Assignment1 - Radiosity

Abgabe

Abgeschlossen

Beendet | 23.04.2023, 23:59
Abgabe am **23.04.2023, 23:09**

Die folgenden Lösungen wurden von Ihnen abgegeben:

Solutions_PA1_Perschke_Mittermair.zip
erstellt von Mittermair, Thomas am 23.04.2023, 23:01

Rückgabe und Feedback

Abgeschlossen

Ihr:e Betreuer:in hat die eingereichte Arbeit für die Bewertung akzeptiert.

Kommentar zu ihrer Abgabe 30.05.2023 von Ritter, Marcel

Code compiles and runs. add 1) * A triangle struct was added to handle the geometry. - It is more common to store three points A, B,C instead of one point a two edge vectors. (But its equivalent of course) * Triangles are refined (split) by an iterative subdivision scheme (1 -> 4), where one can define the number of iterations. - This is a very elegant solution. And simplifies the iteration. - The mentioned performance issue could be cured, by using the knownledge of the subdivision in the intersection and use a hierachrical approach. As triangles are created by a large, one could first interesect with the depth 0 triangles to fine a candidate. When a depth 0 tri was found, then iterate only over triangles that were created via the depth 0 tri. * The datatypes in the Split_Triangles look overly complicated and are error-prone. - std::vector *splitted_triangles = new vector; - raw pointers should be avoided - In older c++ (before 11/14) one would have preferred a pass-by-reference void Split_Triangles(vector& new_tris const vector& old_tris) - Since std 11/14 you can just create alocal vector as result and return it directly. It will be optimized by the compiler to a std::move operation. (which also could be written explicitly) vector Split_Triangles(const vector& old_tris) { vector splitted_triangles; splitted_triangles.reserve(4 * old_tris.length); // ... return splitted_triangles; // or explicitly return strd::move(splitted_triangles); } - When using push_back on a vector, better give a hint on the estimated size. (reserve) Otherwise, the re-sizeing can be costly. - Prefer const references when passing object over pointers. add 2) * An triangle - ray intersection was implemented using cross product to do the inside/ouside of a plane intersection. - Acutally, the handed out code is not so nice. I would also replace the pointer in the signature. Instead of _double* t_ it should be a pass by reference _double& t_ . etc. add 3) * Uniform random sampling was implemented for the triangle to triangle sampling of the monte carlo. - Sub-patches were disabled by setting patch size to one, which is ok, as you can just refine more more trienagles earlier. It is different though, as in the original code, the value stored per face was computer (averaged) over the sub patches. So you would need to store a depth - 1 triangle list for those values, to by equivalent. However, you can just use more random samples on the patches to achieve the similar effect. add 4) * A short report with images is provided. --> There is a small hickup sometimes in the brightness values, which might just come from the just five random samples.

Bewertung

Abgeschlossen

Leistungsübersicht

Bewertet

Anzahl Lösungsversuche	1
Punkte	min. 0 / max. 15
Erreichte Punktzahl	15

▾ Kommentar

see comments in the other section (it was too long > 2500 chars)
elegant solution. some minor coding issues (prefer reference over pointers, std::move())