1. Open a large grayscale image of your choice (e.g. larger than 2000\*2000 pixels, maybe a scan result from your scanner)

2. Use \"Tools/Color Tools/Threshold\" to apply some threshold choosen arbitrarily.

This tool is pretty fast.

3. Now you have a 8bit grayscale image, which actually consists only of color values \"0\" and color values \"255\". Converting it to a 1bit black&white image should be a very fast operation (replacing 0 (8bit) by 0 (1bit) and replacing 255 (8bit) by 1 (1bit)), but it is not:

4. Choose \"Image/Mode/Indexed...\", and there:

4.1 Colormap: \"Use black and white (1-bit) palette)\" (this should be the fastest option possible)

4.2 Dithering: \"none\" (this should be the fastest option possible)

4.3 Now hit \"OK\"

5. The conversion will work, but it will take many seconds.

For an example image of \"2460\*3483\", it takes 34 seconds on my computer (while applying the Threshold tool, which basically does the same, only needs maybe 2 seconds). This slow speed is not acceptable for interactive image processing, and this slowness is not necessary at all. Indeed it is very slow, so I will confirm this bug report. There is, however, a workaround. I found with a little experimentation that if you use the \"Optimal Palette\" option for step 4.1, then the conversion occurs in a flash, literally a fraction of a second for me. This happens, I think, because if you use this option, by the time the algorithm reaches the stage of actually converting the image, it has built a color histogram and \"knows\" that there are exactly two colors in the image, and can assign colors directly, without having to test whether black or white is the best match for each pixel. Because of the existence of this workaround, I am going to lower the severity, but this \*is\* something that should be looked into. Created an attachment (id=38911) [details]. Image where to 1bit conversion is either slow or buggy (with GIMP 2.2.4). That's odd. I just tried to use your suggested workaround and was happy that the conversion was as fast one should expect. However, the conversion was plainly wrong, it produced an image which was just black (the input image was nearly white). I'm attaching a test case image. This may be a related bug. Do following on the atached image: 1. Choose \"Image/Mode/Indexed...\", and there: 1.1 Colormap: optimal palette (wether 2 oder 256 colors, it does not matter)

1.2 Dithering: \"none\" (but I think it does not matter). Interesting -- clearly the generation of an optimal palette is not done correctly for this image. Because this is a different issue, I am going to open a different bug report for it -- using your image as example, if you don't mind. In any case, I've found a solution to the original problem, I think. I found a very simple change that gives a major speedup when converting grayscale images to the mono palette. It's now at least as fast as the Threshold tool, so I'm going to resolve this bug as FIXED. Still just as slow for RGB images, though. Committed to both branches: 2005-03-18 Bill Skaggs &lt;weskaggs@primate.ucdavis.edu&gt; \* app/core/gimpimage-convert.c: when converting grayscale to mono, treat the palette as gray rather than rgb, giving more than tenfold speedup. Fixes bug #170801. The new problem is described in bug #170825. I'd be pleased if you used my test case image. Thank you for fixing this bug so fast, so I just need to wait for the next release. (Building for Windows would be rather messy, so I try to be patient...) . I'm interested in seeing the patch. If you can't attach it then what revision of gimpimage-convert.c contains the fix? Revision 1.156. Here is the patch: --- gimpimage-convert.c 27 Feb 2005 14:03:50 -0000 1.155; +++ gimpimage-convert.c 18 Mar 2005 17:34:34 -0000 1.156; @@ -4057,7 +4057,6 @@ initialize\_median\_cut (GimpImageBaseType; if (palette\_type == GIMP\_WEB\_PALETTE ||; - palette\_type == GIMP\_MONO\_PALETTE ||; palette\_type == GIMP\_CUSTOM\_PALETTE); {; switch (dither\_type). I expect it to yield poorer results though, to some extent (probably most obviously darker, or lighter, in mid-greys, that is a damaged gamma), because the greyscale colour-matching and error-quantifying is in a comparatively non-perceptually-uniform space (that's half the reason why it's faster, though it's not as big a quality difference in a greyscale source as an RGB source). Were side-by-side comparisons done? I hope so, though I could do some here if anyone cares. I don't think anyone has done side-by-side comparisons. differences, but not so strong that I'm really going to lose any sleep. Note that the analysis in comment #1 is correct; the 'optimal' palette finder does a first pass to see if there are already a suitable number of colours, and then knows that it can skip both the expensive quantization and dithering stages. The 'mono' palette option doesn't even bother to start this pre-pass because it could only possibly pay off the extra effort if the entire image is pure black and pure white, which is expected to be a comparatively rare occurance. &gt; The 'mono' palette option doesn't even bother to start this pre-pass because it could only possibly pay off the extra effort if the entire image is pure black and pure white, which is expected to be a comparatively rare occurance. I don't think that this operation is so rare, because for getting an \"optimal distribution between black and white\" for grayscale images (for example: you want to generate faximilies the receiver actually is able to read), you use the; \"threshold\" operation, and then a \"convert to 1bit\" operation to actually adjust the internal memory requirements. So at least for me, this chain of operations is rather frequent. &gt; and then a \"convert to 1bit\" operation to actually adjust the internal memory requirements. If you mean GIMP's internal memory requirements you're wasting your time -- all Indexed images are 8/16bpp internally (the same as greyscale images), so you're not saving any memory (instead you may be using quite a lot more, for various I mean OpenOffice's internal memory requirements. For having 10 or 20 fax pages in one OOo document, it quite makes a difference wether they are 8bit or 1bit.