Subject: Re: Simple way to develope 2D Game
Posted by mirek on Wed, 12 Jun 2019 07:19:38 GMT
View Forum Message <> Reply to Message

Novo wrote on Wed, 12 June 2019 02:20mirek wrote on Tue, 11 June 2019 17:02It is however pretty hard, GPUs are not well suited for this, so the work was postoponed.
A couple of examples of what can be done.

Check this app. It is a Flash Player with hardware acceleration.

Well, sorry for original poster for being off-topic... Anyway:

It all seems to depend on "this" in above quote. If "this" is defined as "reproduce the Painter polygon semantics with 90% accuracy", then it really is a problem and tesselation is at the heart of it, the other issue being batching / OpenGL state changes.

It simply might be less problematic to render polygon with 100000 vertices in software than to tessalate it, send all vertices to GPU and then render.

Even more trouble: As what I draw is often map polygons, I have even tried the tactics of: tesselate map polygon once, store it in the GPU and then draw from data in GPU. Guess what: This is still slower than software rendering, because by the time OpenGL is finished with state changes to draw the polygon, software renderer is already done with that. So it appears that as long as you are doing glDrawElements in equivalent of Painter::Fill / Painter::Stroke, you have already lost. Means ideally you need to do single glDrawElements for the whole rendering, less ideally but still useably single glDrawElements per many Fill/Strokes.

That said, I have it in "postponed" state now. For now, I got stuck at "I need faster tesselator than tess2, but at the same time I need tesselator that is as accurate as tess2". While there are faster tesselators (e.g. https://github.com/mapbox/earcut.hpp), they are not as accurate, trivial polygon examples tend to fail with them. I have tried to implement my own, but the work is still in progress...

Mirek