GDC 2012 Visual Effects Artist Roundtable Summary

This year was the first ever visual effects roundtable and it was a huge success. For the first time a significant percentage of the visual effects artists in the industry were in the same room! And not surprisingly, we all got along quite well. Here I'll summarize some of the topics, conclusions, and continuing impact the roundtable has had, as well as emphasize how excited we all are for another one next year. *Please GDC let us do it again!*

Before I get into the details it's worth noting that since the roundtable we've been able to coalesce as a discipline and have a growing presence on Tech-Artists.org (thanks to Rob Galanakis who set us up with a forum there), as well as a 90+ Facebook community. There clearly was a lot of pent up enthusiasm to find each other, ask questions, and stop reinventing the same wheels over and over again. In addition the market for visual effects artists is excellent - we went around the room and probably 7 or 8 people represented companies that were actively hiring.

But to the details: The first roundtable opened with an invitation to throw out topics that people would be interested in pursuing throughout the 3 days. We didn't have time to address them all, but it was still interesting to get a rough idea of what people wanted to discuss. Below are some of those topics, new ones that came up, and some comments and conclusions.

How can you effectively work with programmers, ask for new features and communicate effectively?

Know your gfx/fx programmers favorite beers and whiskeys! No, seriously. Also communicate to them clearly about what you need and why. Show them your workarounds. Prove that it's a good use of their time, and most importantly be nice! They're busy and want to help you if they can. Probably the most dramatic change you can make: Move your desk and sit next to them! This suggestion was met lots of enthusiasm.

Show them you're invested as well by trying to learn what they do. Even if it's a simple language like Processing, HLSL, or a scripting language. Try to muddle through it - it will help you get the language and understanding to know what's hard, what's easy, and how to ask for it.

If you're on xbox360, learn how to use PIX for profiling. It will show you how rendering actually happens and where in the process your effects are drawn, along with invaluable performance information.

FX Language

Wyeth at Epic brought up the topic of language, and how there is no common language for FX. This was reinforced over the 3 days as people's different pet names for various tricks came up.

- Some examples: *UV displacement* or *UV Distortion* or *Bump Offset*.
- *Alpha Crunch* or *Alpha Treshold* or simply "*that thing where you animate by changing a particles alpha cutoff*."
- What does it mean to "*Pop*." The analogy of music and animation was brought up where you have language like *Attack* and *Decay* along with different keyframes types *Linear*, *Smooth*, *Step*, etc..

What are you favorite tricks?

Macro UV’s: A trick where screenspace or world space uv’s are applied to the particles second uv set. That uv set can scroll a normal texture which in turn can perturb the lookup into your diffuse or alpha channels
creating great billowing smoke and liquid motion.

Alpha Crunch + Scrolling Diffuse + Distorting: The alpha texture is thresholded (crunched) at an animated value so it changes over time. In addition the alpha texture is perturbed by a scrolling normal map, so its silhouette changes. Finally 1 or 2 diffuse texture are scrolled across the particle to create unique patterns like smoke. (Confession: this actually came up at the FX drink up after the roundtable)

There was more! Maybe other people who attended will help fill in the gaps here.

Inspiration:

For artistic inspiration, It was agreed that Elemental Magic I and II is a must have for visual effects artists. For technical inspiration and the next generation of hardware, check out things like Siggraph, Nvidia and new engines running on high end PC’s.

Favorite Tool Features:

Must have fast iteration! Must have a live game connection, or some variation of hot reloading where your fx instantly update on target when they’re saved. It’s possible to author fx in a text file, but editors are not hard to create and help tremendously. If you’re strapped for programming time even something like a python tool that reads/writes XML can get you going.

Rob described a system at CCP where they expose their tools and rendering pipeline through python, so anyone can prototype tools and features. Then when it’s proven useful, it can be promoted to real game code, or just shipped as python.

Who is allowed to write shaders?

At some studios FX artists had full reign over their shaders (either through a node based editor or HLSL), and other studios had a closed system. There were a few in the middle who would allow artists to prototype shaders, and then pass them to engineers for final implementation.

For the studios that had a closed system, their motivation ranged from performance considerations to company culture and code ownership. But It was agreed that the future of visual effects going into the next generation will require FX artists to be involved in the authoring of shaders and materials.

This is especially important because a related topic was **the growing importance of mesh and shader based FX over the more traditional particle based ones**. More and more studios are doing crazy offline simulations in packages like Houdini, which yields tremendously impressive visuals. But those often need to translate back to animated meshes with heavily customized shaders. There are also many cases where mesh + crazy shaders can be much more performant than a pile of overdrawing particles.

Who is allowed to place effects?
Level designers? FX artists only? In most cases there were a variety of people placing FX, and a number of clever solutions came up for how to keep FX that weren’t placed by a FX Artist from accidentally

shipping without approval. One was to create a pre-approved set of FX, package them up and only let level designers place those. This way if they do ship, they're performant and don't look awful. Another was to create extremely obnoxious FX and only let people place those. That way it's immediately obvious when something needs to be fixed. A third clever solution was to have an entire directory for temp FX. Convert temp FX to final FX over the course of the project moving them to a new directory. But don't include the temp files in final builds of the game, then just see what's missing! (If you don't notice it missing, do you care?)

What have people's experience been using middleware?
A few people had integrated middleware solutions, but most were homegrown, in-house and proprietary. One middleware that everyone seemed excited about was PullDownIt for destruction.

Some topics that came up which we didn't have time to address:

- When to use traditional animation techniques vs. math based simulation techniques
- What are the biggest tools in your toolbox (Maya/Houdini/Realflow/etc...)
- How can you effectively schedule FX?
- How do people reuse FX within a project and between projects?
- How can create an FX capable engine from Open Source software?
- What's your workflow? How do you jump start an effect?

Maybe next year!