Subject: Antigrain author on text rasterisation
Posted by Mindtraveller on Mon, 24 Jan 2011 14:07:03 GMT
View Forum Message <> Reply to Message

Finally someone explained why Apple fonts are good, why M$ fonts are worse and Linux fonts are ugly.
http://www.antigrain.com/research/font_rasterization/index.h tml
Author proposes good approach to make rendered fonts good looking.
May be it will be interesting as U++ has it's own text rendering.

Subject: Re: Antigrain author on text rasterisation
Posted by chickenk on Mon, 24 Jan 2011 15:14:24 GMT
View Forum Message <> Reply to Message

Indeed, the author's article is very interesting, I read it a while ago. This was one of the reason I also asked about antigrain integration in U++, and Mirek gave this request the best anwser: he created Painter, which introduces the concepts of antigrain without the downsides.

Maybe Mirek took some inspiration for fonts rendering in Painter from this article as well ?

Subject: Re: Antigrain author on text rasterisation
Posted by dolik.rce on Mon, 24 Jan 2011 16:31:39 GMT
View Forum Message <> Reply to Message

Interesting article, thanks Mindtraveller.

It raises many question about how the U++ text rendering works. (And I am too busy/lazy to look search for the answers in the code). So, is the U++ font rendering used everywhere (i.e. in Draw) or only in Painter? Does U++ use win api? (I already know that it does use FreeFont based Xft on X11...). What approach was chosen in Painter? There is subpixel positioning available, so is it used on texts too?

Sorry for so many questions, but the article woke up my curiosity

Best regards,
Honza

Subject: Re: Antigrain author on text rasterisation
Posted by fudadmin on Mon, 24 Jan 2011 16:58:23 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 24 January 2011 16:31Interesting article, thanks Mindtraveller.

It raises many question about how the U++ text rendering works. (And I am too busy/lazy to look search for the answers in the code). So, is the U++ font rendering used everywhere (i.e. in Draw) or only in Painter? Does U++ use win api? (I already know that it does use FreeFont based Xft on X11...). What approach was chosen in Painter? There is subpixel positioning available, so is it used on texts too?

Sorry for so many questions, but the article woke up my curiosity

Best regards,
Honza

First of all, Mindtraveller, that article is 4 years old (2007?). agg was the reason for Painter (I think, if you have studied our forums)

UPP::Painter = (2% ? agg) + (mirek's brain juice)
Everything else is ugly.

honza, for your reference from UPP DrawTextWin32.cpp:

```
#ifdef PLATFORM_WIN32

#define LLOG(x)

HFONT GetWin32Font(Font fnt, int angle);

void SystemDraw::DrawTextOp(int x, int y, int angle, const wchar *text, Font font, Color ink,
                            int n, const int *dx) {
 Std(font);
 while(n > 30000) {
  DrawTextOp(x, y, angle, text, font, ink, 30000, dx);
  if(dx) {
   for(int i = 0; i < 30000; i++)
    x += *dx++;
  }
  else
   x += GetTextSize(text, font, 30000).cx;
  n -= 30000;
  text += 30000;
 }
 GuiLock __;
 COLORREF cr = GetColor(ink);
 if(cr != lastTextColor) {
  LLOG("Setting text color: " << ink);
  ::SetTextColor(handle, lastTextColor = cr);
 }
 HGDIOBJ orgfont = ::SelectObject(handle, GetWin32Font(font, angle));
 int ascent = font.Info().GetAscent();
 if(angle) {
  double sina, cosa;
```

```
  Draw::SinCos(angle, sina, cosa);
  Size offset;
  ::ExtTextOutW(handle, x + fround(ascent * sina), y + fround(ascent * cosa), 0, NULL, (const
WCHAR *)text, n, dx);
 }
 else
  ::ExtTextOutW(handle, x, y + ascent, 0, NULL, (const WCHAR *)text,
                n, dx);
 ::SelectObject(handle, orgfont);
}

#endif
```

On the other hand, many people with LCD screens still don't know how to enable and tune antialiasing on their Windows or Linux systems...

Subject: Re: Antigrain author on text rasterisation
Posted by Mindtraveller on Mon, 24 Jan 2011 17:12:51 GMT
View Forum Message <> Reply to Message

Of course I'm aware that Mirek used Antigrain and derived Painter package. That is why I posted this link above. My idea was - if U++ will draw text with vertical-only hinting, it will improve visual quality on ALL platforms.

Subject: Re: Antigrain author on text rasterisation
Posted by mirek on Mon, 24 Jan 2011 23:24:39 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 24 January 2011 11:31
So, is the U++ font rendering used everywhere (i.e. in Draw) or only in Painter?

SystemDraw is using host platform facilities to render text.

Quote:
 Does U++ use win api? (I already know that it does use FreeFont based Xft on X11...).

Painter uses only winapi to get font outlines, same is done by FreeType in X11. See Painter/FontWin32 and Painter/FontX11

Quote:
 What approach was chosen in Painter? There is subpixel positioning available, so is it used on

texts too?

Actually, Painter does not distinguish (at least at the moment) text from any other painting.

And yes, you can request subpixel rendering in Painter. In PainterExamples, there is even GUI switch to activate it (so you can freely study differences ...

Subject: Re: Antigrain author on text rasterisation
Posted by mirek on Mon, 24 Jan 2011 23:33:37 GMT
View Forum Message <> Reply to Message

Mindtraveller wrote on Mon, 24 January 2011 12:12Of course I'm aware that Mirek used Antigrain and derived Painter package.

Actually, while I have seen Antigrain sources, I have only recycled about 30 lines (interestigly, those fetching font glyphs in X11 and Win32).

Generally I think AGG approach is somewhat inflexible in the real life, for no good performance gain (we are faster anyway

Quote:
That is why I posted this link above. My idea was - if U++ will draw text with vertical-only hinting, it will improve visual quality on ALL platforms.

Well, maybe we can try. But I am not so sure about it...
