3D in Flex is an exciting field to be working in at the moment. It’s moving very fast: there are several major 3D APIs to choose from, most of which are themselves moving rapidly forward; and developments in the Flash platform itself promise advancements. This does, however, make it a bit tricky to keep up!

3D on the Web

Until the release of Flash Player 10, 3D rendering in Flex was all handled by software. The move to AVM2 (the interpreter for bytecode generated from AS3, as opposed to AVM1, which is for AS2) delivered some significant performance gains.

But it is still many times less powerful than the hardware-accelerated 3D to which we have become accustomed in games for the PC and for consoles. There, the number crunching is handled by the graphics processor, which is optimized to handle these kinds of operations (that’s why graphics cards are sold as “3D” cards).

Knowing your (VM’s) limitations

When everything is done in software, it’s a different story. The calculations required for 3D rendering and animation have to compete for CPU cycles with all the other things your Flex app may be doing. You can’t re-create a full Halo 3– or even Quake-style game in a Flex app and expect the Flash Player to handle the number of polygons...
involved. Results vary depending on processor speed and the 3D library you choose, but don’t expect the Flash Player to handle more than a few thousand polygons without getting sluggish and unresponsive.

Things are set to change somewhat with the forthcoming release of Flash Player 10. Certain basic 3D properties can be set *natively* on sprites and manipulated at runtime: rotations in 3D space, for instance, and shading. This means that the rendering math is taken care of by hardware, which does a much better job of it, leaving the Flash Player free to handle all the other things your Flex app requires.

The 3D capabilities of Flash Player 10, though, are likely to be pretty basic. Don’t expect a fully immersive world, with cameras, z-index management, and so on. You will need to look at one of the software libraries such as Papervision3D to manage that. However, it may be that all your Flex app needs is little touches of 3D to jazz up its look and feel. If so, the native capabilities of Flash Player 10 are likely to be the best option for you.

**Choosing the right tools for the job**

At the other end of the scale, you may be looking to give your users a full-fledged, immersive 3D experience with highly detailed textures and lots of visual complexity. Some use cases for this would be a first-person shooter game, or a Second Life–style virtual world. I would not really recommend the Flash Player (and thus Flex) as the most appropriate platform for this; it just isn’t up to it. You could consider writing for the Unity3D platform; although it has the disadvantage of requiring users to install a browser plug-in that they are unlikely to have already, it does mean that they are getting a “real” 3D experience, powered by OpenGL hardware acceleration.

Where the AS3 3D libraries come into their own is the space between these two extremes: creative use of 3D, perhaps in combination with 2D elements, to enable an experience that, while not trying to compete with PlayStation, engages the user and provides new graphical possibilities.

**3D and HID**

As in much software development, the issues in system design coalesce not so much around what is *possible* but what is *desirable*. The kinds of human interface you can create using 3D are almost unlimited. However, you should be guided by exactly the same usability principles as apply to a 2D human interface.

In most cases, that means keeping the controls simple, intuitive, and familiar. There are of course edge cases. If you’re building a specialized application that demands sophisticated control from the user—software for building SketchUp models, for instance—you might need a fully immersive 3D world, and consider bundling it with specialized hardware such as the “WiiMote” controller for Nintendo’s Wii game console, which can be used as an input to a Flex app thanks to the open source WiiFlash project, and allows tracking of movements in 3D space. But in the context of a web site or a web application, it doesn’t make sense to disorient the user. 3D—like any other UI tool at your disposal—should be the means, not the end in itself.

This may seem like an obvious point, but the fact is that working in 3D is a lot of fun for the developer and has a certain “wow” factor that makes it very tempting to throw it in gratuitously. Resist that temptation! Use it where it really makes sense.