

# Berk Emre Saribas

GAME DEVELOPER · GRAPHICS PROGRAMMER

Eindhoven, Netherlands

☎ +31681109164 | ✉ berksaribas@gmail.com | 📷 berksaribas | 🌐 berksaribas

I am a 3D graphics programmer with computer science background. I am interested and experienced in game development, game engine development and real time computer graphics.

## Education

### Technical University of Munich

Munich, Germany

M.SC. IN INFORMATIK: GAMES ENGINEERING

Apr. 2020 - May. 2022

- GPA: 1.4 German Grade (around 3.7 in US system)
- Tutored Parallel Programming course
- Master's thesis on real-time global illumination, elaborated below under projects section. Grade: 1.0 German Grade (4.0 in US system)

### Sabancı University

Istanbul, Turkey

B.S. IN COMPUTER SCIENCE AND ENGINEERING

Sep. 2015 - Jun. 2019

## Skills

**Technologies** C, C++, Vulkan, DirectX 12, OpenGL

**Specialty** Low/High Level Engine/Graphics Programming, Real-time Ray-tracing, Gameplay programming, Optimization and Debugging

**Game Engines** In-house engines, Unity, Unreal Engine

## Experience

### The Forge Interactive

Eindhoven, Netherlands (Remote)

GRAPHICS PROGRAMMER

Jul. 2023 - Today

C++ Vulkan D3D12 PS4/PS5 Xbox One/Series S/X Nintendo Switch HLSL

- Working on The Forge framework. My work includes optimizations, improving our ray-tracing API, bug fixes and experimentations on new rendering techniques.
- Worked on a VR title from a known company. Worked on PS5, D3D12, D3D11 and Vulkan backends of Unreal Engine to get parity between platforms and fix stereo rendering issues.

### GritWorld

Eindhoven, Netherlands (Remote)

3D GRAPHICS ENGINEER

Aug. 2022 - Jun. 2023

C++ Vulkan GLSL

- Worked on GritGene for Animation, an animation rendering engine that uses real-time graphics techniques from video games to create an interactable working environment for artists.
- Developed a two-pass order independent transparency technique with features such as sample redistribution, early stop for sorting and cache efficient memory layout for samples.
- Improved and extended the Diffuse GI technique that was based on Irradiance Probes by implementing dynamic probe relocation and tracing more rays to achieve temporal consistency for newly activated probes.
- Improved performance for hair strands rendering using mesh shading pipeline and vertex shading.
- Replaced normal, tangent, bitangent buffers that used 12 bytes each with 8 byte compressed quaternion representation.
- Implemented a ray-tracing debug visualization mode to visualize vertex buffers and some material properties to compare their corresponding parts in rasterization, using a split-view.
- Refactored the Vulkan renderer to use Vulkan Dynamic Rendering, replacing the Vulkan's render passes.
- Integrated FSR 2 for upscaling and as an alternative temporal anti-aliasing solution.

### Gram Games

Istanbul, Turkey

GAME DEVELOPER

Jul. 2018 - Apr. 2020

Unity C# Python

- Worked on Merge Magic from the first day of the project and shipped it. Worked on several in-house tools for Merge Magic and other games.
- Optimized Merge Magic's loading pipeline, reducing the loading times by 50%. Some of these improvements were later adapted in Merge Dragons.
- Implemented Seasons and Season Pass feature with another developer and worked on its distribution system.
- Reduced per frame garbage allocations.
- Rewrote the analytics buffer used for analytics events, which runs in parallel. Improved both game performance and CPU utilization. Halved down the overall battery consumption in all Gram titles.
- Created a Reference Finder that is capable of finding all the references of any game asset that has been referenced in Game Data (excel), Assets, or Code.
- Prototyped tens of different hypercasual games, working with designers and artists.
- Worked on an online tennis prototype with a game developer and a designer. Worked both on server and client side using Nakama framework. Implemented client-side prediction and server-side reconciliation.
- Worked on a bubble shooter spin-off of Merge Dragons as a gameplay programmer. This project ended up cancelled.

# Projects

---

## Master's Thesis - Extension of Precomputed Real-time Global Illumination by Specular Reflections (Oct. 2021 - May. 2022)

[HTTPS://GITHUB.COM/BERKSARIBAS/GLOBAL-ILLUMINATION-USING-SPARSE-RADIANCE-PROBES](https://github.com/BERKSARIBAS/GLOBAL-ILLUMINATION-USING-SPARSE-RADIANCE-PROBES)

C++ Vulkan

- Extending radiance probe based precomputed diffuse global illumination with hardware raytracing for glossy reflections.
- Implemented diffuse global illumination by precomputing specular harmonics functions for surface texels and near probes.
- Glossy reflections are achieved with two methods to compare results between them. The first method raytraces perfect mirror reflections and creating a blurred mip-chain for different roughness levels using a bilateral filter. Latter is stochastic raytracing that is temporally denoised using SVGF.
- The implementation is done using C++, using Vulkan as the base graphics API. Vulkan Memory Allocator to handle GPU memory allocations. GLM for math. Optick for profiling. Dear ImGui for GUI.

## Graphics Playground - D3D11 Compute Shader Ray Tracer (Hobby Project)

[HTTPS://GITHUB.COM/BERKSARIBAS/GRAPHICS-PLAYGROUND](https://github.com/BERKSARIBAS/GRAPHICS-PLAYGROUND)

C++ Direct3D 11

- Follows the book "Ray Tracing in One Weekend".
- Implemented with Direct3D 11 compute shaders.
- GUI with Dear ImGui.

## 3D Face Reconstruction (Jun. 2021 - Aug. 2021 at TUM)

[HTTPS://GITHUB.COM/BERKSARIBAS/FACE-RECONSTRUCTION](https://github.com/BERKSARIBAS/FACE-RECONSTRUCTION)

C++ OpenGL Ceres

- Implemented for the 3D Scanning and Motion Capture course.
- It detects landmarks from a 2D image and optimizes Basel Face Model to create a 3D of the original input face.
- Implemented using Ceres - Non-linear Optimization Library

## Physix (Nov. 2020 - Feb. 2021 at TUM)

[HTTPS://GITHUB.COM/YUPHIN/PHYZIX](https://github.com/YUPHIN/PHYZIX)

C++ Direct3D 11

- Implemented for the Game Physics course.
- Various physics simulations including Mass-spring simulation, Rigid-body simulation, Heat diffusion simulation, SPH fluids simulation and Stable fluids simulation.