sequential Image Geometry

- Implicit mesh does not correlate with the mesh topology



- Application
  - Supports: Transmission, compression (partially), rendering, ...
  - Does not support: Scaling of SIG container
- Works with any mesh type and keeps the original topology

Sequential Image Geometry
Bytes per triangle

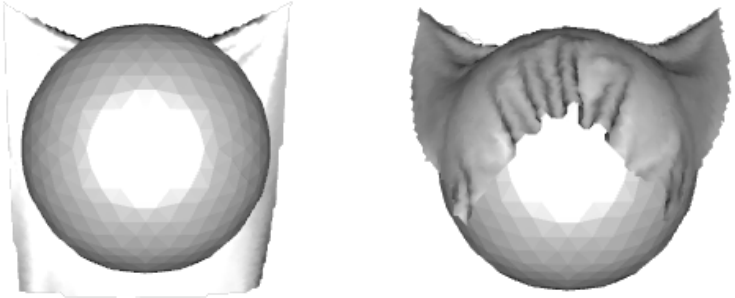


TECHNISCHE UNIVERSITAT DARMSTADT

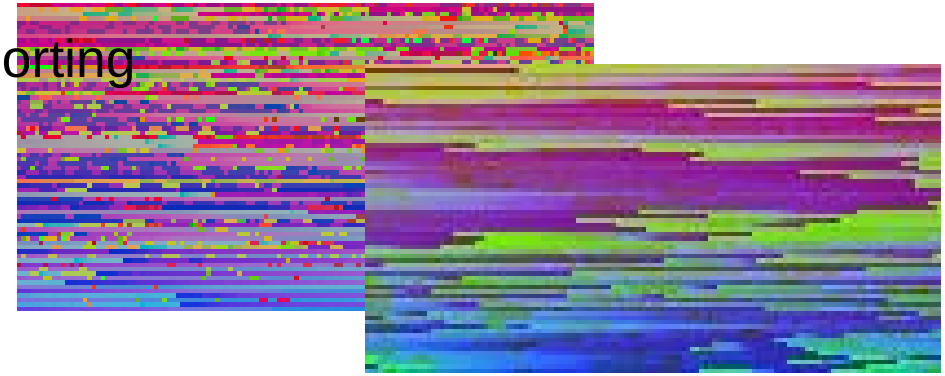
Fraunhofer IGD

Sequential Image Geometry
Image Streaming and compression

- Encode mesh animation in video encoder/decoder



- Optimize compression by resorting mesh components

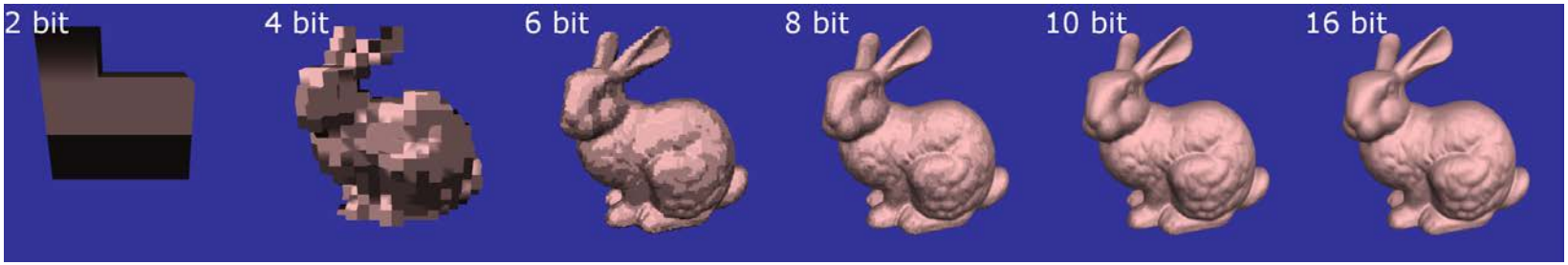


Coordinate Image (Detail):

91.4 KB → 44.7 KB

Progressive Binary Container

- 2012: TypedArray (Binary Buffer) support in JavaScript / XHR (W3C)
- Utilize TypedArray container for meshes
  - TypedArray container can directly be uploaded to GPU buffer
  - Gzip compression supported by the browser
- Progressive Updates similar to SIG, but not limited through fixed-size (e.g., 8 bit) image channels



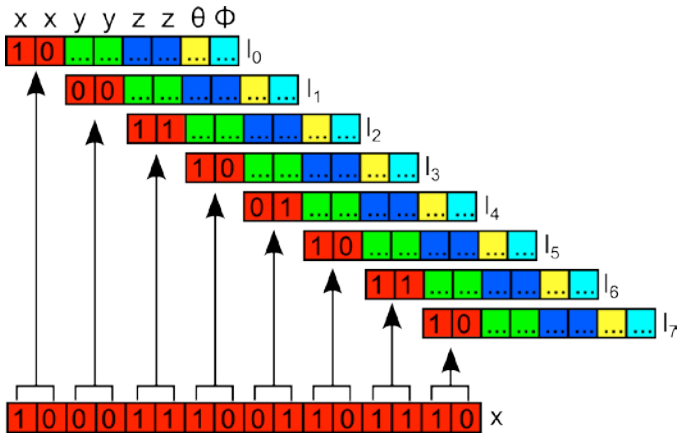
M. Limper et al., **"Fast, Progressive Loading of Binary-Encoded Declarative-3D Web Content,"** *IEEE Computer Graphics and Applications*, vol. 33, no. 5, pp. 26-36

TECHNISCHE UNIVERSITAT DARMSTADT

Fraunhofer IGD

Progressive Binary Container

- Advantage
  - Simple progressive updates by refining the quantization level
- Disadvantages
  - Transmits always the full topology / number of triangles → no LOD
  - Costly GPU updates (bit shifts and adds)



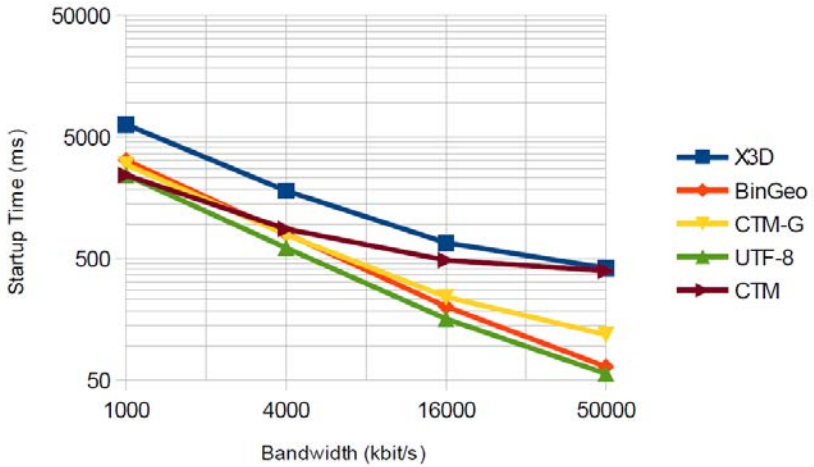
TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

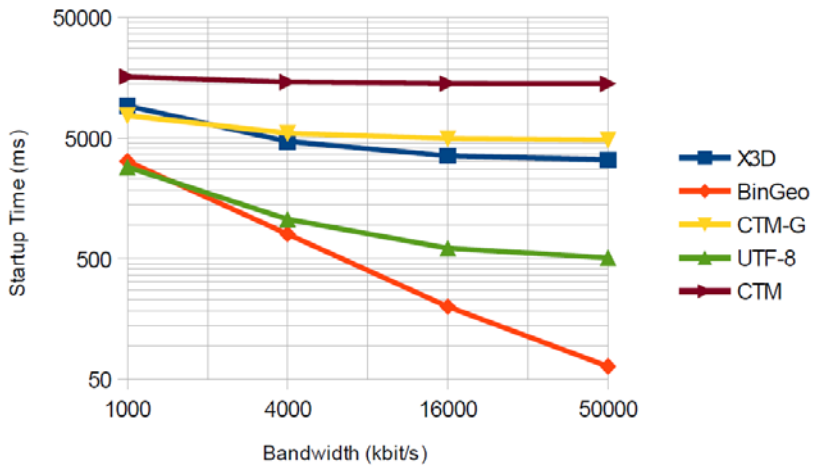
Binary Mesh Container Comparison

- **Case Study:** Compact Representation ↔ Fast Decompression

Desktop



iPad 3



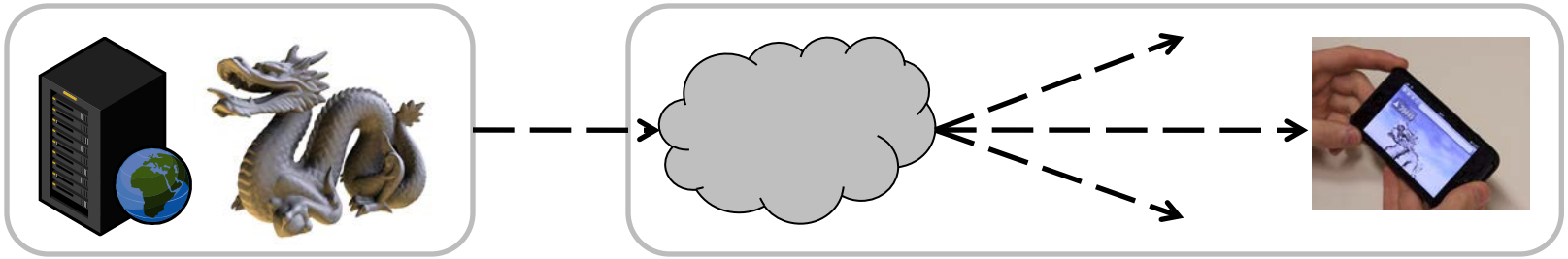
- **Still missing: Progressive topology refinement and LOD**

M. Limper et al., **"Fast Delivery of 3D Web Content: A Case Study"**, *Proc. 18th Int'l Conf. 3D Web Technology* (Web 3D 13), ACM, 2013, pp. 11–18.

TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

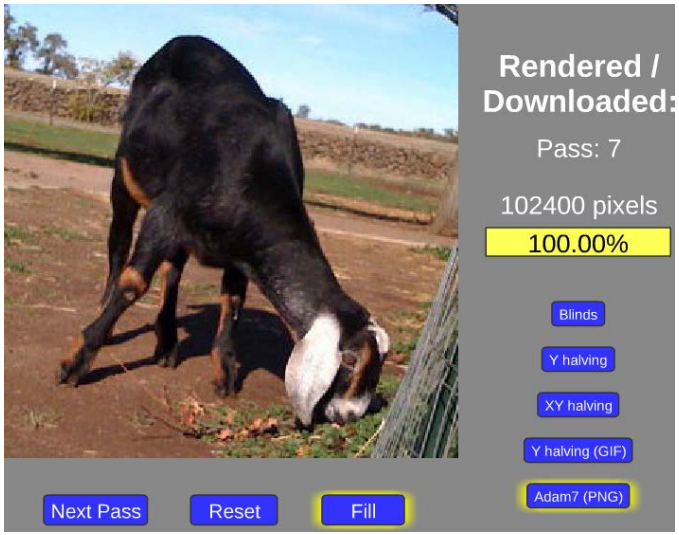
X3DOM POP Buffer: Simplified progressive transmission



- Progressive Meshes? (Hoppe ‘96, Alliez/Desbrun ‘01, Peng ‘05, …)
  - Most don’t encode **general** 3D meshes, just 2-manifolds
  - Tradeoff: **Bandwidth Usage** vs. **Decode Time**
  - Web & Mobile clients: Limited computational power
- Simple Web-capable Implementation, as known from Images?

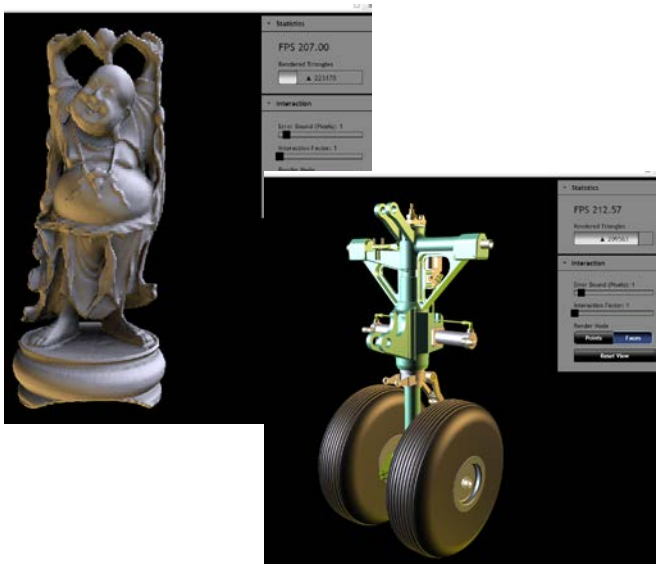
Progressive Transmission: Interlacing Schemes

Images (PNG, Adam7)



http://beej.us/blog/data/image-interlacing/

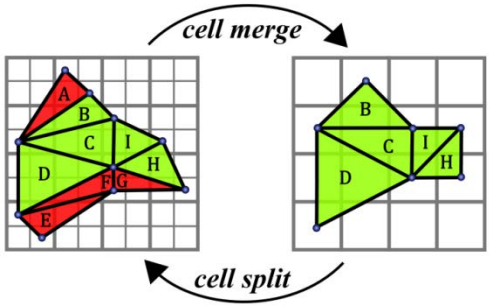
Meshes (POP Buffer)

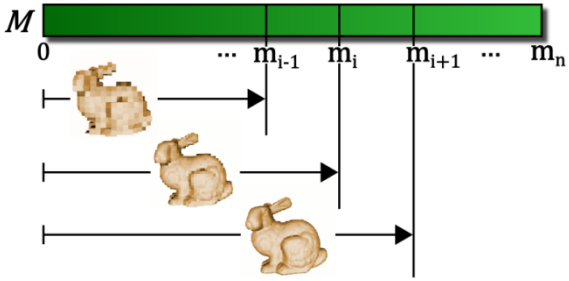


M. Limper et al., **„The POP Buffer: Rapid Progressive Clustering by Geometry Quantization“**, *Computer Graphics Forum (Proc. Pacific Graphics 2013)*, to appear

Progressively Ordered Primitive (POP) Buffer

- Coarse **quantization** of geometry (e.g.. 5 Bits) → **degenerate triangles**
- Successive removal of degenerate triangles at each quantization level





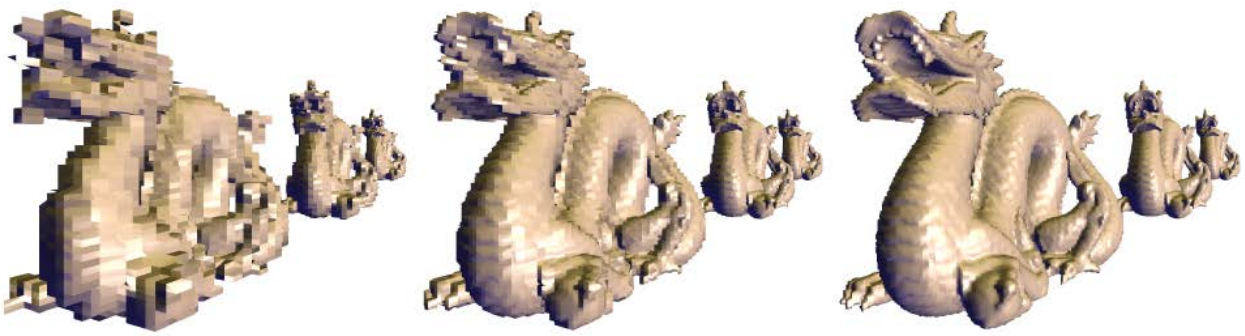
- **Reordering** of triangle data according to degeneration property

- Progressive **streaming** and **LOD**

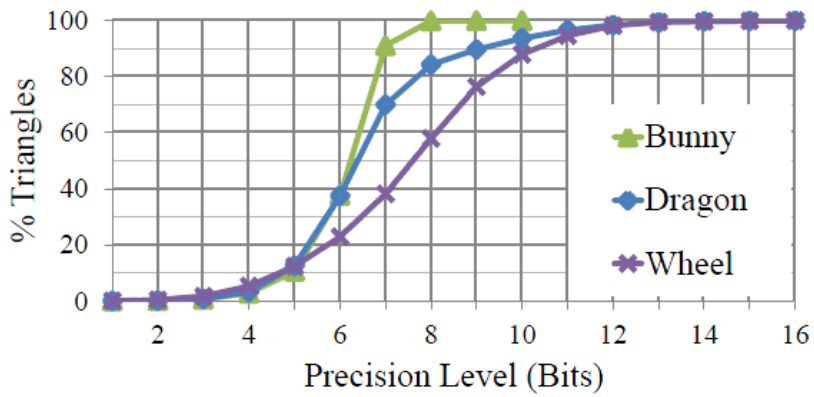
TECHNISCHE UNIVERSITÄT DARMSTADT

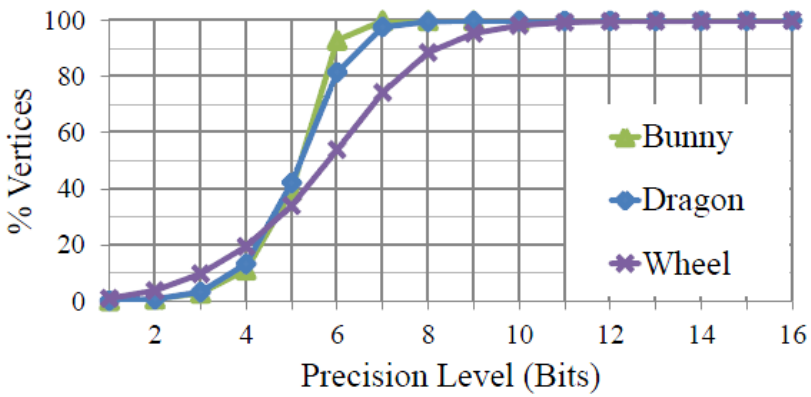
Fraunhofer IGD

X3DOM POP Buffer
Progressive Streaming and LOD



(a) **4 bit, 3% loaded** (b) **5 bit, 13% loaded** (c) **6 bit, 38% loaded**





TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

X3DOM POP Buffer Benefits

- Handles arbitrary topology
- Conversion at interactive rates
- No CPU-based decoding steps
- Stateless structure (less GPU traffic, Instancing)
- No GPU memory overhead
- Straightforward implementation (WebGL)

Try it yourself

- Paper video and demos available at www.x3dom.org/pop

- Demos need WebGL-capable browser

Thank you for listening!

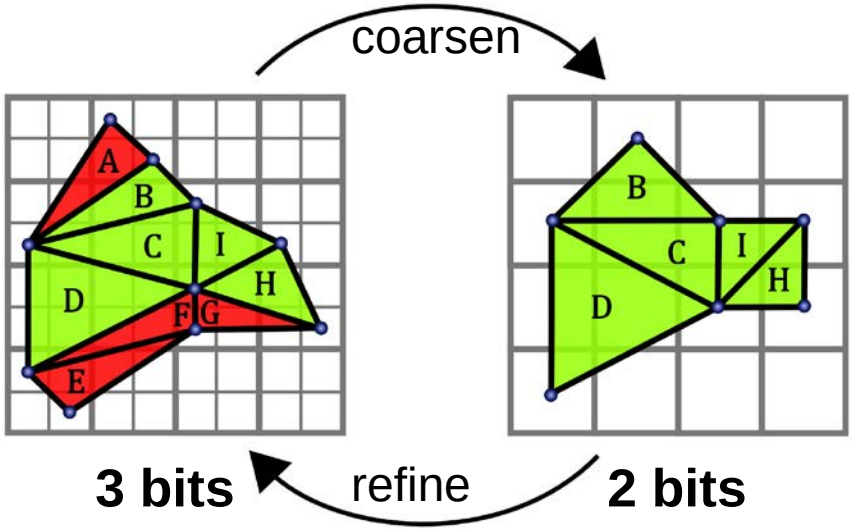
Questions?

TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

Appendix:
POP Buffer Details

Ordering Triangles: Finding degenerated geometry



Non-degenerate triangles:

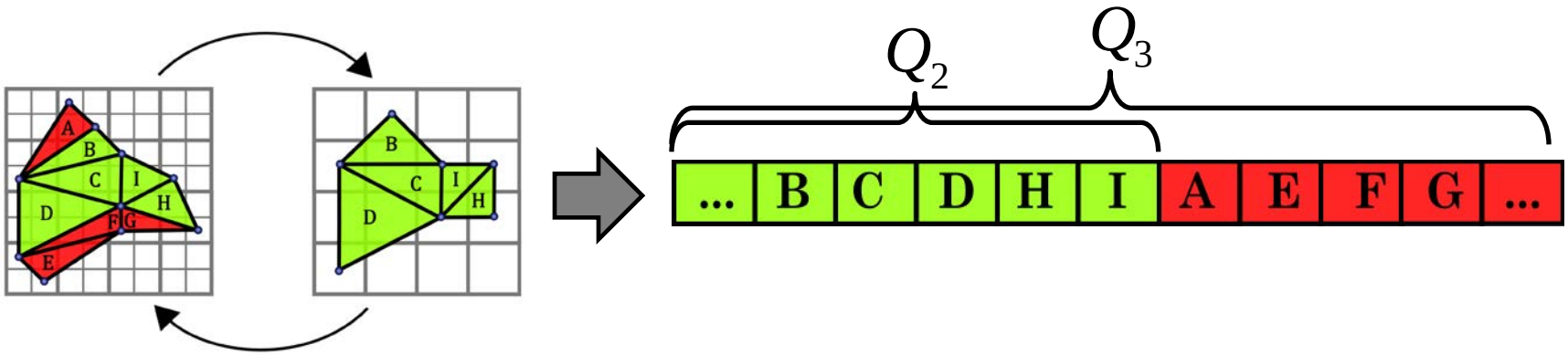
…

$Q_2 = \{B, C, D, H, I\}$

$Q_3 = \{A, B, C, D, E, F, G, H, I\}$

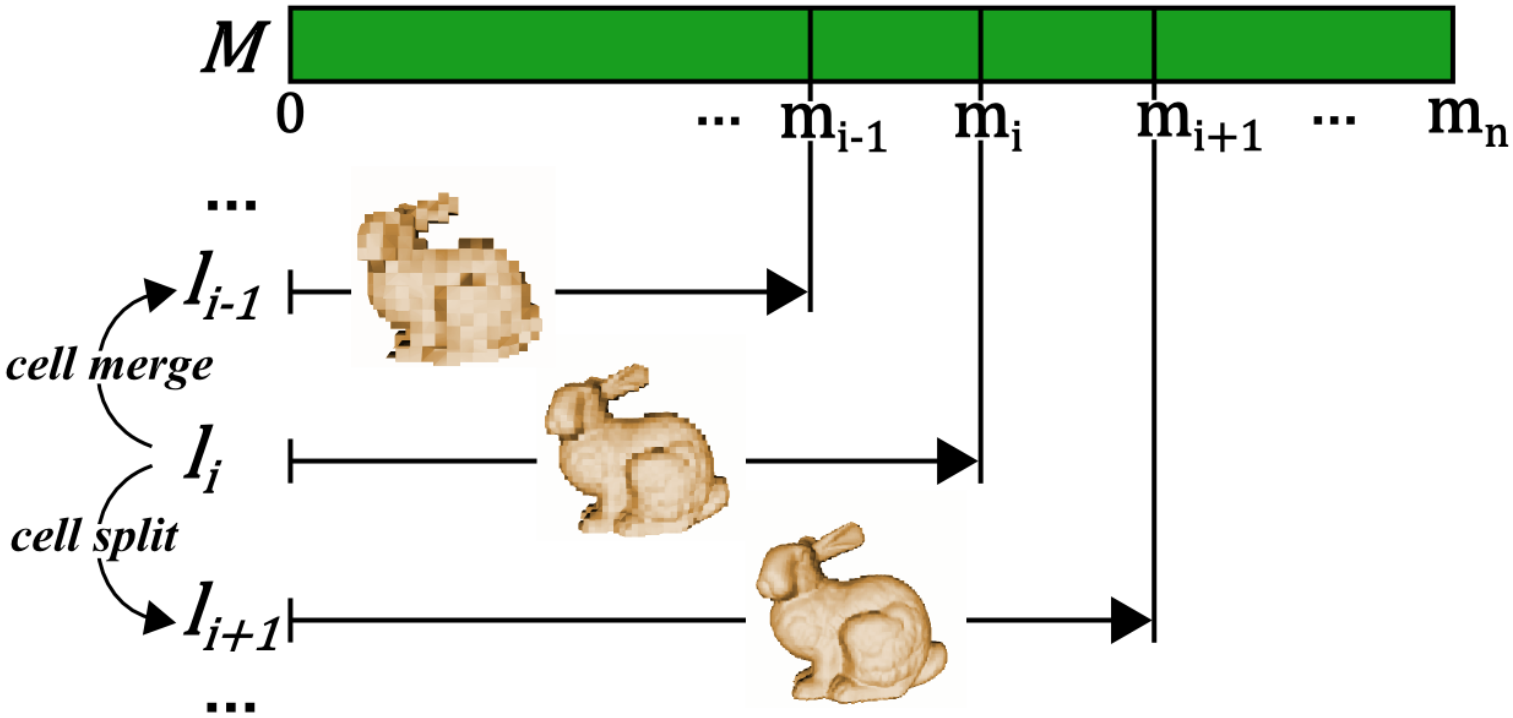
…

Ordering Triangles: Sorting



- Classify triangles, sort them
  - Order within sets can be freely chosen
  - Fast sorting (linear time), ~5-6 Mio. Δ / s

Progressively Ordered Primitive (POP) Buffer

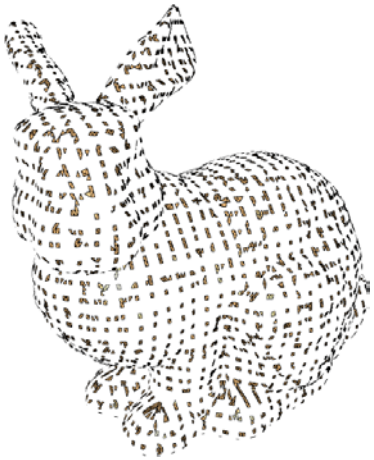


TECHNISCHE UNIVERSITÄT DARMSTADT

Fraunhofer IGD

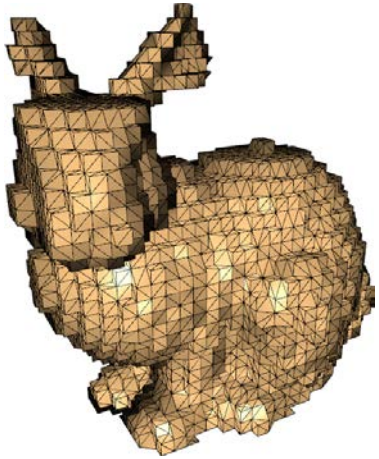
Progressive Transmission & Rendering

$\mathbf{m_i}$ Triangles,
rendered with 16 bits



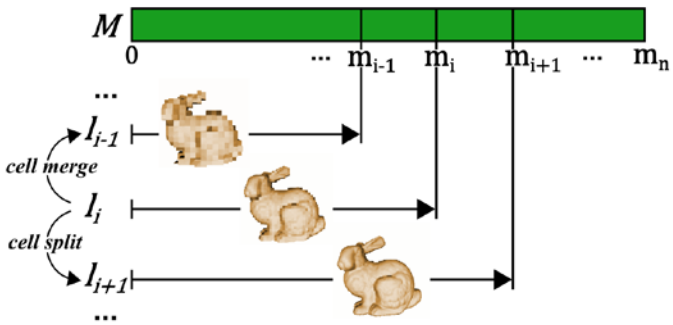
Vertex Shader

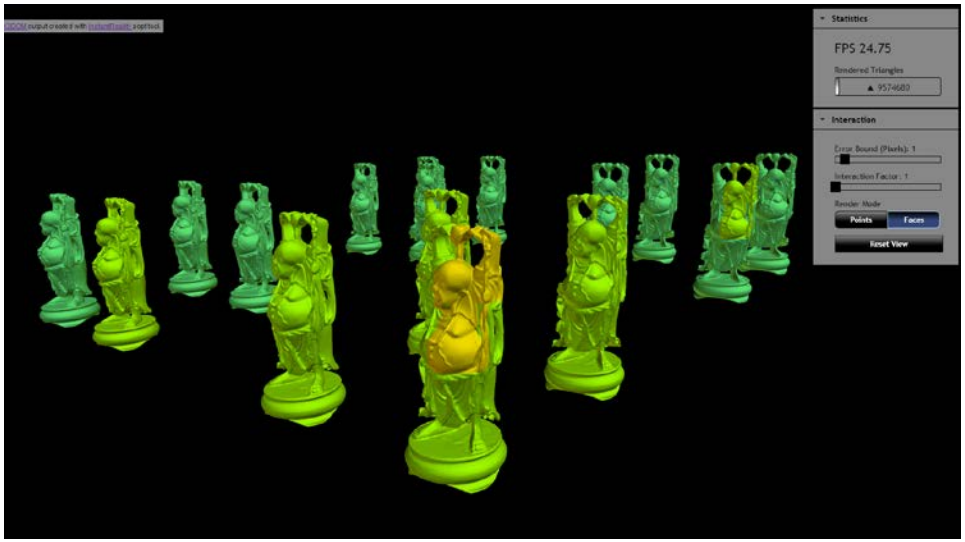
same $\mathbf{m_i}$ Triangles,
rendered with **i** bits



- Don‘t send less bits/vertex, just send less vertices!
  - Also applies for index data, normals, texture coordinates
  - No additional decoding, guarantees **stateless** property

The Stateless POP Buffer





- One static buffer for all detail levels
- One static buffer for all instances

View-Dependent LOD: Closing Cracks

- Partition mesh for view-dependent LOD
- Problem: Cracks between sub-meshes
- Simple solution: Protected vertices
  - Move border vertices to begin of buffer
  - Always render those with full precision
  - Can be used for shape / feature preservation