Übungen zur Vorlesung "Architektur und Programmierung von Grafik- und Koprozessoren"
Übungsblatt 6

Sommersemester 2019

6 Parallelprogrammierung und Ray Tracing

Aufgabe 6.1

a.) Implementieren Sie den seriellen Algorithmus REDUZIERE() aus der Vorlesung mit C++11. Verwenden Sie als Vorlage die Datei `reduce.cpp`.

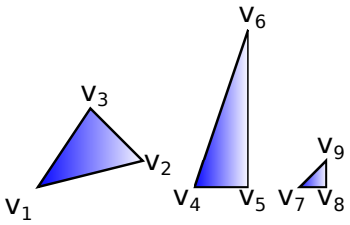
b.) Implementieren Sie den EREW Algorithmus REDUZIEREWORKTIME() aus der Vorlesung mit C++11. Dazu können Sie das *parallele for* Konstrukt aus der Datei `parallel_for.h` verwenden, oder alternativ ein paralleles for Konstrukt (und nur ein solches!) aus einer freien Bibliothek wie *OpenMP* oder Intels *Threading Building Blocks*. Sie dürfen annehmen, dass $n = 2^k$ Elemente reduziert werden.

Aufgabe 6.2

Implementieren Sie den parallelen Algorithmus zur Bestimmung der Präfixsumme mittels Pointer Jumping aus Aufgabenblatt 5. Verwenden Sie dazu die Vorlage (`prefix_sum.cpp`) und wie in Aufgabe 6.1 die beigefügte parallele for Schleife oder eine Alternative aus einer freien Bibliothek. Denken Sie daran, dass POINTERJUMPING ein *CREW* Algorithmus ist!

Aufgabe 6.3

Gegeben seien die nachfolgend eingezeichneten Dreiecke mit den zugehörigen Eckpunkten.



v_1	$(1, 1, 0)$
v_2	$(5, 2, 1)$
v_3	$(3, 4, 0)$
v_4	$(7, 1, 0)$
v_5	$(9, 1, 1)$
v_6	$(9, 7, 1)$
v_7	$(11, 1, 0)$
v_8	$(12, 1, 1)$
v_9	$(12, 2, 1)$

Die Dreiecke sind bereits in einen BVH Knoten einsortiert worden, für den nun entschieden werden soll, ob er weiter unterteilt wird. Bestimmen Sie eine optimale Split Ebene mit Hilfe der *Surface Area Heuristic* und der Sweeping Methode entlang der horizontalen Achse. Berechnen Sie dazu die Kosten für alle möglichen sinnvollen Splits und identifizieren Sie den Split mit den niedrigsten Kosten. Die Kosten, die beim Traversieren des Baumknotens entstehen, schätzen Sie genauso hoch ein wie die Kosten, die beim Schnitttest zwischen Strahl und Dreieck entstehen.

Das Übungsblatt wird am 23.05.2019 besprochen.