Subject: Re: Optimized storage of 1BPP images
Posted by mr_ped on Thu, 07 Feb 2008 16:13:50 GMT
View Forum Message <> Reply to Message

1728x2210x50 pages in RGBA is whopping 700MB, that's bad.

But, the tiff itself is only 1BPP? So in such case the file has approximately 23MB only! (no problem to cache it)

If you manage to produce (rescale) final viewport from these raw data, you need less than 10MB for viewport (2000x1200 pixels) content in RGBA. (that's 23 + 10 = ~33MB so far .. that's nothing for modern PC, but of course there will be additional memory consumption caused by OS itself and underlaying technologies like U++)

So if you prefer to rescale the viewport out from RGBA data (may save you some coding of your own scale routines for 1BPP -> RGBA scaling), you may still unpack+cache only 2-3 pages from the raw tiff, and work with those. Predict when new page will be approximately needed, and unpack in background thread soon enough into cache the next page.
This will be additional 15MB per cached RGBA page. (and I think cache of size 3 or 4 is plenty unless somebody want to scroll really really fast trough all pages) 4*15 = 60; 60 + 33 = 93MB. Still well under 100MB for core application data. Doesn't sound bad to me.

But you will get somewhat complex caching/unpacking/rescaling code. I don't think the complexity is beyond single focused developer, but depends how much time you can spend on this application development.
I believe to read the new page from disk is question only couple of tenths of second, the unpack into RGBA original size picture is under 0.2s for sure and all of this could be fitted into background thread to be finished before you will need that page first time.

The only performance-wise problematic place IMO is the on-the-fly production of viewport data. This can choke a bit on slower CPUs with very large window area. Than again if you will cache even this during scrolling, so you will scale during scrolling only newly visible area, the slow few lines per sec scrolling can be done already on 1GHz CPU.
If somebody want to go trough pages fast with PageDown, you will have to decode the 1BPP pages on the fly ... still if I would get 0.1 - 0.3s response on (many) PageDown hits, I would be not happy, nor sad. Depends how your users use this viewer most often.
If they go all around the page slowly in high zoom, it should work very smoothly. If they go very fast down, it will hurt.

Than again, the KPDF on my old 1GHz Athlon XP is pain in *ss to go trough 30-50 pages PDF with page down, often it takes several seconds to show content of current page, if I go down too fast beyond the pre-cached pages. (than again, PDF is much more complex thing than raw 1BPP image)