Contribution ID: **40** Type: **not specified**

Improving frame timing accuracy in Mesa, DRM and X

Thursday, October 3, 2019 10:45 AM (45 minutes)

Smooth animation of graphics requires that the presentation timing of each frame be controlled accurately by the application so that the contents can be correctly adjusted for the display time. Controlling the presentation timing involves predicting when rendering of the frame will be complete and using the display API to request that the frame be displayed at a specific time.

Predicting the time it will take to render a frame usually draws upon historical frame rendering times along with application heuristics. Once drawn, the display API is given the job of presenting the content to the user at the specified time. A failure of either of these two mechanisms will result in content being delayed, and a stuttering or judder artifact made visible to the user.

Historical timing information includes both the time taken to render a frame with the GPU along with the actual time each frame was displayed to the user. Ideally, the application will also be given some estimate of how long it will take to ready the frame for display once the presentation request has been delivered to the display system. With these three pieces of information (application GPU time, actual display time, presentation overhead), the application can estimate when its next frame will be ready for display.

The following work is underway to provide applications this information and to improve the accuracy of display presentation timing in the Linux environment.

1. Vulkan GOOGLE_display_timing extension implementation in Mesa. This offers applications some fairly straightforward measurements that can help predict when a frame timing target might be missed.
2. Heuristics in the X Composite and Present extension implementations to improve accuracy of reported display times to Present-using applications
3. Additions to Composite that replace the above heuristics with precise timing information for Compositing managers modified to support these additions.
4. Semi-automatic compositing support added to the Composite extension which allow in-server compositing of some windows to reduce variability in the display process.

This presentation will describe the above work and demonstrate the benefits of the resulting code.

Code of Conduct

Yes

GSoC, EVoC or Outreachy

No

Presenter: PACKARD, Keith (Valve)

Session Classification: Main Track

Track Classification: Talk (full slot) (closed)