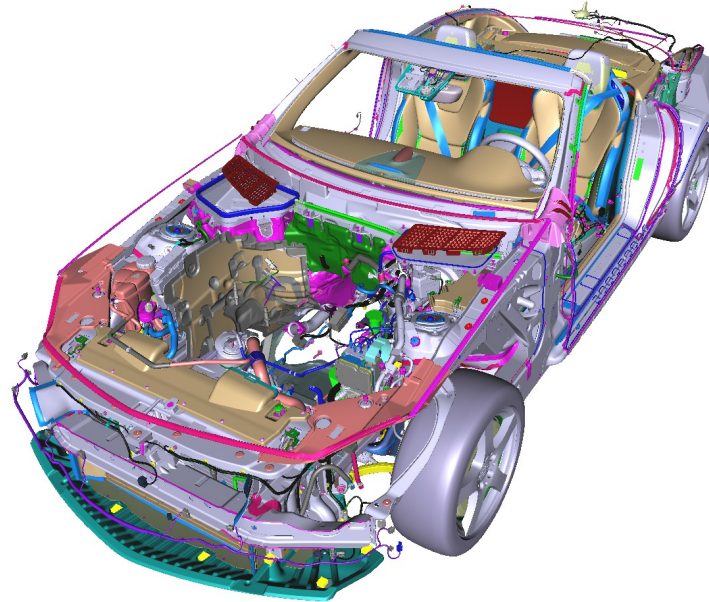


# High-Performance Visualization of Massive CAD Data with WebGL



WebGL and glTF BOF @ SIGGRAPH 2015

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

# webVis / instant3Dhub

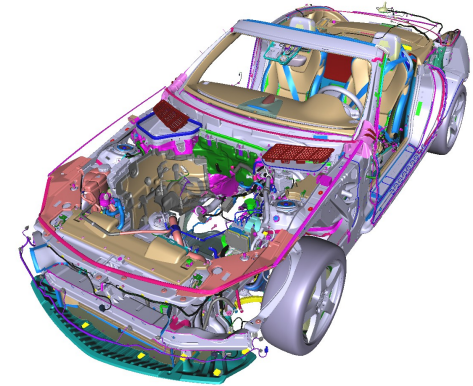
The screenshot displays the instant3Dhub webVis interface. On the left, a hierarchical tree lists components with their IDs and counts. The central area shows a 3D CAD model of a car chassis with various parts highlighted in different colors. The right side features a toolbar with navigation and manipulation tools. At the bottom, there are buttons for 'Snapshot', 'Clipplanes', and 'Measure'.

Component ID	Count
C172 11411	-
> C172 080412	55 / 55
> C172 080416	15 / 16
> C172 080420	67 / 67
> C172 080424	3 / 3
> C172 080428	14 / 14
> C172 080440	3 / 3
> C172 080442	8 / 8
> C172 080444	4 / 4
> C172 080448	7 / 7
> C172 0806	226 / 227
> C172 10	826 / 826
> C172 12	1200 / 1203
> C172 1200	1 / 1
> C172 1204	166 / 168
> C172 1208	27 / 28
> C172 120800	-
> C172 120804	4 / 5
> C172 120808	22 / 22
> C172 02083	3 / 3
> C172 02084	3 / 3
> C172 02084 001	3 / 3
> C172 02767	3 / 3
> C172 02770	1 / 1
> C172 02771	3 / 3
> C172 02774	1 / 1
> C172 03378	1 / 1
> C172 03380	1 / 1
> C172 05011	1 / 1
> C172 05012	1 / 1
> C172 05274	1 / 1

# Data Transcoding

## Typical Automotive CAD Data Set:

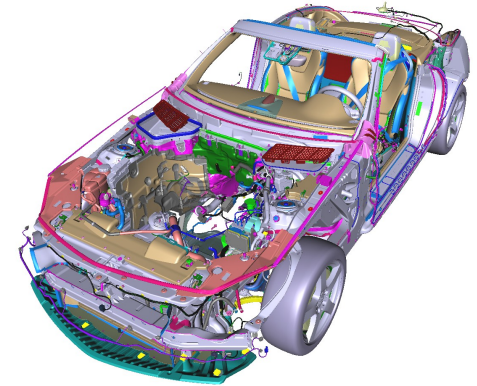
- 20-60 Mio. triangles
- **8-12 GB** in orig. CAD format (e.g., OpenJT)



# Data Transcoding

## Typical Automotive CAD Data Set:

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



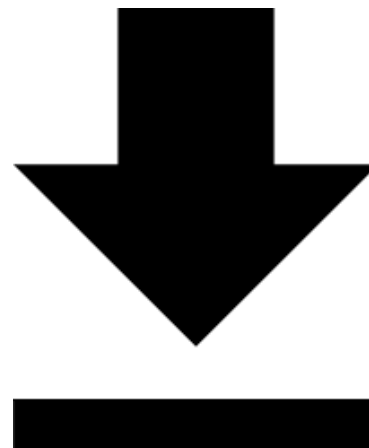
## Geometry After Transcoding:

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

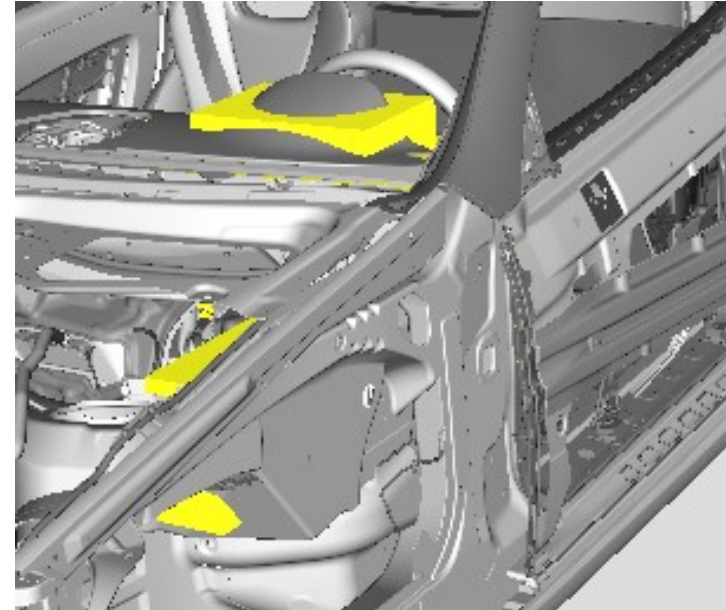
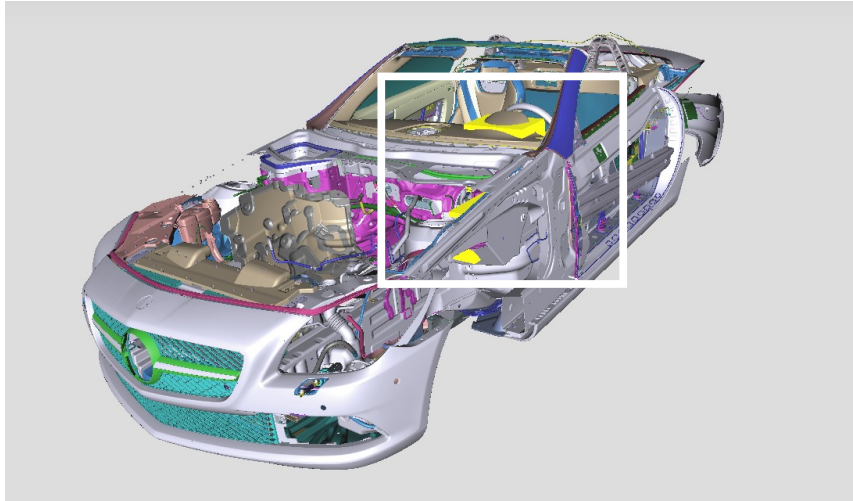
# Parsing, Culling, Loading

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

- 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

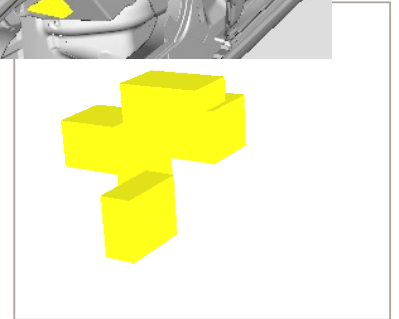
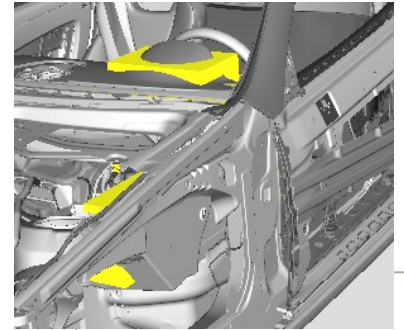


# Visual Feedback: Example

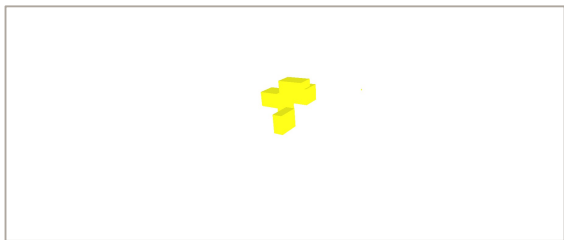


# Flexible Rendering Pipeline Configuration

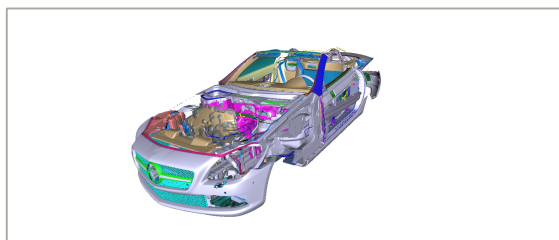
- Always try to avoid re-drawing from scratch
- Challenge: Dynamic auxiliary geometry (user enables / disables, downloads finish, ...)
- hare3d: Flexible configuration of pipeline stages / slots



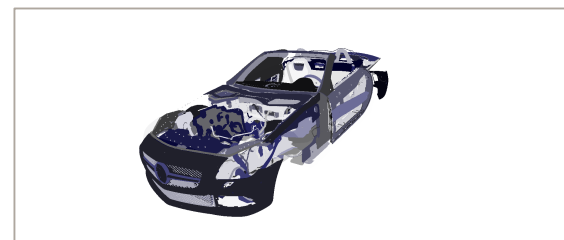
# Flexible Rendering Pipeline Configuration



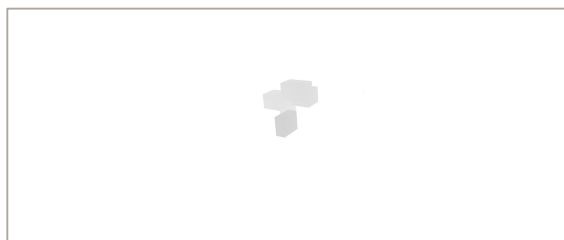
Auxiliary Color



Main Color



Part IDs



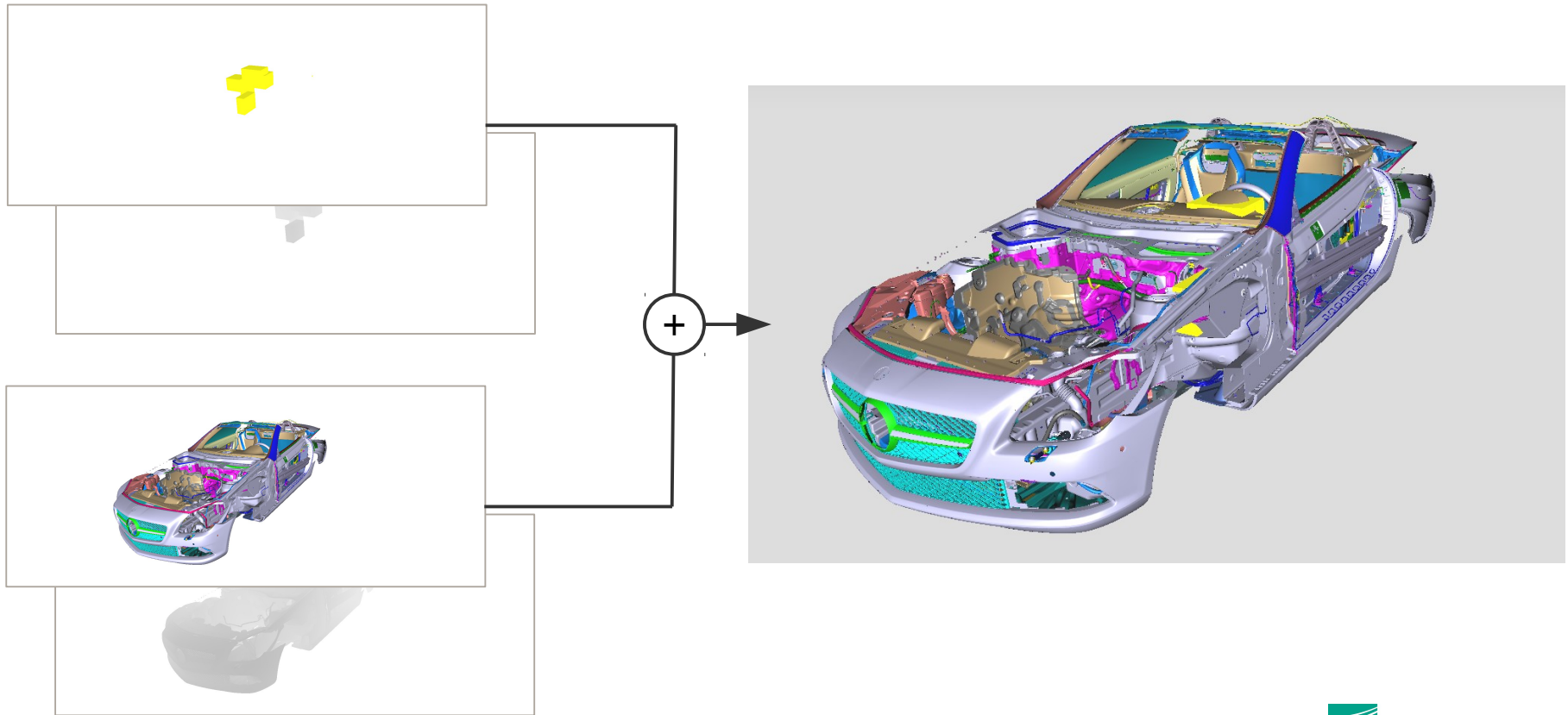
Auxiliary Depth



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

