Portfolio

In this section I will showcase a selection of projects and parts of projects I deem especially interesting and relevant.

TestAndPlot

This is an extract from a larger project I was part off for the course *Advanced Systems Lab* as part of my Master's degree. It is a excellent showcase on how to utilise meta-programming and higher-kinded types to create a clean user-faced api.
It automatically creates a randomized test-set, measures run-time, flops and cycles and finally, plots them with gnuplot. Additionally, because of C interoperability it detects the return type of the provided function and deletes any pointer types.

File of interest: github.com/Heerdam/TestAndPlot/blob/master/include/test.hpp

Profiling tools generally do not provide any utility to count integer operations. Even Valgrind fails in that regard.
With the help of meta-programming a class, which complies with the C++ specification of an integer including implicit and explicit casting, can be injected in any properly templated function, including simd functions and thus, keep track of any operation done on itself.

File of interest: github.com/Heerdam/TestAndPlot/blob/master/include/ops_counter.hpp
File of interest: github.com/Heerdam/TestAndPlot/blob/master/include/simd_ops_counter.hpp

TurboGUI

This project implements the fastest possible OpenGl backend for ImGui by leveraging mapped flipflop memory buffers and manual synchronisation for asynchronous transfer.

File of interest: github.com/Heerdam/TurboGUI/blob/master/include/tb_gui.h

MdVis

MdVis is a high performance visualizer written in C++ and OpenGl. It visualises trajectory outputs of physical simulations. It is a prototype and has - admittedly - various design flaws. However, it is blazing fast leveraging the same methodology as TurboGui.

Project: github.com/Heerdam/MdVis

Turbofetch

This project is an assortment of CMake snippets leveraging the FetchContent API for a collection of my most used libraries. A library is simply added to a CMake project by copy-pasting the snippet.
Various snippets allows adding libraries that normally do not provide utility for the API or are not CMake projects at all.

Project: github.com/Heerdam/TurboFetch

Chlötzlispüeler

This is the project I am currently working on. It highlights my natural coding style and what I would deem proper C++.
In this project I am implementing a SPH (Smoothed Particle Hydrodynamics) solver based on the paper Divergence-Free SPH for Incompressible and Viscous Fluids. SPH is a numerical method to solve the Navier-Stokes-equation using convolution based particles as basis functions. Predominantly, this is used for ultra realistic fluid-simulations in scientific computing, engineering and movies. However, I see other interesting applications as well, which I intend to explore in this project.
The overarching design approach is to provide a modular, state-less solver by utilising templates and higher-kinded types. Different types of stencil, for example for the differential or kernels, might lead to different results in stability or performance.
This project is still very much in work-in-progress. The implementation itself is neither finished, nor are there any tests yet. The goal is to provide multi-threading and GPU functionality down the road and ultimately, use it in Unreal Engine 5.1.

File of interest: github.com/Heerdam/Chl-tzliSp-eler/blob/master/include/dfsph.hpp