

## Re: State of Linux graphics

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2005-09/0078.html>

---

**From:** Keith Packard ([keithp\\_at\\_keithp.com](mailto:keithp_at_keithp.com))

**Date:** 09/01/05

To: Discuss issues related to the xorg tree <xorg@lists.freedesktop.org>

Date: Wed, 31 Aug 2005 20:59:23 -0700

On Wed, 2005-08-31 at 18:58 -0700, Allen Akin wrote:

> On Wed, Aug 31, 2005 at 02:06:54PM -0700, Keith Packard wrote:

> | On Wed, 2005-08-31 at 13:06 -0700, Allen Akin wrote:

> | > ...

> |

> | Right, the goal is to have only one driver for the hardware, whether an

> | X server for simple 2D only environments or a GL driver for 2D/3D

> | environments. ...

>

> I count two drivers there; I was hoping the goal was for one. :-)

Yeah, two systems, but (I hope) only one used for each card. So far, I'm not sure of the value of attempting to provide a mostly-software GL implementation in place of existing X drivers.

> | ... I think the only questions here are about the road from

> | where we are to that final goal.

>

> Well there are other questions, including whether it's correct to

> partition the world into "2D only" and "2D/3D" environments. There are

> many disadvantages and few advantages (that I can see) for doing so.

I continue to work on devices for which 3D isn't going to happen. My most recent window system runs on a machine with only 384K of memory, and yet supports a reasonable facsimile of a linux desktop environment.

In the 'real world', we have linux machines continuing to move "down-market" with a target price of \$100. At this price point, it is reasonable to look at what are now considered 'embedded' graphics controllers with no acceleration other than simple copies and fills.

Again, the question is whether a mostly-software OpenGL implementation can effectively compete against the simple X+Render graphics model for basic 2D application operations, and whether there are people interested in even trying to make this happen.

## Linux-Kernel: Re: State of Linux graphics

- > / ... *However, at the*
- > / *application level, GL is not a very friendly 2D application-level API.*
- >
- > *The point of OpenGL is to expose what the vast majority of current*
- > *display hardware does well, and not a lot more. So if a class of apps*
- > *isn't "happy" with the functionality that OpenGL provides, it won't be*
- > *happy with the functionality that any other low-level API provides. The*
- > *problem lies with the hardware.*

Not currently; the OpenGL we have today doesn't provide for component-level compositing or off-screen drawable objects. The former is possible in much modern hardware, and may be exposed in GL through pixel shaders, while the latter spent far too long mired in the ARB and is only now on the radar for implementation in our environment.

Off-screen drawing is the dominant application paradigm in the 2D world, so we can't function without it while component-level compositing provides superior text presentation on LCD screens, which is an obviously increasing segment of the market.

- > *Conversely, if the apps aren't taking advantage of the functionality*
- > *OpenGL provides, they're not exploiting the opportunities the hardware*
- > *offers. Of course I'm not saying all apps \*must\* use all of OpenGL;*
- > *simply that their developers should be aware of exactly what they're*
- > *leaving on the table. It can make the difference between an app that's*
- > *run-of-the-mill and one that's outstanding.*

Most 2D applications aren't all about the presentation on the screen; right now, we're struggling to just get basic office functionality provided to the user. The cairo effort is more about making applications portable to different window systems and printing systems than it is about bling, although the bling does have a strong pull for some developers.

So, my motivation for moving to GL drivers is far more about providing drivers for closed source hardware and reducing developer effort needed to support new hardware than it is about making the desktop graphics faster or more fancy.

- > *"Friendliness" is another matter, and it makes a ton of sense to package*
- > *common functionality in an easier-to-use higher-level library that a lot*
- > *of apps can share. In this discussion my concern isn't with Cairo, but*
- > *with the number and type of back-end APIs we (driver developers and*
- > *library developers and application developers) have to support.*

Right, again the goal is to have only one driver per video card. Right now we're not there, and the result is that the GL drivers take a back seat in most environments to the icky X drivers that are required to provide simple 2D graphics. That's not a happy place to be, and we do want to solve that as soon as possible.

## Linux-Kernel: Re: State of Linux graphics

- > / ... *GL provides*
- > / *far more functionality than we need for 2D applications being designed*
- > / *and implemented today...*
- >
- > *With the exception of lighting, it seems to me that pretty much all of*
- > *that applies to today's "2D" apps. It's just a myth that there's "far*
- > *more" functionality in OpenGL than 2D apps can use. (Especially for*
- > *OpenGL ES, which eliminates legacy cruft from full OpenGL.)*

The bulk of 2D applications need to paint solid rectangles, display a couple of images with a bit of scaling and draw some text. All of the rest of the 3D pipeline is just standing around and watching.

- > / ... *picking the right subset and sticking to that is*
- > / *our current challenge.*
- >
- > *That would be fine with me. I'm more worried about what Render (plus*
- > *EXA?) represents -- a second development path with the actual costs and*
- > *opportunity costs I've mentioned before, and if apps become wedded to it*
- > *(rather than to a higher level like Cairo), a loss of opportunity to*
- > *exploit new features and better performance at the application level.*

I'm not concerned about that; glitz already provides an efficient mapping directly from the Render semantics to OpenGL. And, Render is harder to use than OpenGL in most ways, encouraging applications to use an abstraction layer like cairo which can also easily support GL directly.

- > / ...*The integration of 2D and 3D acceleration into a*
- > / *single GL-based system will take longer, largely as we wait for the GL*
- > / *drivers to catch up to the requirements of the Xgl implementation that*
- > / *we already have.*
- >
- > *Like Jon, I'm concerned that the focus on Render and EXA will*
- > *simultaneously take resources away from and reduce the motivation for*
- > *those drivers.*

Neither of us gets to tell people what code they write, and right now a developer can either spend a week or so switching an XFree86 driver to EXA and have decent Render-based performance or they can stand around and wait for 'some one else' to fix the Mesa/DRI drivers so that we can then port Xgl to them. Given the 'good performance in one week' vs 'stand around and hope GL gets fast enough', it's not hard to see why people are interested in EXA drivers.

Plus, it's not all bad — we're drawing in new developers who are learning about how graphics chips work at a reasonably low level and who may become interested enough to go help with the GL drivers. And, I'm seeing these developers face up to some long-standing DRI issues surrounding memory management. EXA on DRM (the only reasonable EXA architecture in my mind) has all of the same memory management issues

## Linux-Kernel: Re: State of Linux graphics

that DRI should be facing to provide FBO support. Having more eyes and brains looking at this problem can't hurt.

- > / *I'm not sure we have any significant new extensions to create here;*
- > / *we've got a pretty good handle on how X maps to GL and it seems to work*
- > / *well enough with suitable existing extensions.*
- >
- > *I'm glad to hear it, though a bit surprised.*

As I said, glitz shows that given the existing standards, some of which are implemented only in closed-source drivers, we can provide Render semantics in an accelerated fashion.

- > / *This will be an interesting area of research; right now, 2D applications*
- > / *are fairly sketchy about the structure of their UIs, so attempting to*
- > / *wrap them into more structured models will take some effort.*
- >
- > *Game developers have done a surprising amount of work in this area, and*
- > *I know of one company deploying this sort of UI on graphics-accelerated*
- > *cell phones. So some practical experience exists, and we should find a*
- > *way to tap into it.*

Right, it will be interesting to see how this maps into existing 2D toolkits. I suspect that the effects will be reflected up to applications, which won't necessarily be entirely positive.

- > / *Certainly ensuring that cairo on glitz can be used to paint into an*
- > / *arbitrary GL context will go some ways in this direction.*
- >
- > *Yep, that's essential.*
- >
- > / *...So far, 3D driver work has proceeded almost entirely on the*
- > / *newest documented hardware that people could get. Going back and*
- > / *spending months optimizing software 3D rendering code so that it works*
- > / *as fast as software 2D code seems like a thankless task.*
- >
- > *Jon's right about this: If you can accelerate a given simple function*
- > *(blending, say) for a 2D driver, you can accelerate that same function*
- > *in a Mesa driver for a comparable amount of effort, and deliver a*
- > *similar benefit to apps. (More apps, in fact, since it helps*
- > *OpenGL-based apps as well as Cairo-based apps.)*

Yes, you *\*can\**, but the amount of code needed to perform simple pixel-aligned upright blends is a tiny fraction of that needed to deal with filtering textures and *\*then\** blending. All of the compositing code needed for the Render extension, including accelerated (MMX) is implemented in 10K LOC. Optimizing a new case generally involves writing about 50 lines of code or so.

- > *So long as people are encouraged by word and deed to spend their time on*
- > *"2D" drivers, Mesa drivers will be further starved for resources and the*

## Linux-Kernel: Re: State of Linux graphics

> *belief that OpenGL has nothing to offer "2D" apps will become*  
> *self-fulfilling.*

I'm certainly not encouraging them; the exa effort was started by engineers at Trolltech and then spread to the x.org minions. Again, the only encouragement they really need is the simple fact that a few days work yields a tremendous improvement in functionality. You can't get that with a similar effort on the GL front at this point.

> */ So, I believe applications will target the Render API for the*  
> */ foreseeable future. ...*  
>  
> *See above. :-)*

I'll consider Xgl a success if it manages to eliminate 2D drivers from machines capable of supporting OpenGL. Even if the bulk of applications continue to draw using Render and that is translated by X to OpenGL, we will at least have eliminated a huge duplication of effort between 2D and 3D driver development, provided far better acceleration than we have today for Render operations and made it possible for the nasty closed-source vendors to ship working drivers for their latest video cards.

I see the wider availability of OpenGL APIs to be a nice side-effect at this point; applications won't (yet) be able to count on having decent OpenGL performance on every desktop, but as the number of desktops with OpenGL support grows, we will see more and more applications demanding it and getting people to make purchasing decisions based on OpenGL availability for such applications.

Resources for Mesa development are even more constrained than X development, but both of these show signs of improvement, both for social and political reasons within the community as well as economic reasons within corporations. Perhaps someday we really will have enough resources that watching a large number of people get side-tracked with the latest shiny objects won't bother us quite so much.

-keith

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@vger.kernel.org  
More majordomo info at <http://vger.kernel.org/majordomo-info.html>  
Please read the FAQ at <http://www.tux.org/lkml/>

---

- application/pgp-signature attachment: [This is a digitally signed message part](#)