Bruno Lourenço

www.bmlourenco.com bmlourenco@gmail.com

SUMMARY

- C++ programmer specializing in game engine development and computer graphics.
- Nine years of professional software development experience.

SKILLS & ABILITIES

- Proficient with C++, C and Java. Familiar with Python and C#.
- Familiar with Unreal, Unity and Godot game engines.
- Working knowledge of OpenGL, Direct3D 11 and Vulkan.
- Shader programming with GLSL and HLSL.
- Relevant mathematics and techniques for 3D graphics programming.
- Understanding of computer architecture in the context of software optimization.
- Relational databases and SQL. Network protocols and socket programming.

RELEVANT EXPERIENCE

Software Engineer – Sensei – 2021-current

- Developing a simulator of AI agents acting as customers on retail stores filled with cameras and a set of companion tools based on Unreal Engine (C++) to support the planning and deployment of AI-powered autonomous stores.
- Full rewrite and redesign of the simulator's core from a prototype state to allow capturing frames from virtual cameras to disk. The number of cameras of a typical store is on the order of hundreds, yielding thousands of frames per second to be stored onto disk.
- Made the capture cycle (render frame to a render target, transfer from VRAM to RAM, transfer from RAM to disk) asynchronous, with several frames in-flight, synchronized with fences between every stage. The transfer from RAM to disk stage was made multi-threaded with a job system. This achieved a 50x better frame throughput to disk, allowing the generation of a typical 15 min simulation within a 1 hour window..
- Redesigned the simulator to allow the separation of the agent's simulation from the rendering of frames and their storage, enabling different camera subsets of the same simulation to be rendered concurrently amongst multiple machines. This entailed saving the simulation state (the transforms of agent bones and movable objects) into a file, which is then played back by each of the rendering instances. This allowed scaling the simulator for large stores with thousands of cameras while still taking a reasonable time to generate all the frames.
    - Both the writing/reading of simulation frames to/from disk is made asynchronously on worker threads through a circular-buffer producers/consumers implementation.
    - By noticing the amount of repeated strings on the simulation file (mostly names and paths to assets) created a separate file to function as a string table, allowing replacing all the strings on the original file with an integer index that references the string on the table. This resulted on a 10% to 20% reduction on the simulation file size.
- Developed a 3D level-editor-like tool to create stores with walls, props, products, technical ceilings, cameras and network hardware. My main contributions:
    - The selection tool, allowing multi-selection and object cloning.
    - Visual effects for highlighting selected and hovered objects.

    - 3D transform gizmo for translating and rotating with free movement and axis/plane restrictions.
- Implemented accurate simulation of physical cameras lens distortion using the Brown-Conrady model. Devised a method to minimize aliasing caused by the distortion by finding the highest displacement factor caused by the lens and using that factor to modify the resolution of the original rendered image.
- Developed a system to evaluate the camera coverage of every point inside a store and evaluate potential camera placements. This is achieved by moving an agent through a grid on the store and checking the visibility of specific key points of the agent for every camera. The visibility check is done by projecting the point into the camera viewport, applying the lens distortion displacement and checking for occlusions through a ray cast. This process generates a heat-map-like texture that can be visually inspected on the virtual store to find bad coverage spots.

Bhazel – Personal project – 2019-2020 – More Details
- Work in progress C++ and Vulkan game engine.
- Physically-based renderer with image-based lighting and HDR.
- Cascaded shadow mapping and parallax occlusion mapping.
- Post-processing effects: bloom, FXAA and tone mapping.
- 2D batch renderer and Dear ImGui integration.

Godot Game Engine contributor – Open Source project – 2017-2020 – More Details
- Investigating and fixing user reported bugs and implementing small features.
- Contributing for the rendering, core, editor and Android export subsystems.

Ringgz – Released Android game – 2014 – More Details
- Fast-paced arcade game inspired on classic Brick Breakers with 10K+ downloads on Google Play.
- Developed with Java, libGDX, Box2D and OpenGL ES 2.0.
- Shader based gameplay-reactive background, particle systems and custom batch-rendering.

OTHER EXPERIENCE

Android Developer – Bliss Applications – 2016-2019
- Developing and maintaining native Android apps for various clients on agile multi-disciplinary teams.
- Examples include a running app for wearables, an e-commerce wine store and a companion app for Wi-Fi speakers.

Android Developer – Independent – 2013-2016
- Developed and published live-wallpapers, utility apps and games.
- Total of 4,5M app downloads and 65k daily active users.

EDUCATION

MSc in Information Systems and Computer Engineering – Instituto Superior Técnico – 2009-2013
- Master Thesis: E-Lumination - Lighting 3D Scenes Using Examples – More Details

BSc in Information Systems and Computer Engineering – Instituto Superior Técnico – 2006-2009

Mobile Application Penetration Tester Certification – eLearnSecurity – 2017 – More Details