Los Angeles, CA

Paolo Surricchio

Graphics Programmer
www.paolosurricchio.com

425-502-0940
me@paolosurricchio.com

Skills

- Excellent at C/C++
- Excellent at DirectX 11
- Shipped titles 360, PS3, PS4 and PC
- Good knowledge of 3D math
- Excellent at shader programing/debugging
- Good knowledge of GCN architecture
- Confident with STL
- Confident with hardware debugging/profiling tools
- Great debugging skills
- Good knowledge of OpenGL 4+
- Confident with Microsoft VS
- Confident with sub-version tools

Experience and Projects

Respawn Entertainment: Apex (PC, Xbox and PS4) – Senior Rendering Programmer *August 2020 / Present Day*

- FX rendering

Santa Monica Studio – Senior Staff Rendering Programmer *May 2018 / July 2020*

- held the position of Point of Contact for the Rendering team toward content creators (artists, designers, production); I worked closely with Technical Director, reporting tasks and requirements for the game and designing solutions that fit the production schedule.

Santa Monica Studio: God of War (PS4) – Senior Rendering Programmer *December 2015 / May 2018*

- completely re-wrote material fx system (system responsible for spawning a reaction when something hits something else), porting the old system and data to the new one with a *"zero-day-down"* policy; designed an easy to use but powerful system taking into account sound, combat design, level design and fx artist requirements; worked with tech-art to write a PyQt UI for the framework to easily enter and modify data, and automatically translate the UI in our data definition language in text file, worked with physics to reuse ray cast data to allow for optimal reuse of collision data; throughout the project the specs kept on changing but the system evolved adapting to new requirements while keeping the same core philosophies and principles
- found and fixed bugs and optimized decal system (better stenciling options, many culling and rendering optimizations)
- inherited cloth system and fixed many multithreading bugs, optimized cloth jobs distribution
- took responsibility for a lot of new hire ramp up process in the middle of the production cycle
- took responsibility for creating an in-depth wiki about all particle features, inspiring others as an example of *"how ideally things should be done"*
- throughout the project I took upon myself to clean up a lot, A LOT, of technical debt, and received praises from colleagues more than once thanking me for having cleaned up something and how that made their tasks easier
- spent a lot of time working with content creators to create day-one to shipping solutions, concentrating on what we would need as a team, where we could cut corners, and how we could make sure the solutions would scale
- inherited and shipped an entire new gpu based particle system engine and worked closely with fx artists to create state of the art fx GPU pipeline; I took charge of the entire eco-system of fx, from high level combat scripting API, to low level GPU optimizations for both async compute shader particle simulation and rendering; fixed many fx bugs, amongst which many gpu syncing issues and particle simulation mistakes; worked on many fx features suchas weather fx (camera following wrapping volumes) for infinite snow/rain, MFX particle raycast for particles spawning other particle systems, screen space emitters (per pixel mesh emitters), and half res particles, as well as many small particle features for particle simulation, worked really closely with fx artists to make sure all features in the new system would respect the same logic and design philosophy, gpu buffer memory optimizations, wrote around 50 wiki pages explaining all new systems and requirements (particle lifetime, scripting and combat design API) and maintained along the project
- throughout the project I implemented countless engine and rendering optimizations

Amazon Game Studio: Lumberyard Engine – SDRE II *Sept 2015 / December 2015*

- fixed lots of bugs amongst which: texture streaming multithreading bugs, Qt editor bugs, and asset pipeline bugs

Siggraph 2015 presentation: User Centric Tools Programming in Firewatch *Los Angeles, August 2015*

- after the experience on Firewatch, I was contacted to speak about how I supported a team of small artists and empowered them to create the vast world of Firewatch
- user centric tools programming is a design approach to tools programming that shifts the focus on specific goal solving

and usability studies centered around a specific user trying to solve a specific problem

Campo Santo: Firewatch (PC, Mac, Linux, PS4) – Lead Rendering/Engine Programmer *May 2014 / Sep 2015*

- lead graphics/core engine programmer: on a team of ten I was completely in charge of everything that was graphics/core related: from tools to shaders, from the streaming system to level editing support
- changed the SECTR streaming library to have non spatial streaming and support recursive streaming levels
- starting from the Marmoset shader framework in Unity 4.3, ported, maintained, optimized and customized the framework and ported it to the Unity 5 deferred pipeline
- modified unity shaders to support translucency and full deferred terrain rendering with the deferred pipeline
- supported artists with a varied set of tools, from geometry painters to new materials, from procedurally generated content authoring tools to memory and streaming visualizer
- completely in charge of moving the team through the Unity 4.3 to Unity 5 transition (no team downtime)
- implemented the whole environment system where each independent component (fog, sun, sky, …) can be changed independently from the other through easy-to-use triggers places around the world
- ported and fixed most shader errors to PS4
- reverse engineered most issues by the way of graphic profiling builds of the game (the game is developed with Unity without source access)
- implemented many Post-FX that allowed both environment artist and concept artist to quickly iterate in the engine in real time: WYSIWYG approach to all graphic tools

High Moon Studios: Call of Duty: Advanced Warfare (Xbox 360, PS3, PC, Xbox One, PS4) *Feb 2013 / May 2014*

- floating point to BC5/DXN normal map compression for high quality smooth normals
- HDR to MDR reflection probe encoding for HDR lighting with low memory impact
- fixed complicated energy loss bug in the lighting pipeline for SH probe
- implemented tools and low level code for deferred light sprite in a forward pipeline
- implemented many post-FX shaders on current-gen hardware from next-gen specs
- implemented scene luminance detection for tonemapping on XBox 360
- fixed and improved custom scripting language allowing designers to keep more debug info in debug version but optimizing content and memory occupation in release version
- worked on many parts of the asset build pipeline for custom current-gen optimizations
- detected, fixed and optimized light clipping bug for dynamic lights
- detected shadow map optimization and implemented a tool to allow artist to selectively optimize shadow map per mesh
- optimized current-gen version of shaders and fixed many shader bugs
- changed the entire light-mapping engine pipeline from hard-coded to completely dynamic, from tools to game, to allow artists and designer to dynamically change lightmap resolution baking option for high quality bakes or really fast bakes
- worked closely to artists and designers and taught them to use profiling tools to optimize game content autonomously

High Moon Studios: Deadpool (Unreal Engine 3, Xbox 360, PS3 and PC) *May 2012 / Feb 2013*

- implemented dynamic wound rendering technology to display dismemberment on main character as well as enemies; scalable technology capable of efficiently drawing artist painted textures on characters at the exact position the character was damaged at. Integrated this system from gameplay API to low level rendering and material system
- implemented highly usable and optimized subsurface scattering skin material (Beckmann specular approximation, multiple normal map samples for subsurface scattering) used by the lead character artist on all characters
- implemented easy-to-use water material with refraction/reflection/diffraction and dynamic caustic projection
- fixed and optimized Unreal3 real time reflection technology and integrated the technology in the shader editor
- bug fixes and performance optimization on current gen hardware among which: gpu profiling, memory profiling, physics collision memory and performance optimization, found and fixed complicated Scaleform memory leak

---

Education

DigiPen Institute of Technology	Redmond, WA	GPA: 3.77
Master of Science in Computer Science,		May 2012
Universita' degli Studi Milano Bicocca	Milan, Italy	Final Grade: 101 out of 110
Bachelor Degree in Computer Science,		July 2010