v3dv: experience using gfxreconstruct/apitrace traces for performance evaluation

Alejandro Piñeiro Iglesias <apinheiro@igalia.com>

v3dv checkpoints

- [Aug 20] Minimal Vulkan 1.0 feature set
- [Nov 20] Vulkan 1.0 conformant
- [Nov 20] Started to test on real world apps
- [Dec 20] Started to work on performance
- [Dec 21] Vulkan 1.1 conformant
- [Jul 22] Vulkan 1.2 conformant

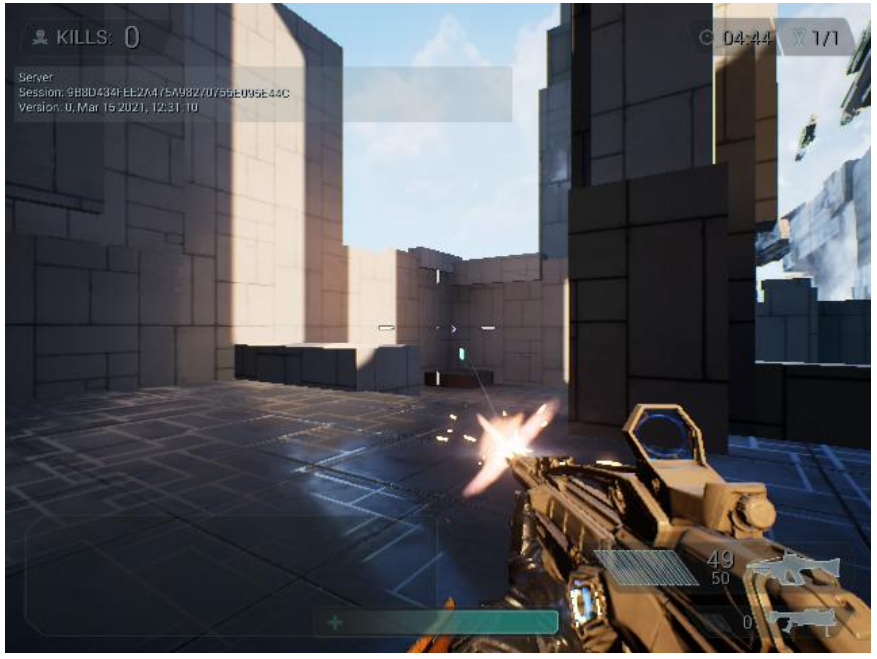
igalia

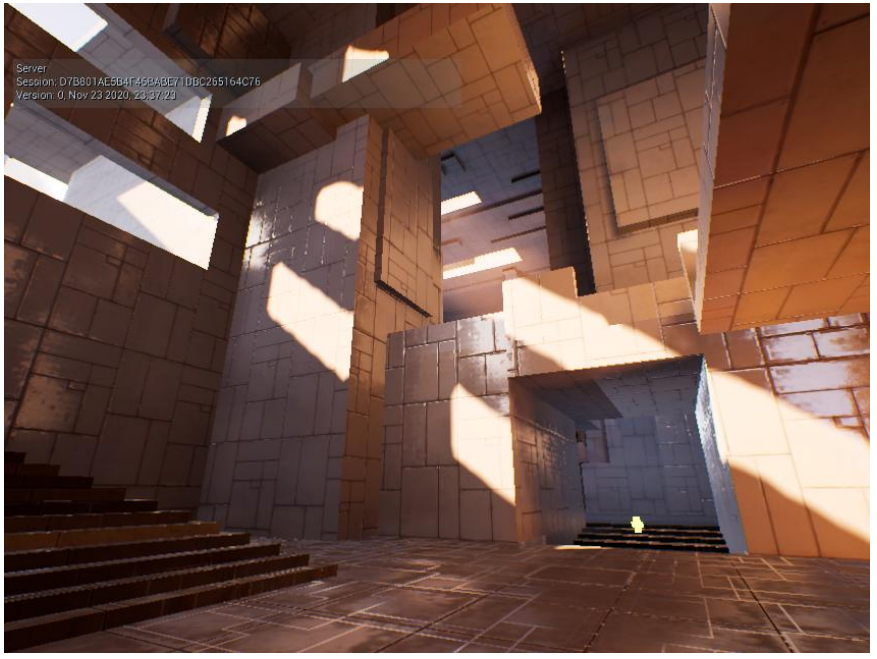
Performance work

- Driven initially by native Unreal Engine 4 samples
  - Generally GPU limited
  - Very expensive shading
- Focus on backend shader code optimizations
  - OpenGL/ES driver also benefited

igalia

Performance work – UE4 samples





igalia

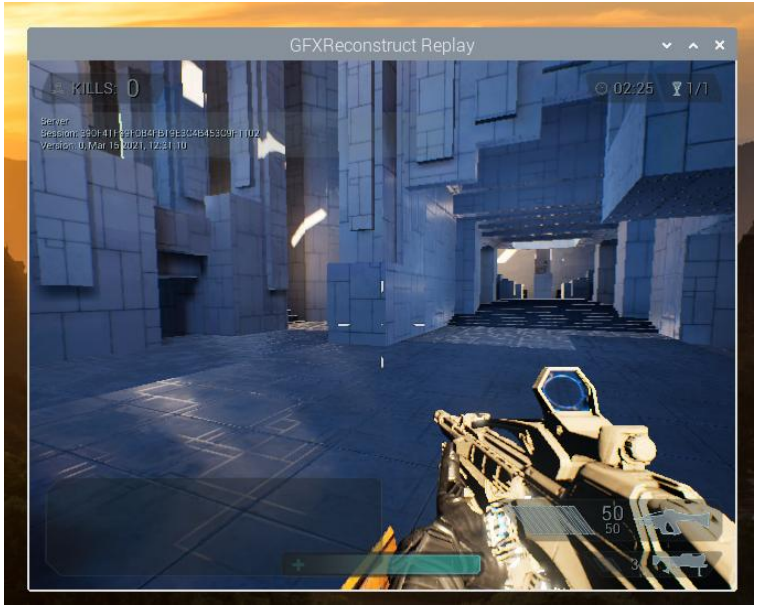
Performance work

- Process:
  - Capture generated shader code
  - Identify non optimal code traces
  - Figure out how that code is generated
  - Design & Implement optimizations
  - Verify results:
    - shader-db + **GFXReconstruct/apitrace** + manual testing

igalia

GFXReconstruct/apitrace

- GFXReconstruct/apitrace



igalia

GFXReconstruct/apitrace

- Two main uses:
  - Visual checking for artifacts
  - Performance comparison based on final FPS
- Automatized through an script
  - Run all traces from a directory
  - Can pass a list of mesa commits for comparisons
  - Several additional parameters

igalia

Pros

- Visual checking easier
  - CI already includes checking for a small amount of frames, but not enough
  - More practical that needing to fake play games
- FPS easy to understand, gave clear values on the early stages of performance work

igalia

Cons

- FPS unreliable, already known as a flawed performance metric
- Right now each hypothetical performance improvement has a small individual impact
  - At this moment it is used as a sanity check

Contact

- IRC: #videocore@OFTC
- Mailing list: mesa-devel@freedesktop.org
- Gitlab: https://gitlab.freedesktop.org/mesa/mesa
- Blogs:

  https://blogs.igalia.com/itoral

  https://blogs.igalia.com/apinheiro

igalia

Q&A

We are hiring: www.igalia.com/jobs