

cairo-gl

Eric Anholt

What raster backends are there?

- xlib: built on XRender
- xcb: built on XRender, but better.
- image: built on pixman
- glitz: built on an abstraction of XRender, now deleted
- drm: experiment by Chris Wilson to do custom Cairo drivers

Why not Xlib?

- Half of it was never implemented
- Slow

Why not image?

- Most of us want to look at the graphics we draw
- The cost of uploading image results to the screen

Why not glitz?

- Glitz abstracted my abstraction layer
- Few performance wins, some performance hits
- Unmaintained

Why not drm?

- I'm a driver developer.
- I'm lazy.
- It's really hard.

Introducing cairo-gl

- Direct-rendered cairo on proven hardware abstraction
- Everyone needs a good GL driver anyway
- Implementation is simple.
 - cairo-gl: 6362 LOC (+43889 LOC in i965 3D driver)
 - cairo-xlib: 8317 LOC (+3956 LOC in i965 2D driver)
 - cairo-drm-i965: 15982 LOC
- Maybe we can make cairo as fast as everyone hopes it can be.

How does it work

- Cairo has 5 basic operations:
 - paint
 - mask
 - fill
 - stroke
 - glyphs

mask

- Basic operation for cairo backends
- Source is a color, gradient, or texture
- Mask is a color, gradient, or texture
- Multiply source by mask alpha and blend
- First operation supported by cairo-gl

paint

- This is mask without a mask
- We don't implement it so cairo makes it into a mask call

fill and stroke

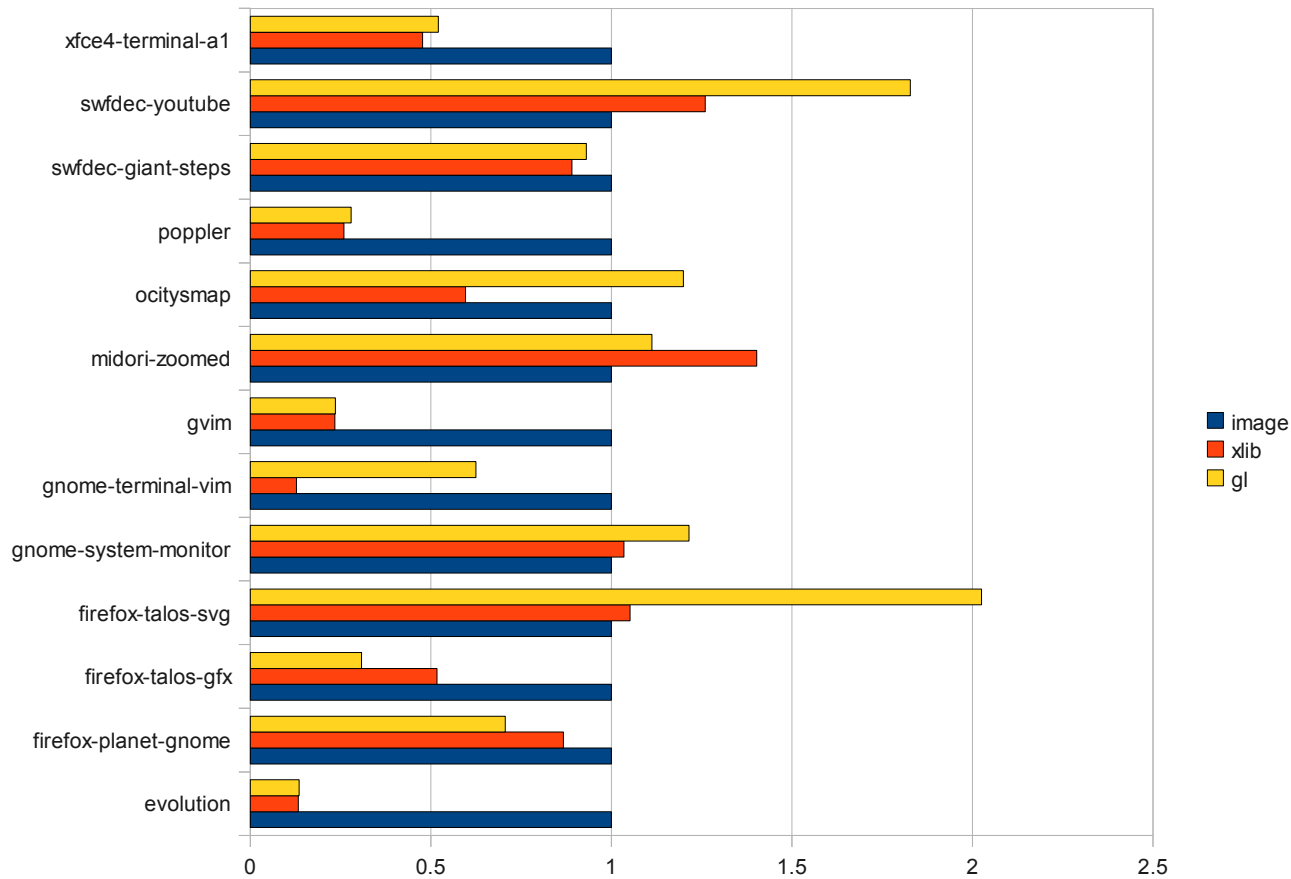
- Cairo computes a polygon outline
- Cairo tessellates the polygon outline to spans
 - x1
 - x2
 - y
 - coverage
- Cairo-gl takes spans and emits rectangles with coverage in the color attribute
- The GPU likes this: lots of vertices with the same state

glyphs

- App uses pango or something to lay out text
- Cairo gets a list of glyph IDs and positions to draw
- Cairo-gi loads the glyphs from the font into a texture atlas
- Emits rectangles loading the mask from the texture atlas
- Lots of vertices, the GPU is good with this

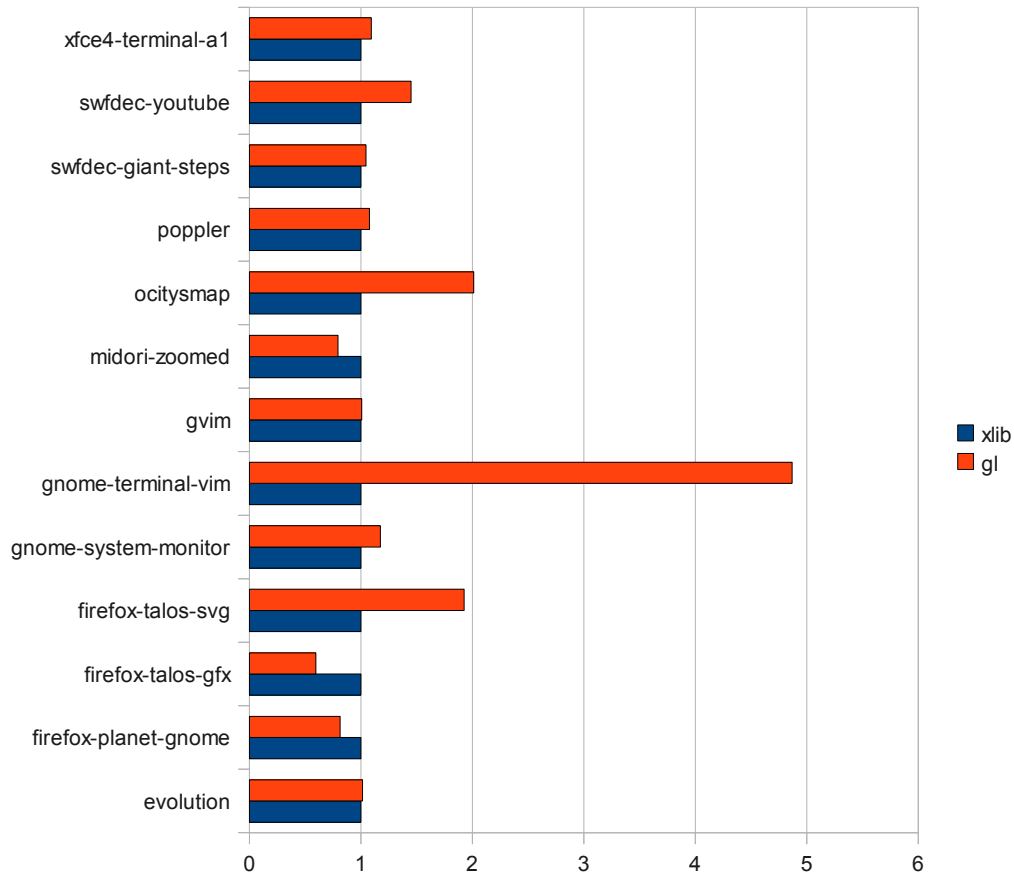
How is performance?

- bigger is better



A more exciting chart

- bigger is better



What's still painful?

- 2D is all about glyphs (firefox-talos-gfx)

System Profiler (as superuser)

Profiler View Help

Start Profile Save As

Samples: 19173

Functions	Self	Total
cairo_show_glyphs	0.06 %	75.07 %
_cairo_gstate_show_text_glyphs	0.11 %	74.95 %
_cairo_surface_show_text_glyphs	0.04 %	73.53 %
_cairo_gl_surface_show_glyphs	0.50 %	73.37 %
glDrawArrays	0.11 %	43.32 %
neutral_DrawArrays	0.16 %	43.20 %
vbo_exec_DrawArrays	0.14 %	42.90 %
_clip_and_composite	0.06 %	40.59 %
brw_draw_prims	1.26 %	37.03 %
_cairo_surface_composite	0.04 %	36.67 %
_cairo_surface_mask	0.03 %	36.54 %
_cairo_surface_fallback_mask	0.06 %	36.44 %
_cairo_gl_surface_composite	0.21 %	36.27 %

Descendants	Self	Cumulative
glDrawArrays	0.11 %	43.32 %
neutral_DrawArrays	0.11 %	43.16 %
vbo_exec_DrawArrays	0.11 %	42.88 %
brw_draw_prims	1.20 %	36.98 %
brw_validate_state	0.81 %	21.23 %
_intel_batchbuffer_flush	0.00 %	5.60 %
brw_upload_state	0.69 %	5.30 %
intel_batchbuffer_reset	0.00 %	1.55 %
brw_validate_textures	0.10 %	0.58 %
drm_intel_bufmgr_check_aperture_space	0.00 %	0.35 %
drm_intel_bo_unreference	0.09 %	0.20 %
vbo_all_varyings_in_vbos	0.09 %	0.09 %
brw_clear_validated_bos	0.08 %	0.08 %
intel_batchbuffer_data	0.04 %	0.06 %
drm_intel_gem_check_aperture_space	0.05 %	0.05 %
upload_sf_vp	0.05 %	0.05 %
drm_intel_bo_reference	0.05 %	0.05 %
intel_finalize_mipmap_tree	0.05 %	0.05 %
brw_upload_vs_prog	0.04 %	0.04 %
upload_sf_prog	0.04 %	0.04 %
__i686.get_pc_thunk.bx	0.03 %	0.03 %
prepare_binding_table_pointers	0.03 %	0.03 %
brw_prepare_vertices	0.03 %	0.03 %
upload_clip_unit	0.03 %	0.03 %
brw_state_cache_check_size	0.02 %	0.02 %

Callers	Self	Total
_cairo_gl_surface_composite	0.04 %	28.25 %
_cairo_gl_glyphs_draw	0.01 %	13.16 %
_cairo_gl_span_renderer_flush	0.00 %	1.00 %
_cairo_gl_surface_fill_rectangles	0.00 %	0.86 %
_cairo_surface_composite	0.04 %	0.04 %
_cairo_gl_flush_glyphs	0.02 %	0.02 %
_cairo_gl_surface_span_renderer_finish	0.01 %	0.01 %

What's still painful

- gradients too (firefox-talos-svg)

System Profiler (as superuser)

Profiler View Help

Start Profile Save As

Samples: 20723

Functions	Self	Total
_composite_spans_draw_func	0.01 %	63.99 %
_cairo_surface_composite_polygon	0.01 %	63.96 %
_cairo_surface_fill	0.01 %	59.43 %
_cairo_surface_fallback_fill	0.04 %	59.37 %
_cairo_gstate_fill	0.03 %	59.36 %
cairo_fill_preserve	0.00 %	59.36 %
_cairo_tor_scan_converter_gener...	18.84 %	34.75 %
_cairo_gl_operand_init	0.02 %	20.77 %
_cairo_gl_surface_create_span_re...	0.04 %	20.24 %
_cairo_surface_create_span_rend...	0.00 %	20.24 %
_cairo_gl_create_gradient_texture	18.58 %	19.82 %
glDrawArrays	0.01 %	18.53 %
neutral_DrawArrays	0.03 %	18.51 %
sub_oxc_DrawArrays	0.05 %	18.45 %

Callers	Self	Total
_clip_and_composite	0.01 %	63.83 %
_create_composite_mask_pattern	0.00 %	0.15 %
_cairo_surface_fallback_fill	0.00 %	0.00 %

Descendants	Self	Cumulative
_composite_spans_draw_func	0.01 %	63.99 %
_cairo_surface_composite_polygon	0.01 %	63.96 %
_cairo_tor_scan_converter_generate	18.84 %	34.75 %
_cairo_gl_render_bounded_spans	0.46 %	10.93 %
_cairo_gl_emit_rectangle	0.99 %	9.44 %
_cairo_gl_emit_span_vertex	0.83 %	0.83 %
__i686.get_pc_thunk.bx	0.17 %	0.17 %
glBufferDataARB	0.02 %	0.02 %
glBindBufferARB	0.00 %	0.00 %
_cairo_gl_render_unbounded_spans	0.15 %	3.44 %
cell_list_render_edge	1.27 %	1.27 %
_cairo_gl_emit_rectangle	0.11 %	0.11 %
_cairo_gl_emit_span_vertex	0.06 %	0.06 %
-- kernel --	0.00 %	0.05 %
__i686.get_pc_thunk.bx	0.03 %	0.03 %
malloc	0.00 %	0.00 %
In file /lib/i686/cmov/libc-2.10.2.so	0.00 %	0.00 %
_cairo_surface_create_span_renderer	0.00 %	20.24 %
_cairo_gl_surface_create_span_renderer	0.03 %	20.23 %
_cairo_gl_operand_init	0.01 %	19.35 %
_cairo_gl_create_gradient_texture	18.14 %	19.33 %
glBindBufferARB	0.00 %	0.00 %
glMapBufferARB	0.00 %	0.00 %
glUnmapBufferARB	0.00 %	0.00 %
_cairo_gl_set_src_operand	0.00 %	0.34 %

More experiments

- Move to a smaller transformation matrix
- Transformation matrix and other constants as attributes
- Shorter gradients shaders
- Loop-Blinn design to replace spans
- Replace QUADS with TRIANGLES (index buffer or not?)
 - ARB_primitive_restart?

Other directions

- GLES 2.0 support
- Non-Mesa GL drivers
- Large surfaces

Conclusion

- Faster than today's Xlib
- Easy code to hack on
- Not yet beating CPU drawing on integrated graphics
- Lots of room to improve