AMD

TRESSFX 3.X

PART OF THE GPUOPEN INITIATIVE

GPUOPEN.COM

HIGH-LEVEL CONTENTS

- Brief TressFX overview
- TressFX 3.x library
  - Update to AMD's TressFX example implementation
  - Maya plugin
  - Viewer and runtime library (with full source)
  - Fur support
  - Skinning
  - Future optimizations
  - New memory-friendly OIT method (ShortCut)

A BRIEF HISTORY

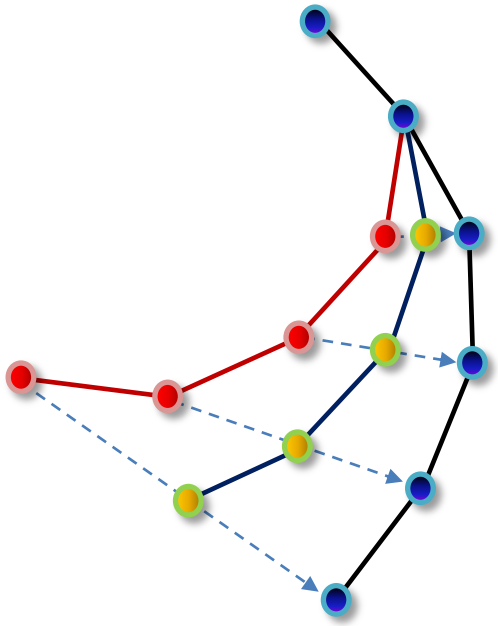
- TressFX began as a collaboration between AMD and Crystal Dynamics
- First used in Tomb Raider
  - PC and consoles
  - Set new quality bar for hair in games
- Optimized for AMD GCN architecture
  - Radeon HD 7000 or later
  - Consoles
- AMD is now also collaborating with Eidos-Montréal
  - Started with Tomb Raider code
  - Improvements and additions integrated into Dawn Engine™
  - Will be used in future Deus Ex Universe projects

CRYSTAL DYNAMICS

SQUARE ENIX

TRESSFX OVERVIEW

- Two parts to TressFX
  - Physics simulation on the GPU using compute shaders
  - High-quality rendering
- Simulates and renders individual hair strands

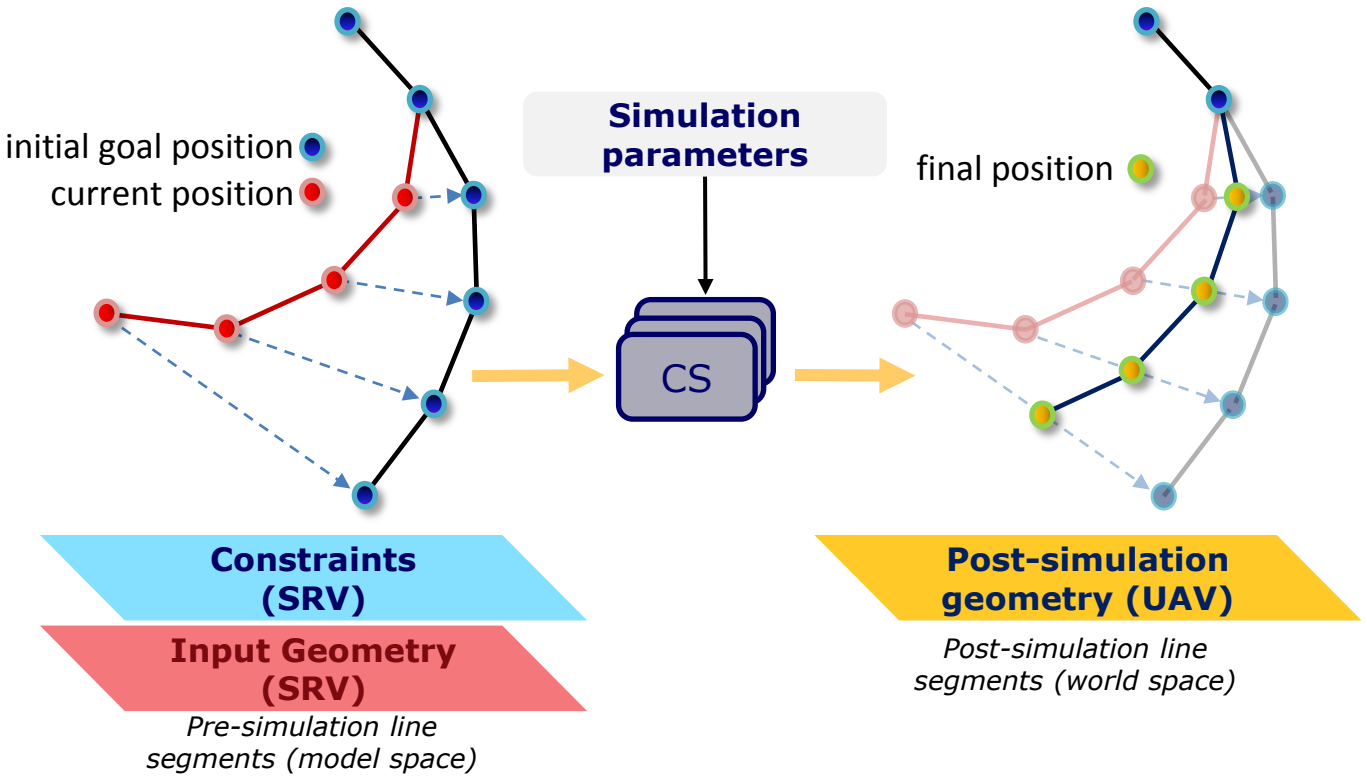


- initial goal position
- current position
- final position

SIMULATION

RENDERING

TRESSFX SIMULATION



- SIMULATION COMPUTE SHADERS
  - Edge length constraint
  - Local shape constraint
  - Global shape constraint
  - Model Transform
  - Collision Shape
  - External Forces (wind, gravity, etc.)

TRESSFX RENDERING

Good Lighting + Anti-Aliasing + Volume Shadows + Transparency

TRESSFX 3.X LIBRARY

TRESSFX SAMPLE HISTORY

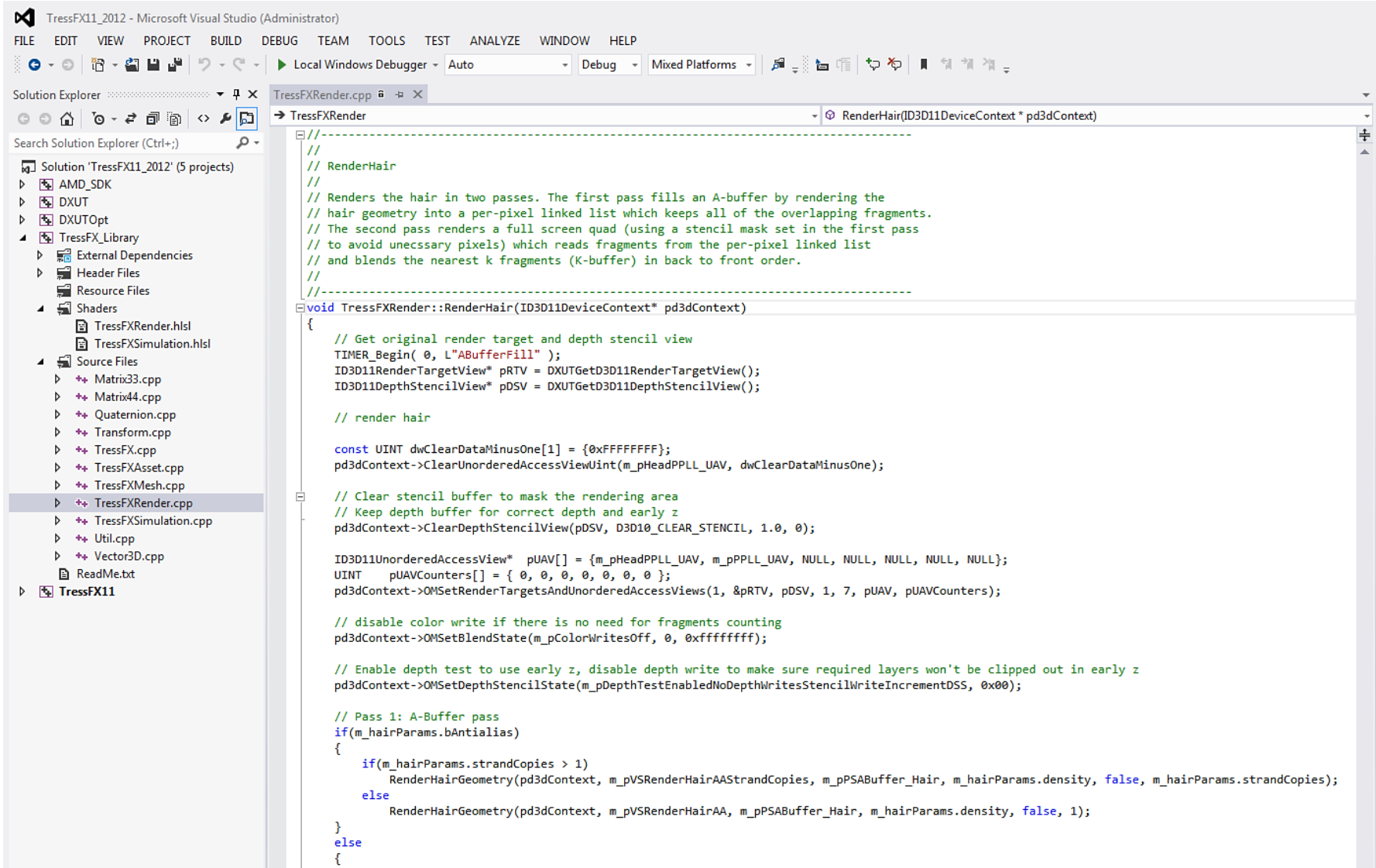
- TressFX 1.0
  - Example implementation (with full source)
- TressFX 2.x
  - Simulation performance improvements
    - Master and slave strands
    - Compute shader optimizations
    - 1.6 ms → 0.3 ms
  - Rendering performance improvements
    - Deferred rendering
    - Distance adaptive LOD
    - Pixel shader optimizations
    - 1.7 ms → 1.2 ms
    - With LOD, 0.3 ms (or lower) at a distance

TRESSFX 3.X

- TressFX 3.0
  - Maya plugin
    - with full source
  - Viewer and runtime library
    - with full source
  - Fur support
  - Skinning
  - Free
- TressFX 3.1
  - New ShortCut OIT method
    - Lower memory
    - Potentially better performance
    - But at slightly reduced quality vs. standard PPLL TressFX

TRESSFX 3.X VIEWER AND RUNTIME LIBRARY

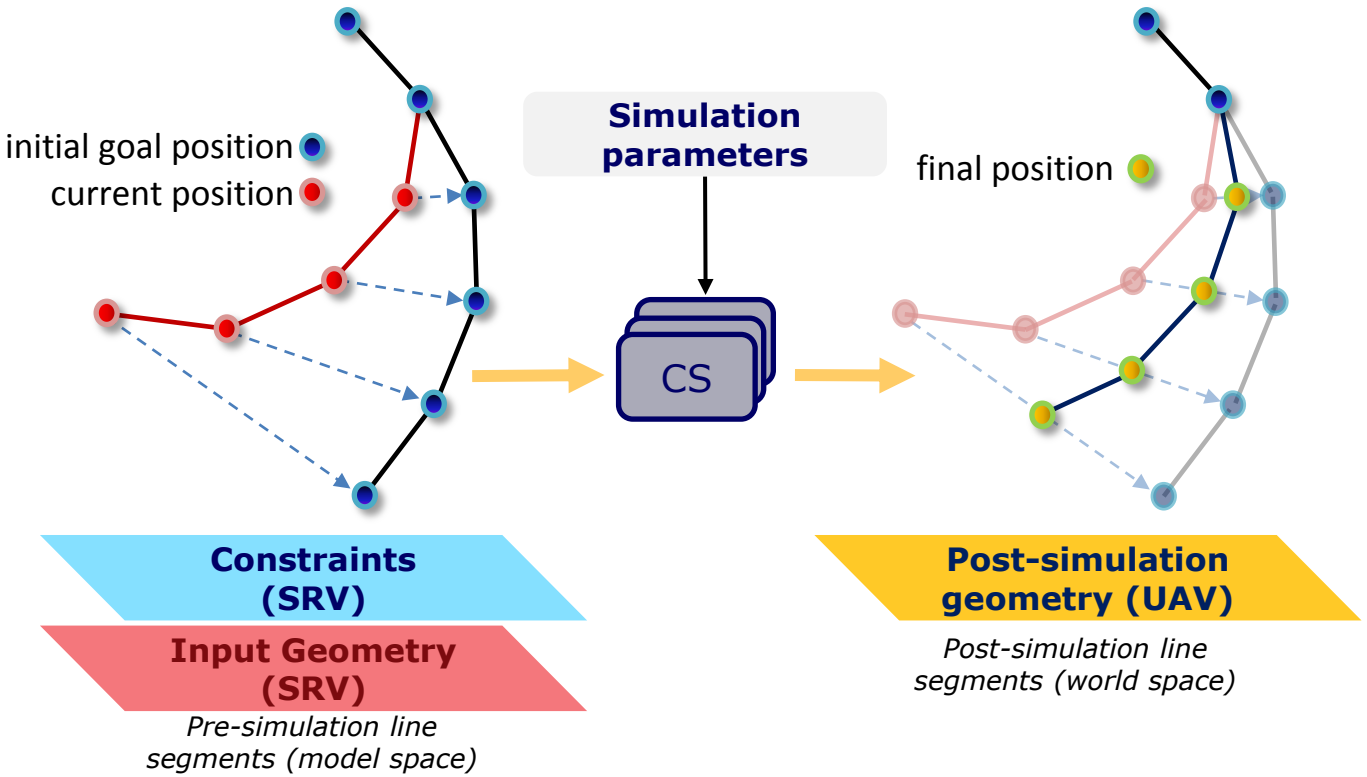
- Viewer provided
  - Preview TressFX assets
- Runtime reorganized into a library
  - Easier integration
- Full source
  - Free
  - No black boxes



TRESSFX 3.X FUR SUPPORT

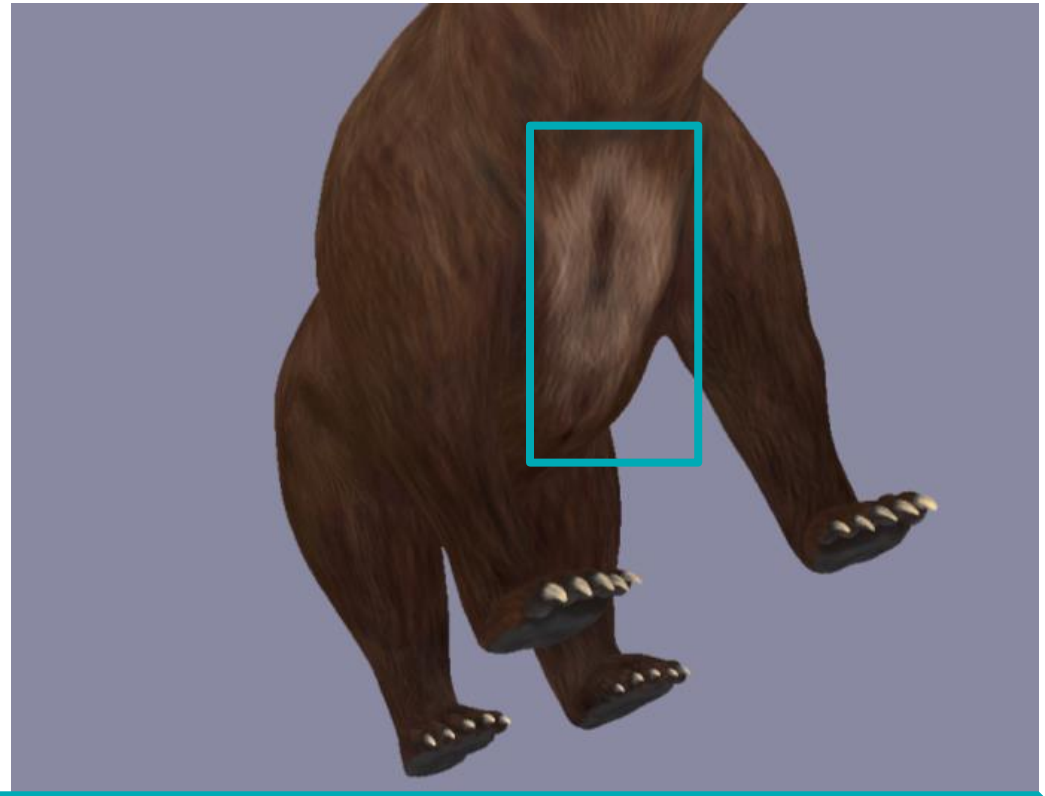
- Core principles are the same
  - Simulation on GPU using compute
  - Good lighting
  - Anti-aliasing
  - Volume shadows
  - Transparency
- Texture coordinates
  - Allows variation in fur color
- Skinning
  - A simple head transform was enough for human hair
  - Fur requires skinning support

TRESSFX 3.X FUR SIMULATION

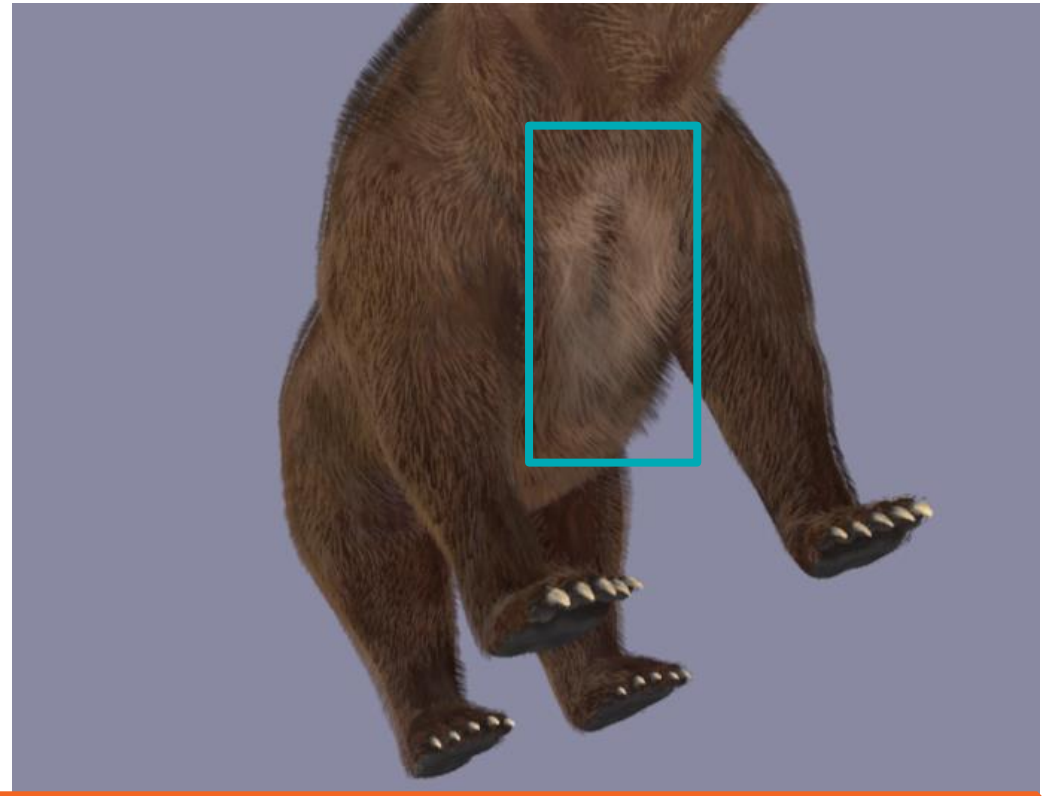


- SIMULATION COMPUTE SHADERS
  - Edge length constraint
  - Local shape constraint
  - Global shape constraint
  - Model Transform
  - Collision Shape
  - External Forces (wind, gravity, etc.)

TRESSFX 3.X TEXTURE COORDINATES

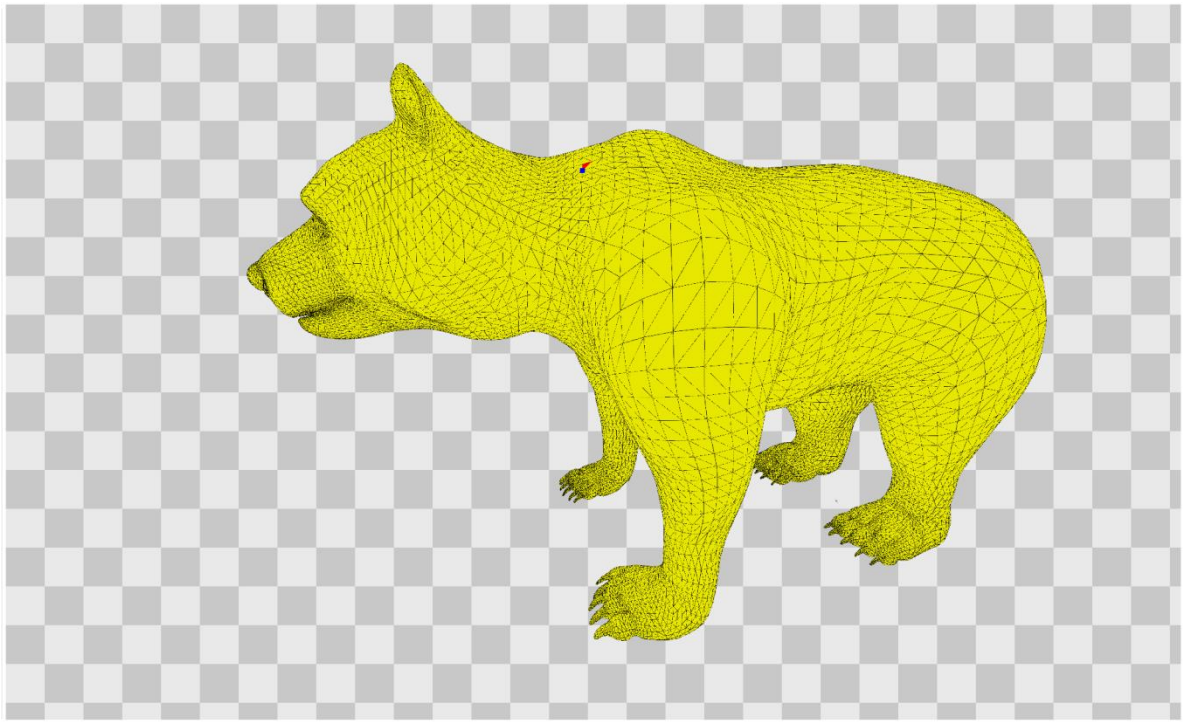


TEXTURED MESH

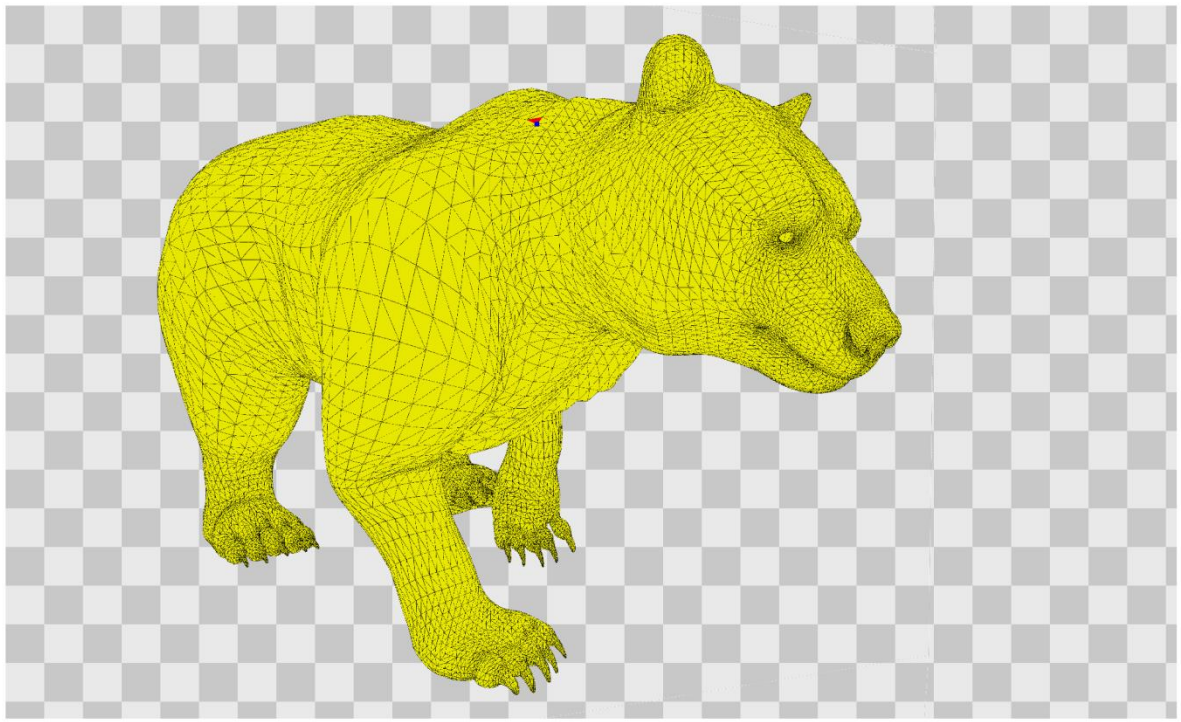


FUR PICKS UP COLOR VARIATION FROM TEXTURE

TRESSFX 3.X SKINNING



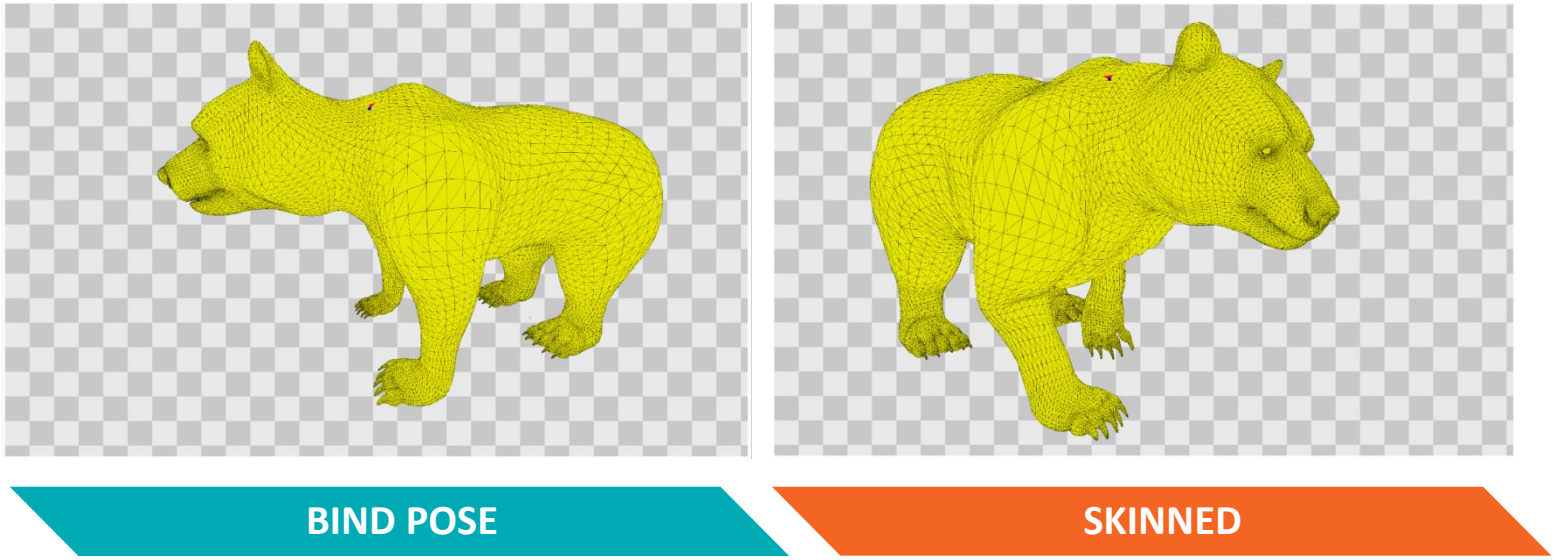
BIND POSE



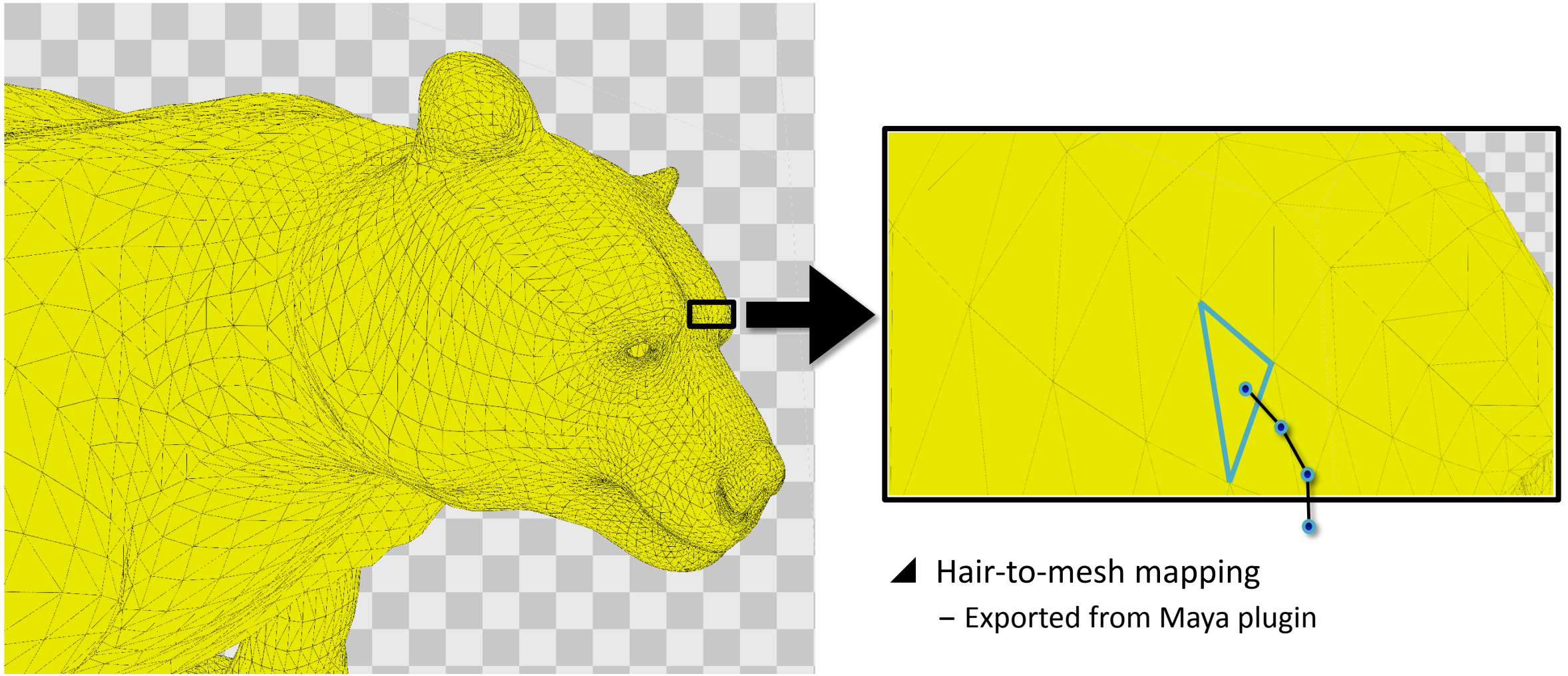
SKINNED

TRESSFX 3.X SKINNING

- How to get skinned vertex position data to TressFX library?
- Up to you
- TressFX viewer currently uses Stream Out
  - DirectX 11
  - No geometry shader
  - See “Getting Started with the Stream-Output Stage” on MSDN
- UAVs at vertex shader stage
  - DirectX 11.1
  - DirectX 12



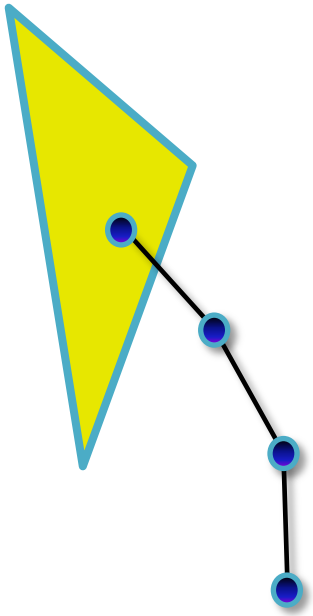
TRESSFX 3.X SKINNING



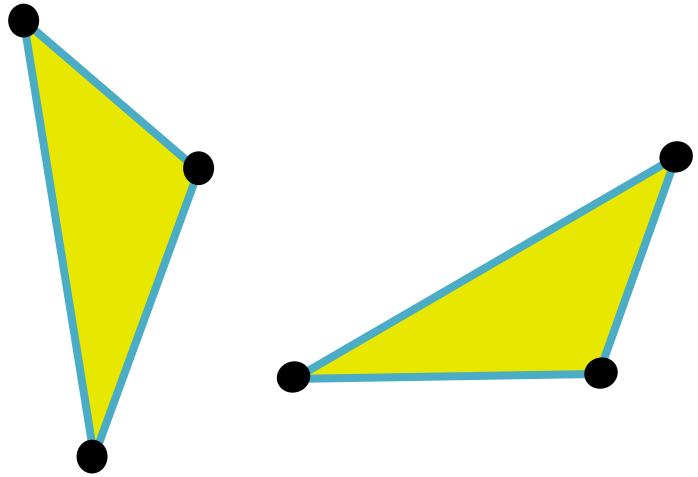
- Hair-to-mesh mapping
  - Exported from Maya plugin

TRESSFX 3.X SKINNING

- Use hair strand index to get triangle index



- Use triangle index to get bind-pose verts and skinned verts

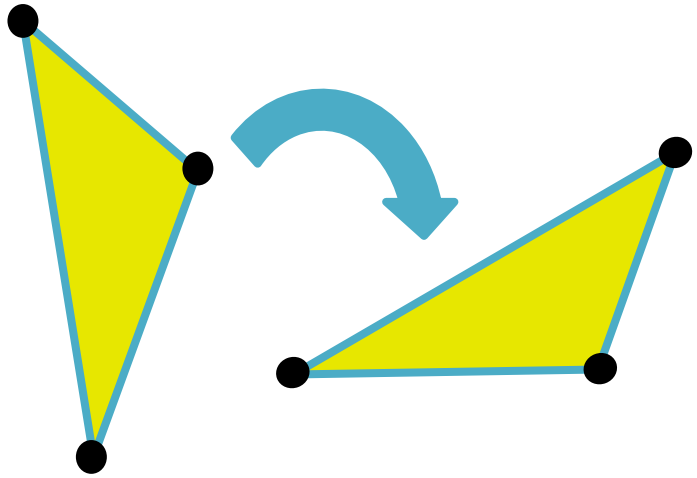


Bind-pose vertex positions

Skinned vertex positions

TRESSFX 3.X SKINNING

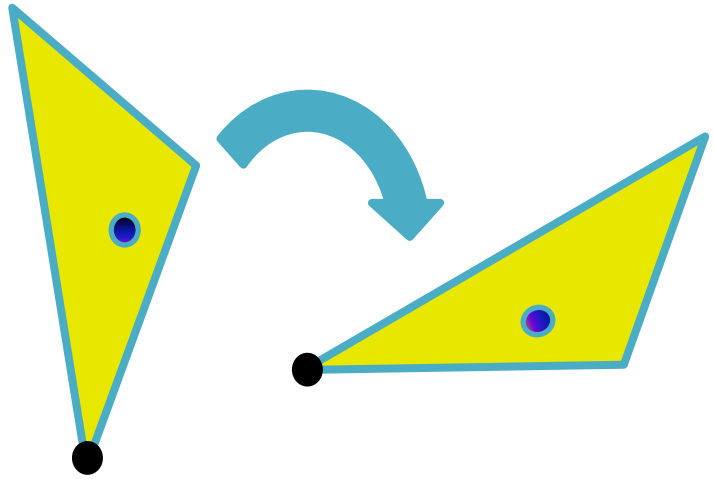
- Calculate transform from bind-pose to skinned



Bind-pose vertex positions

Skinned vertex positions

- Use barycentric coordinates for hair root to calculate final hair transform

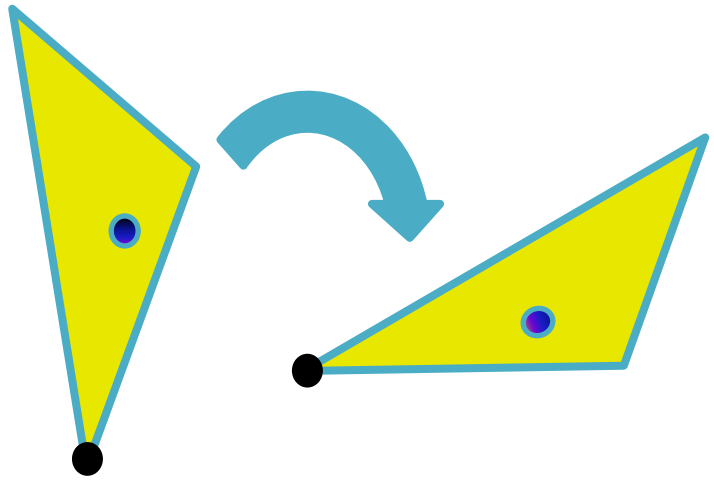


Bind-pose vertex positions

Skinned vertex positions

TRESSFX 3.X SKINNING

- Yeah, okay, but why not just skin the hair directly?
- This way doesn’t impose any requirements on how the game engine does the animation update
  - Morph targets/blend shapes
  - Whatever, we just need the updated vertex positions
- But may code a fast path for ordinary skinning with max 4 bones



TRESSFX 3.X OPTIMIZATIONS

- Already in TressFX 2.2
  - Master and slave strands
  - Distance adaptive LOD
  - Deferred rendering
  - Lots of shader optimizations
- Coming in TressFX 3.x
  - Depth pre-pass
  - Adjust $K_{overdraw}$
  - More shader optimizations

TRESSFX 3.1 *SHORTCUT* RENDERING

- A second OIT implementation provided with the 3.1 release
  - Inspired by Eidos-Montréal's GDC 2015 talk
  - Enabled in the sample with the ShortCut checkbox
- Alpha is computed from all hair fragments, while color is based on shading a few front layers
- Comparison with current PPLL method
  - Memory requirements reduced due to small and fixed number of layers
  - Relative performance depends on depth complexity
    - Requires an extra geometry and screen pass
    - Potential savings from processing fewer layers
  - Result not quite the same quality as PPLL method

DISCLAIMER & ATTRIBUTION


AMD