gltF 2.0 in Google Search

Adrian Perez
July 2019
Who I am and why I’m here

- Hello, I’m Adrian from Google
- Previously: Bungie, Suckerpunch
- I’m here on behalf of the team that delivered the “View in 3D / View in your space” capability in Search
Who I am and why I’m here

- Our team had domain experience having shipped Sceneform in 2018
- Sceneform had a lot of challenges delivering on its vision via .obj / .fbx
- The glTF format addresses many of these issues, and for us it’s a match made in heaven.
What is Sceneform

- Helps non-experts make AR activities on Android
- App side: a easy to use java API using Android idioms (scene management, material modification, hit testing, UI helpers, &c)
- Tool side: A wizard for importing .obj/.fbx files, a viewer, and build rules for Android Studio / Gradle
- Both built on Filament, a modern PBR renderer
- Many 1p/3p clients, e.g. Google Maps, Target.
Challenges achieving Sceneform’s vision

• Creating automatic metallic-roughness PBR materials from an obj/fbx is really subjective
  - Stingray materials made the best case better, but can’t be a requirement
  - Ex: some use contexts ignore per-vertex color, so some assets have it in there as an authoring aid
  - OBJ files are a try-your-best, provide after-the-fact editing (e.g. which edges are hard? transparency?)

• Loading obj/fbx on device is really impractical
  - Our solution was to build a runtime format (.sfb) which e.g. combined encoded images into a single flatbuffer structure

• Delivering (rather than packing in) our runtime format is really unpalatable
  - Our primary clients want to put known-quantity assets in their CDN and be done with it
  - It can be made to work (Google Maps), but is a hassle (e.g. we need to customize the delivered asset per-version because of small material incompatibilities as features get added to Filament)
glTF 2.0 To the Rescue

• No need for per-asset compiled materials
  - Permutation space is much smaller
  - Lighting mode, Blending mode, &c
  - Material pack can be downloaded once and re-used

• No need for asset pre-processing
  - Format is simple enough to parse live
  - Mild filament fixups (quaternion tangent spaces, normalized blend weights)

• No need to get in the way
  - We built a fuzzed, sandboxed loader with tight runtime/binary constraints
  - gltf/glb’s can live wherever and delivered however and be loaded safely.
  - Loaded data is driven by Sceneform UX code
Next Steps for glTF @ Google

- Make our sandboxed glTF loader available in Sceneform
  - Loads 5-10x faster than our existing glTF loader and can safely load downloaded assets
- Work to address remaining ambiguities
  - ADOBE_materials_thin_transparency (or a similar glass-supporting transparency model) would remove a stumbling block for our partners
- Address corner cases with our implementation
  - Prioritized by partner requests
- Wider platform support
  - E.g. a small/fast loader works great in WASM
Thank You!

- https://developers.google.com/ar/develop/java/sceneform/
- https://github.com/google/filament
- https://careers.google.com/jobs/