Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Johannes Unterguggenberger, Bernhard Kerbl, Jakob Pernsteiner, and Michael Wimmer

TU Wien, Institute of Visual Computing & Human-Centered Technology, Austria
Clusters!

"Gawain" model © by Unity Technologies, provided through their "The Heretic: Digital Human" package.
Skinned Meshes

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

- Skinned, animated models
- Underlying skeleton
  - Bone hierarchy
  - Hierarchical transformation

"Gawain" model © by Unity Technologies, provided through their "The Heretic: Digital Human" package.
Skinned Mesh...lets

Conservative Meshlet Bounds for Robust Culling of **Skinned Meshes**

- Skinned, animated models
- Underlying skeleton
  - Bone hierarchy
  - Hierarchical transformation
- Vertices (the skin)
  - Assigned to one or multiple bones
  - Transformed by bones (weighted)

"Gawain" model © by Unity Technologies, provided through their "The Heretic: Digital Human" package.
Meshlets!

Conservative **Meshlet** Bounds for Robust Culling of Skinned Meshes

- **Meshlets**
  - Small clusters of geometry
  - E.g., small patch of indexed geometry
- New shader stages (task and mesh) for efficient processing
  - within rasterization pipelines
  - Using **Vulkan** with **Nvidia** extension
    - Max. 64 vertices and 126 triangles

Classical graphics pipeline:

```

**Johannes Unterguggenberger et al.**
Conservative **Meshlet** Bounds for Robust Culling of Skinned Meshes

- **Meshlets**
  - Small clusters of geometry
  - E.g., small patch of indexed geometry
- New shader stages (**task** and **mesh**) for efficient processing
  - *within* rasterization pipelines
  - Using **Vulkan** with **Nvidia** extension
    - Max. 64 vertices and 126 triangles

Classical graphics pipeline:

- **Vertex Shader**
- **Tess.Ctrl. Shader**
- **Tess.Eval. Shader**
- **Geometry Shader**
- **Rasterizer**
- **Fragment Shader**
Task and mesh shader pipeline:
- Efficient processing of meshlets
- Culling in task shader stage

Conservative **Meshlet** Bounds for Robust Culling of Skinned Meshes

- **Meshlets**
  - Small clusters of geometry
  - E.g., small patch of indexed geometry

- **New shader stages** (task and mesh) for efficient processing
  - within rasterization pipelines
  - Using **Vulkan** with **Nvidia** extension
    - Max. 64 vertices and 126 triangles
Rendering Artefacts To Be Avoided
Rendering Artefacts To Be Avoided

a meshlet

premature (i.e., non-conservative) view frustum culling
Rendering Artefacts To Be Avoided

a meshlet
a meshlet

premature (i.e., non-conservative) back-face culling
AVOID!

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Rendering Artefacts To Be Avoided
Rendering Artefacts To Be Avoided

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Fine-Grained Culling of Meshlets
Fine-Grained Culling of Meshlets
Rendering Artefacts To Be Avoided

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
What happens to our meshlet under animation?
What happens to our meshlet under animation?

Let's animate that bone, which in turn transforms the meshlet's vertices.

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
The meshlet's shape changed...
The meshlet's original bounding box...

...which is a problem in terms of its bounds.
...no longer encompasses all **vertex positions** in the transformed state.

Also, the face normals distribution has changed under animation.
Meshlet Bounds Algorithm:
1. Compute all *vertex bounds*
2. Combine into *meshlet bounds*
Vertex Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Meshlet Bounds Algorithm:
1. Compute all *vertex bounds*
2. Combine into *meshlet bounds*

Initialize:
Bounds at $t=0$
Calculate:
1. Bounds from $t=0$ to $t=1$ per vertex
Calculate:
1. Bounds from $t=0$ to $t=1$ per vertex
2. Meshlet bounds = \text{min} and \text{max} bounds
   Normals distribution based on vertex bounds, too
Vertex Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Choice of space where to store/calculate vertex/meshlet bounds

Bounds from $t=0$ to $t=1$
Choice of space:
Space of the most influential bone

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Bounds from $t=0$ to $t=1$
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Choice of space:
Space of the most influential bone

We are going to:
- Compute initial bounds in that space
- Transform bounds with that space
We will:

- Compute initial bounds in that space
- Transform bounds with that space

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Calculate bounds for all animation intervals of interest!

- Transform bounds **with** that space
Calculate bounds for all animation intervals of interest!

from $t=0$ to $t=1$
from $t=1$ to $t=2$
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Calculate bounds for all animation intervals of interest!

from $t=0$ to $t=1$
from $t=1$ to $t=2$
Calculate bounds for all animation intervals of interest!

from $t=0$ to $t=1$
from $t=1$ to $t=2$
Meshlet Bounds

**Conservative Meshlet Bounds** for Robust Culling of Skinned Meshes

Calculate bounds for all animation intervals of interest!

from $t=0$ to $t=1$
from $t=1$ to $t=2$
Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Calculate bounds for all animation intervals of interest!

from \( t=0 \) to \( t=1 \)
from \( t=1 \) to \( t=2 \)
from \( t=2 \) to \( t=3 \)
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

- Low-overhead culling in task shaders
- Low memory consumption
- Low memory bandwidth

for each meshlet in a precomputation step

from $t=0$ to $t=1$
from $t=1$ to $t=2$
from $t=2$ to $t=3$

Store 1 bounding box and 1 normals distribution "cone"
for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Meshlet Bounds

for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Meshlet Bounds

for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Is this a "well-behaved" meshlet?

from $t=0$ to $t=1$
from $t=1$ to $t=2$
from $t=2$ to $t=3$
for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Meshlet Bounds

Is this a "well-behaved" meshlet?

from $t=0$ to $t=0.5$ and $t=0.5$ to $t=1$
from $t=1$ to $t=1.5$ and $t=1.5$ to $t=2$
from $t=2$ to $t=2.5$ and $t=2.5$ to $t=3$
Meshlet Bounds

for each meshlet in an adaptive precomputation step

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Is this a "well-behaved" meshlet?

from $t=0$ to $t=0.3$ and $t=0.3$ to $t=0.6$ and $t=0.6$ to $t=1$
from $t=1$ to $t=1.3$ and $t=1.3$ to $t=1.6$ and $t=1.6$ to $t=2$
from $t=2$ to $t=2.3$ and $t=2.3$ to $t=2.6$ and $t=2.6$ to $t=3$

Trade precomputation effort for better runtime performance, due to better bounds.

From our test models:
- $\geq 94\%$ view frustum cullable
- $\sim 60\%$–$90\%$ backface cullable
In Our Paper...

Please consult our paper for:
- Vertex bounds algorithm
- Conservative rotation bounds extension (Derivative of Rodrigues' Rotation Formula)
- Vertex bounds combination for linear blend skinning
- Conservative normal bounds from vertex bounds
- Results, Percentages, ...
- Discussion
- Future work

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Results

![Graph showing performance metrics for different GPU scenes and settings.](image)

<table>
<thead>
<tr>
<th>GPU</th>
<th>Scene</th>
<th>BFC only</th>
<th>VFC only</th>
<th>BFC+VFC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Culled</td>
<td>Faster</td>
<td>Culled</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>8.1%</td>
<td>31.3%</td>
</tr>
<tr>
<td>2060</td>
<td>849k</td>
<td>11.5%</td>
<td>8.5%</td>
<td>31.4%</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>7.8%</td>
<td>31.3%</td>
</tr>
<tr>
<td>3070</td>
<td>849k</td>
<td>11.5%</td>
<td>7.9%</td>
<td>31.4%</td>
</tr>
</tbody>
</table>
## Results

### Table: Performance Comparison

<table>
<thead>
<tr>
<th>GPU</th>
<th>Scene</th>
<th>BFC only</th>
<th>VFC only</th>
<th>BFC+VFC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Culled</td>
<td>Faster</td>
<td>Culled</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>8.1%</td>
<td>31.3%</td>
</tr>
<tr>
<td>2060</td>
<td>849k</td>
<td>11.5%</td>
<td>8.5%</td>
<td>31.4%</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>7.8%</td>
<td>31.3%</td>
</tr>
<tr>
<td>3070</td>
<td>849k</td>
<td>11.5%</td>
<td>7.9%</td>
<td>31.4%</td>
</tr>
</tbody>
</table>

### Diagram: Performance Graphs

- **no culling**
- **BFC on, VFC off**
- **BFC off, VFC on**
- **BFC on, VFC on**

- **RTX 2060**
- **RTX 3070**
Results

![Bar chart and table showing performance metrics for different GPU models and scene sizes with and without BFC and VFC optimizations.](chart.png)

<table>
<thead>
<tr>
<th>GPU</th>
<th>Scene</th>
<th>BFC only</th>
<th>VFC only</th>
<th>BFC+VFC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Culled</td>
<td>Faster</td>
<td>Culled</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>31.3%</td>
<td>26.3%</td>
<td>39.9%</td>
</tr>
<tr>
<td>2060</td>
<td>849k</td>
<td>31.4%</td>
<td>27.1%</td>
<td></td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>31.3%</td>
<td>26.1%</td>
<td>39.9%</td>
</tr>
<tr>
<td>3070</td>
<td>849k</td>
<td>31.4%</td>
<td>27.2%</td>
<td>39.7%</td>
</tr>
</tbody>
</table>

Note: The table shows the percentage improvement in performance when using BFC and VFC optimizations compared to baseline performance.
Results

<table>
<thead>
<tr>
<th>GPU</th>
<th>Scene</th>
<th>BFC only</th>
<th>VFC only</th>
<th>BFC+VFC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Culled</td>
<td>Faster</td>
<td>Culled</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>8.1%</td>
<td>31.3%</td>
</tr>
<tr>
<td>2060</td>
<td>849k</td>
<td>11.5%</td>
<td>8.5%</td>
<td>31.4%</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>7.8%</td>
<td>31.3%</td>
</tr>
<tr>
<td>3070</td>
<td>849k</td>
<td>11.5%</td>
<td>7.9%</td>
<td>31.4%</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>GPU</th>
<th>Scene</th>
<th>BFC only</th>
<th>VFC only</th>
<th>BFC+VFC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Culled</td>
<td>Faster</td>
<td>Culled</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>8.1%</td>
<td>31.3%</td>
</tr>
<tr>
<td>2060</td>
<td>849k</td>
<td>11.5%</td>
<td>8.5%</td>
<td>31.4%</td>
</tr>
<tr>
<td>RTX</td>
<td>400k</td>
<td>11.4%</td>
<td>7.8%</td>
<td>31.3%</td>
</tr>
<tr>
<td>3070</td>
<td>849k</td>
<td>11.5%</td>
<td>7.9%</td>
<td>31.4%</td>
</tr>
</tbody>
</table>

![Graph showing performance comparison between RTX 2060 and RTX 3070 with different scene types and BFC/VFC settings.](image-url)
Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Johannes Unterguggenberger, Bernhard Kerbl, Jakob Pernsteiner, and Michael Wimmer
TU Wien, Institute of Visual Computing & Human-Centered Technology, Austria

"Gawain" model © by Unity Technologies, provided through their "The Heretic: Digital Human" package.
Rodrigues' Rotation Formula

Rotating a vector in space, given axis and angle of rotation.

Quaternions can be converted into that form.
Conservative Rotation Bounds
Conservative Rotation Bounds

\[
\nu' = \nu \cos \theta + (n \times v) \sin \theta + n(n \cdot v)(1 - \cos \theta). \tag{3}
\]

We use its first-order derivative by \( \theta \) to find those angles that lead to maximum extents in each of the principal axes’ directions. Setting that first-order derivative of Equation (3) by \( \theta \) to zero in order to find the extrema results in Equation (4)

\[
x_{\theta} = -\tan^{-1} \left( \frac{n \times v}{n(n \cdot v) - v} \right), \tag{4}
\]

which yields a vector of angles \( x_{\theta} \) in radians that represents the rotation angles which lead to maximum extents in each principal axis direction. Please note that the operations in Equation (4) mean component-wise application of the division and \( \tan^{-1} \).
Appendix

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Johannes Unterguggenberger, Bernhard Kerbl, Jakob Pernsteiner, and Michael Wimmer
 TU Wien, Institute of Visual Computing & Human-Centered Technology, Austria
Vertex Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Key Distinction to Previous Work:
Our algorithm computes bounds per animation interval, i.e., NOT at a specific animation time.
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Previous approaches: conservative bounds at $t=0.5$
Our approach: conservative bounds \textbf{from } t=0 \textbf{ to } t=1

Previous approaches: conservative bounds \textbf{at } t=0.5

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes
Our approach: conservative bounds from $t=0$ to $t=1$

We want to know: How much do any animated bounds deviate from an initial position?

Previous approaches: conservative bounds at $t=0.5$
Our approach: conservative bounds from \( t=0 \) to \( t=1 \)

We want to know: How much do any animated bounds deviate from an initial position?

Because if we know across all animation states => very efficient culling in task shader

no computation/memory fetch for different animation states per meshlet with constant scaling factor
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds \textbf{from } t=0 \textbf{ to } t=1
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds from $t=0$ to $t=1$

**Vertex Bounds Algorithm:**
1. Compute bounds of all *individual bones* of influence (weight $\neq 0$) as if they had weight 1
   a) Initial bounds
   b) Step-wise towards target bone
   c) Conservatively extend
2. Combine weighted into *vertex bounds* according to the skinning method used (LBS)
**Our approach:**

Conservative bounds from \( t=0 \) to \( t=1 \)

**Vertex Bounds Algorithm:**

1. Compute bounds of all *individual bones* of influence (weight \( \neq 0 \)) as if they had weight 1
   a) Initial bounds
   b) Step-wise towards target bone
   c) Conservatively extend
2. Combine weighted into *vertex bounds* according to the skinning method used (LBS)

**Contribution:**

Conservative extension by maximum rotations in x, y, z through a derivative of Rodrigues' Rotation Formula
Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Our approach: conservative bounds from $t=0$ to $t=1$

**Vertex Bounds Algorithm:**
1. Compute bounds of all individual bones of influence (weight $\neq 0$) as if they had weight 1
   a) Initial bounds
   b) Step-wise towards target bone
   c) Conservatively extend
2. Combine weighted into vertex bounds according to the skinning method used (LBS)
Our approach: conservative bounds from $t=0$ to $t=1$
Our approach: conservative bounds from $t=0$ to $t=1$
from $t=0$ to $0.5$ and $t=0.5$ to $1$

Adaptive precomputation step for arbitrarily narrow bounds.
Our approach: conservative bounds from $t=0$ to $t=1$

from $t=0$ to $0.5$ and $t=0.5$ to $1$

from $t=0$ to $0.3$ and $t=0.3$ to $0.6$ and

and $t=0.6$ to $1$

Trading precomputation effort for better runtime performance, due to better bounds.
Meshlet Bounds

Conservative Meshlet Bounds for Robust Culling of Skinned Meshes

Calculate bounds for all animation intervals of interest!

- from $t=0$ to $t=1$
- from $t=1$ to $t=2$
- from $t=2$ to $t=3$

- High memory consumption
- Increased memory bandwidth
- More cache misses