Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

Implementation of rendering system in Rust

Eduard Lavuš

Supervisor: doc. Ing. Jiří Bittner, Ph.D.
Field of study: Open Informatics
Subfield: Computer Games and Graphics
May 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Lavuš** Jméno: **Eduard** Osobní číslo: **474497**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Studijní obor: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Implementace zobrazovacího systému v jazyce Rust

Název bakalářské práce anglicky:

Implementation of rendering system in Rust

Pokyny pro vypracování:

Proveďte rešerši metod používaných pro zobrazování virtuálních scén v současných herních enginech.
Vyberte důležitou podmnožinu zmapovaných metod a implementujte ji ve vlastním zobrazovacím systému založeném na jazyce Rust. Pro implementaci využijte 3D rozhraní Vulkan. V práci rozeberte výhody a nevýhody jazyka Rust ve srovnání s jazykem C++ pro implementaci dané úlohy.
Vytvořte demonstrační aplikaci, která ukáže možnosti vytvořeného systému na nejméně třech scénách různé složitosti. Součástí aplikace bude vyhodnocení rychlosti zobrazování (benchmark) a identifikace úzkých hrdel výpočtu.

Seznam doporučené literatury:

[1] Jason Gregory. Game Engine Architecture (3rd edition). CRC Press, 2018.
[2] Tomas Akenine-Moller et al. Real-Time Rendering (4th edition). CRC Press, 2018.
[3] Lagarde, S., and C. D. Rousiers. 'Moving frostbite to physically based rendering.' SIGGRAPH 2014 Conference, Vancouver. 2014.
[4] Daniel Šimek. Rozšiřitelný zobrazovací řetězec založený na odloženém stínování. Diplomá práce ČVUT FEL, 2013.
[5] Tomáš Dřínovský. Nepřímé osvětlení pomocí trasování kuželů. Diplomá práce ČVUT FEL, 2013.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Jiří Bittner, Ph.D., Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **11.02.2020** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **30.09.2021**

doc. Ing. Jiří Bittner, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

Acknowledgements

Thanky.

Declaration

I declare that this work is all my own work and I have cited all sources I have used in the bibliography.

Prague, May 22, 2020

Abstract

Abstract me away

Keywords: TODO key, TODO word

Supervisor: doc. Ing. Jiří Bittner, Ph.D.

Abstrakt

Abstrakt

Klíčová slova: TODO key, TODO word

Překlad názvu: Implementace zobrazovacího systému v jazyce Rust

Contents

Chapter 1

Introduction

Talk about Vulkan API and why it is great

Talk about what the project aims to achieve in short and long term, mention Rust

Chapter 2

Related work

Talk about V-EZ, Vulkano, gfx-hal, mention tephra

Chapter 3

Design

3.1 Rust

Talk about why Rust was chosen, include cpp vs Rust examples

3.2 Object lifetime management

Talk about how object lifetime is managed, maybe compare to tephra, talk about cargo features toggling

3.3 Synchronization

Talk about how some objects are internally synchronized
Talk about how GPU synchronization is left for future work

3.4 Validations

Talk about how only implicit validation are guaranteed, but some explicit validations are implicitly handled by api design and type system

3.5 Memory management

Talk about how device memory management is done through user-supplied memory allocator

3.6 Windows

Talk about how windows are handle, what is a surface and a swapchain and how they are supported

Chapter 4

Implementation

4.1 Cargo features

Talk about what cargo features are available and why they are beneficial

4.1.1 Vrc, Vutex etc.

Talk about Vrc and Vutex aliases

4.2 Generics

Talk about how Rust generics are used and optimized for comfort with minimal overhead

4.2.1 Deref

Talk about Deref trait usage

4.2.2 User code a.k.a. `dyn FnMut`

Talk about usage of `dyn`

Chapter 5
Benchmarks

Talk about benchmarks

5.1 Scene 1

5.2 Scene 2

5.3 Scene 3

Chapter 6

Conclusion

Conclude