
DTU

26/10/23

Real-Time Visual and Machine Learning Systems

Agenda

- Real-Time Systems
- Course Info
- Who am I
- Who are you?
- The Course Material
- Until Next Time
- Exercises
- Installing Rust
- Hello World
- Advent of Code

Real-Time Systems – Examples

Graphics

Computer Vision

Shadertoy

Performant Systems, like deep learning or edge computing

Real-Time Path Tracing?!

Real-Time Systems - Graphics

Games

3D Viewers – like PoTree

Tools

Real-Time Systems – Computer Vision

Tik Tok

SLAM systems

Real-Time Systems – Shadertoy

Write graphics shaders directly in your browser!

Art Coding with tutorial
Inigo Quilez's Snail
Bidirectional Path Tracing
Spectral Path Tracing
A 8k demo by my old boss

Real-Time Systems – Performant Systems

Path tracing

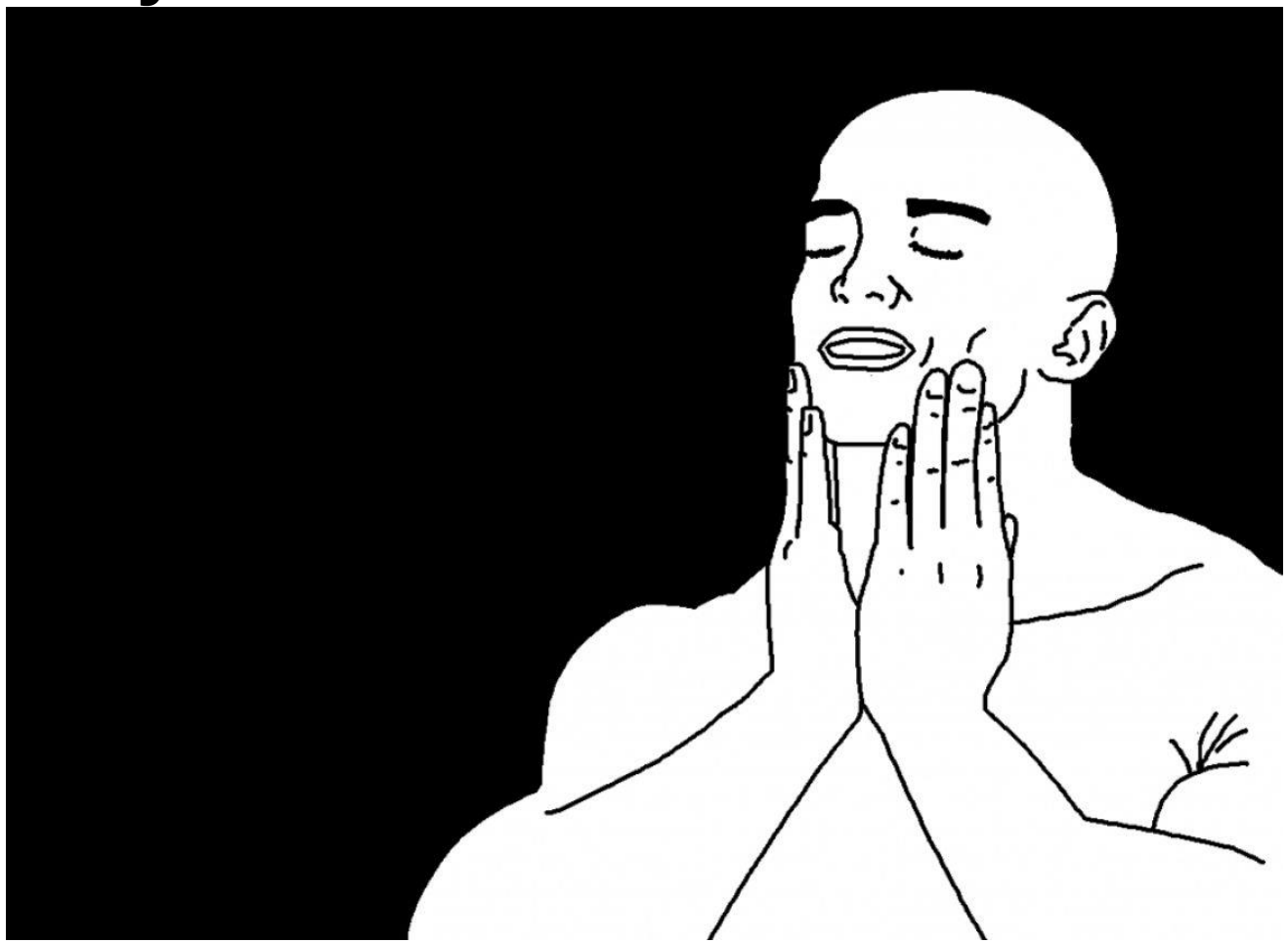
Training Neural Networks (You know this one)

Real-Time Systems – Real-time Path Tracing?!

Cyberpunk 2077

An explanation

Real-Time Systems – When it works



Link

Course Info

Websites

The Guide

The Course

Lectures and Exercises

Every Thursday afternoon

from 26/10 to 7/12

Project Work

Every day from 2/1-11/1

Handing In

Make a Github repository gathering all your 3 hand-ins and the final report

When mentioning experimentation and A/B testing, just tag the version, no need to keep mountains of code around

Invite me

Solo work only

Test Run

Please interrupt if you don’t understand something or something is unclear

Material is likely to be adjusted, changed and actually written as we go

If you have suggestions for the material, they are very welcome

Who am I

M.Sc. Digital Media Engineering, specialized in computer graphics

Worked in industry doing point cloud rendering and graphics engine work

Pioneer Centre Compute Guy

Who are you?

An introduction round!

Who are you?
How comfortable are you with programming?
What is your project about?

The Course Material

Multiple levels – levels 1-5 – An example

Specializations

The material goes into depth with memory, but loosens its grip with concurrency, types and profiling

It is impossible to cover all specializations in-depth, it is expected that you delve into the concepts that are relevant to you and your interests

Still under development

Why Rust?

The Good

- We need manual memory management
- C++ was the alternative, and that would have been harder
- The borrow checker is a tough, but fair, teacher
- You spend a minimal amount of time on importing and building libraries
- It made it very easy for me to give you 30 different mini-projects which you could just run

The Bad

- The ecosystem is not as mature as C++
- The borrow checker can be tough and you might feel stuck in the beginning

The Curious

- Compare Rust's compiler output to other languages using the online compiler Godbolt
- Rust vs. Python example, squares and squares squares
- Why you are getting so many assembler instructions

Until Next Time

Do Advent of Code until you start feeling comfortable

Read levels 1 and 2 for all modules
And most importantly level 1, 2 and 3 for module 1 (memory hierarchies)

Exercises

Do all of module 0

Install Rust
Read the material
Set up the debugger in VS Code
Advent of Code

Installing Rust

Follow along with the intro page

- Have you installed Rust?
- Have you installed VS Code?
- Did you add the extensions?
  - C/C++ (may/may not be needed on your platform)
  - CodeLLDB (for debugging)
  - Rust Syntax (Syntax highlighting)
  - rust-analyzer (LSP)
  - WGSL (Syntax highlighting)
  - wgsl-analyzer (LSP)

Hello World

- Create a new project using cargo new
- Run the hello world program – cargo run (--release) in the terminal
- Get the debugger to run

Advent of Code 2021

A humorous Christmas calendar with 2 tasks per day.

Sign up here: https://adventofcode.com/

Rest of the session – do Advent of Code for the year 2021 using Rust (I have solutions)