Interesting and confusing. Frankly, looks like driver bug.

That said, U++ does some weird things to reduce the required bandwidth and memory footprint. Namely, it splits large images into horizontal bands and scans those bands for uniform color areas. Those areas, if any, are then painted using DrawRect.

I believe that what we see on screenshot is that some of that optimization somehow collapsed or got recomputed wrongly (band part started too much in the left). I think that would point to driver bug.

Well, I plan to buy some cheap Epson ink-jet to check this out. (Unfortunately, it has to be Epson - no other is doing 'odd' DPI printers anymore

Quote:
Just an opinion, but I think the best printing quality can be obtained using the printer's hardware resolution (and optionally dropping it with integer dividers if it is excessively high). For correct page size and scaling I always use Draw::GetPagePixels() and Draw::GetPixelsPerInch() to get the map scale right on both screen and printer.

IMO, as long as we abstract from the particular image rescaling problem, the difference should not be next to nonexistent, for drawing text and polygons. After all, some coordinate rescaling is involved in either case...

I think we should aim for fixing the DrawImageOp routine - it should not be hard to switch to "physical mode" for the time the Image is printed, then switch back to normal "dots" mode, rescaling coordinates and target image size in the process...

Mirek