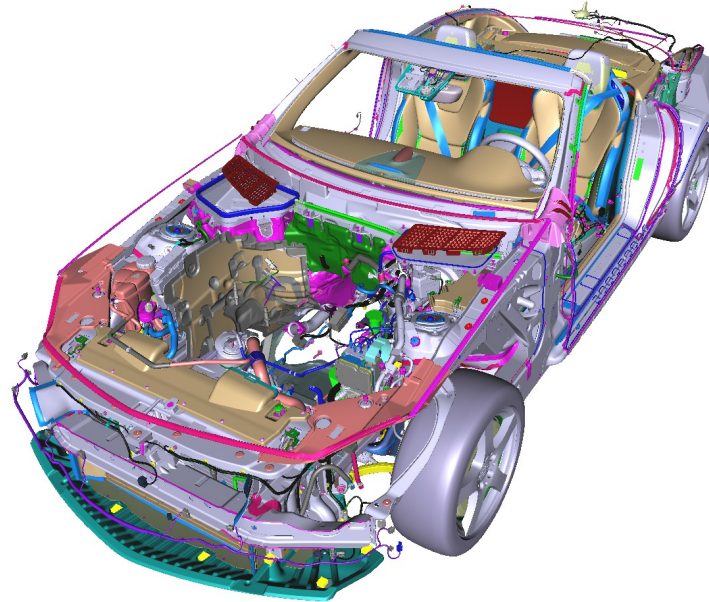


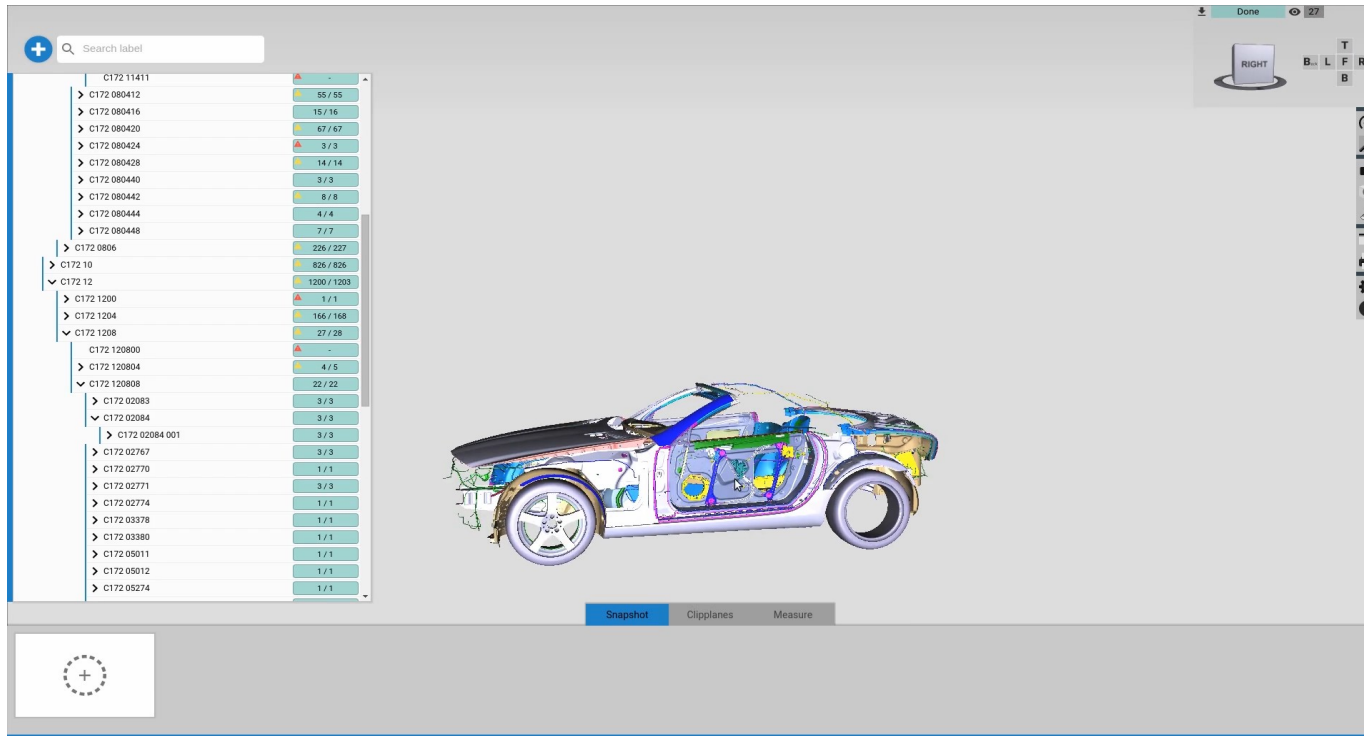
High-Performance Visualization of Massive CAD Data with WebGL



WebGL and glTF BOF @ SIGGRAPH 2015

Christian Stein, Maik Thöner, Max Limper, Johannes Behr
VCST Group, Fraunhofer IGD

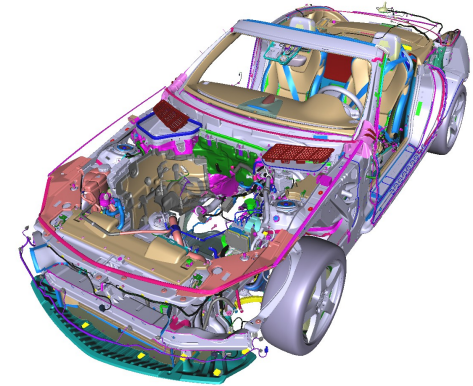
webVis / instant3Dhub



Data Transcoding

Typical Automotive CAD Data Set:

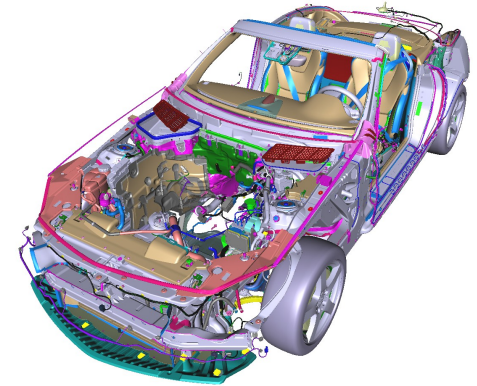
- 20-60 Mio. triangles
- **8-12 GB** in orig. CAD format (e.g., OpenJT)



Data Transcoding

Typical Automotive CAD Data Set:

- 20-60 Mio. triangles
- **8-12 GB** in orig. CAD format (e.g., OpenJT)



Geometry After Transcoding:

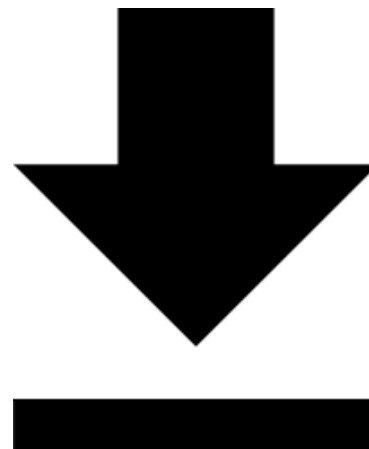
- Quantized, Stripified, GPU-friendly encoding
- **150-700 MB, ~8-12 bytes / triangle**

Parsing, Culling, Loading

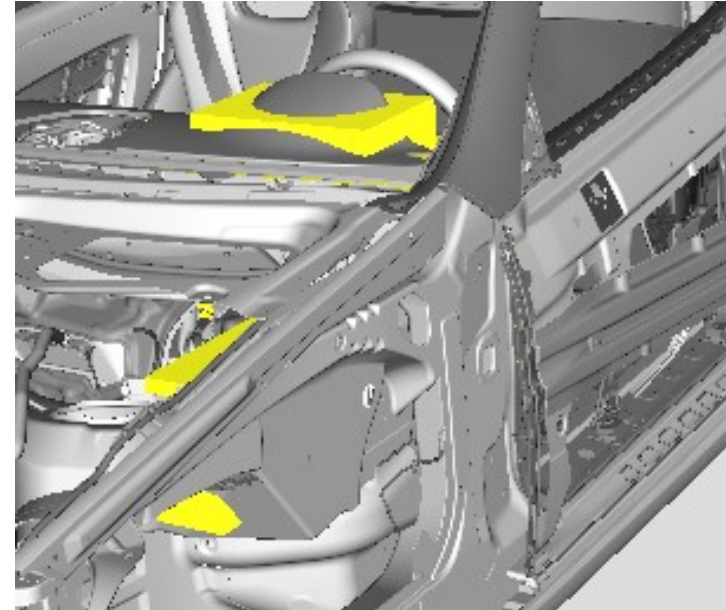
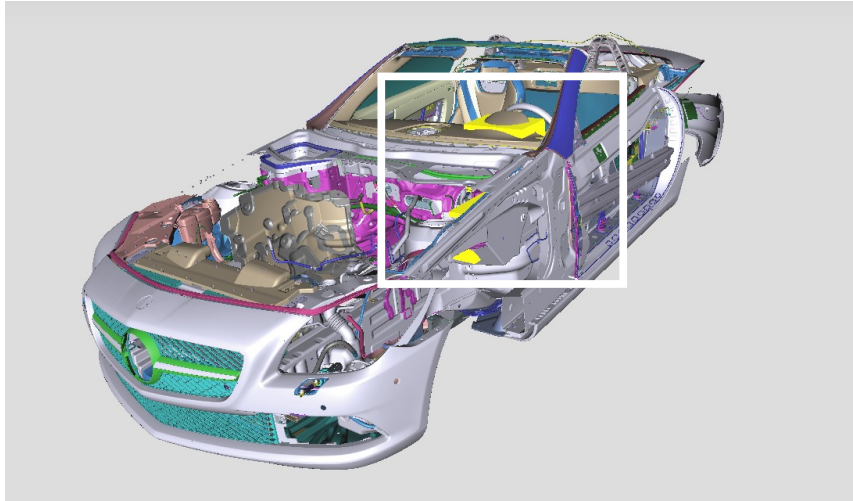
Still: How can we conveniently process a 500MB* data set?

(* transmission: ~250MB using gzip)

- Do *everything* on a budget
 - structure parsing
 - spatial index traversal
 - GPU uploads
 - rendering
- Load largest visible parts first (→ bounding volumes)
- Visual feedback, provide usable intermediate results

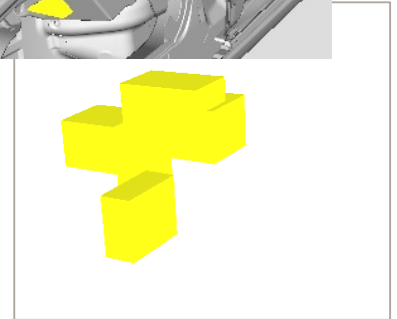
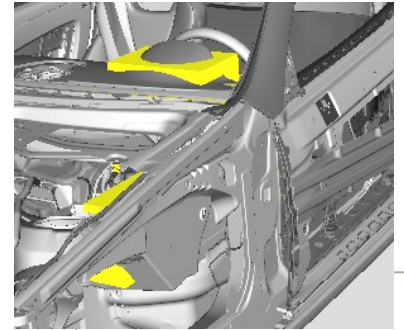


Visual Feedback: Example

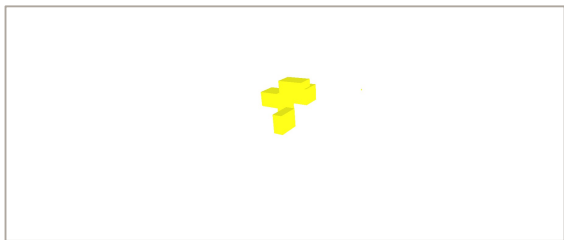


Flexible Rendering Pipeline Configuration

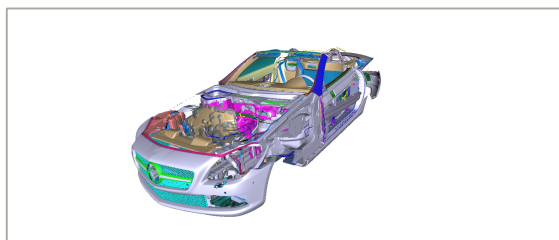
- Always try to avoid re-drawing from scratch
- Challenge: Dynamic auxiliary geometry (user enables / disables, downloads finish, ...)
- hare3d: Flexible configuration of pipeline stages / slots



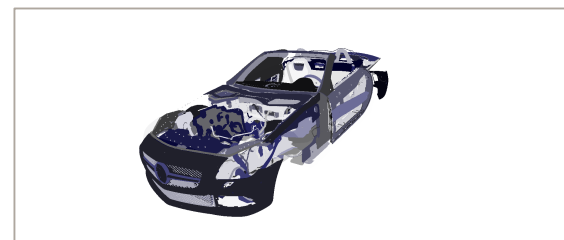
Flexible Rendering Pipeline Configuration



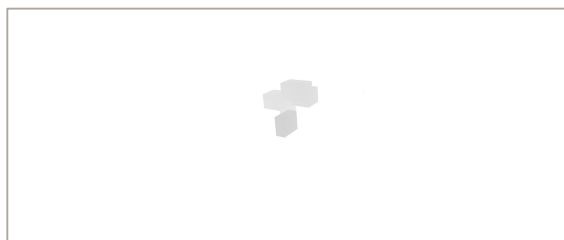
Auxiliary Color



Main Color



Part IDs

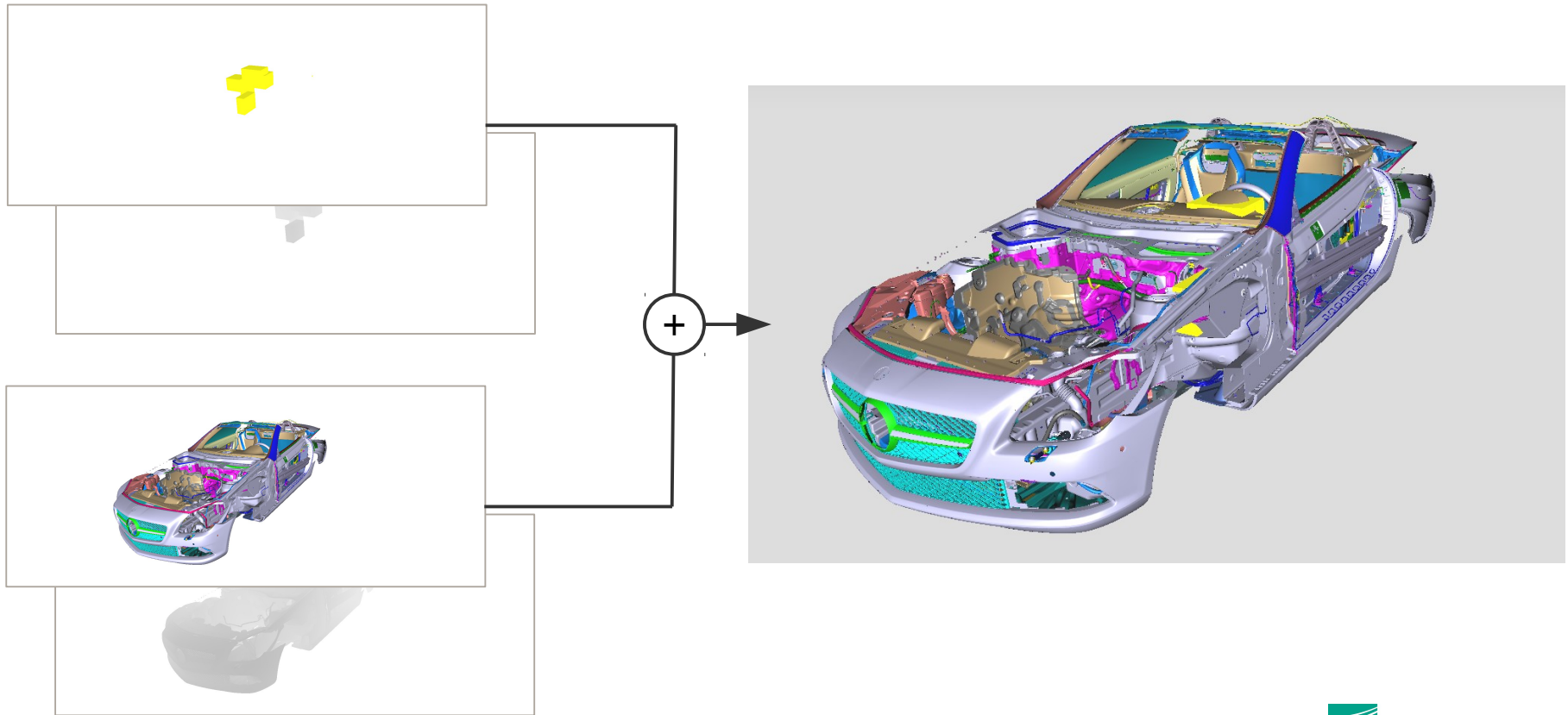


Auxiliary Depth



Main Depth

Flexible Rendering Pipeline Configuration



Work in Progress

- Memory management / replacement strategies
- Use hardware occlusion queries with iterative rendering
- Screen-space techniques
(e.g., deferred shading / highlighting)

Demo Time!

Context: webVis / instant3Dhub

