FAKULTA APLIKOVANÝCH VĚD ZÁPADOČESKÉ UNIVERZITY V PLZNI

STUDENTSKÁ VĚDECKÁ KONFERENCE 2018

An Algorithm Recreating 3D Triangle Mesh Faces from Its Edges

Marek Zábran[1]

1 Introduction

3D triangle meshes are usually represented by a set of points with geometrical coordinates and a set of faces represented by triplets of these points. Multiple algorithms reconstructing a set of faces from a set of points and a set of edges were proposed in the literature, however, none of these can effectively reconstruct the set of faces using only the set of edges.

We propose such an algorithm, which recreates a set of non-oriented triangle faces from only the set of its edges. The input is expected to be a closed 3D edge-manifold triangle mesh of any genus. The algorithm is simple, purely topological and runs in O(n).

Several practical examples demonstrate that it is capable of reconstructing faces even from fairly large input data, as well as from input data that is prone to errors in reconstruction due to a high occurrence of possible inner faces.

2 Inner face

Should the faces of a triangle mesh not be clearly specified, for example should they be represented by only a sketch (wireframe), a problem arises. In such a case, it is possible to detect individual edges, but it is generally not clear which edges form a face.

This may sound like a trivial problem that is solvable by generating all possible triangles. Unfortunately, the number of these triangles is generally bigger than the number of actual faces, due to the possible presence of so-called *inner faces*.

An inner face (Fig.1) is a special false face which can be created, but which does not exist physically as it is not part of the surface and its presence makes mesh non-manifold.

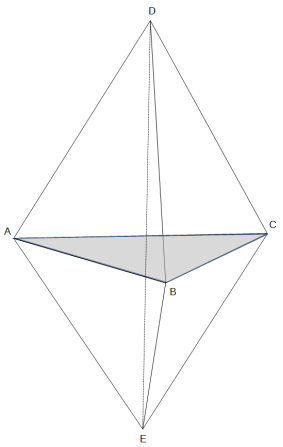

Figure 1: Two connected tetrahedra forming a hexahedron. Gray triangle is the inner face.

[1] student of bachelor study program Computer Science and Engineering, study field Computer Science, e-mail: zabran@students.zcu.cz

3 The proposed algorithm

Our algorithm was developed investigating similar algorithms as Varley&Company (2010). Since in our case the mesh is specifically triangular, the problem is simplified to inner face removal problem and the only mechanism these mesh recreating algorithms use to avoid inner face creation is the following well known role:

- **A triangle with at least one trihedral vertex cannot be an inner face.**

It is a very simple and effective rule, but not very robust, as a mesh can as well posses no trihedral vertex at all. But following this idea, similar rules can be found for other elements of the mesh like triangles or edges. These newly found rules has been proven much more reliable as they can be used repeatedly and so far seam to solve every closed 3D edge-manifold triangle mesh.

4 Results

Testing was performed on over 40 correct meshes in OBJ format, first obtaining a set of edges from a set of faces, then generating new set of faces from these edges using our algorithm and finally comparing this set of faces to the original set of faces.

All these generated meshes have been found topologically equal to the original, even in the case of very complex meshes with numerous inner faces, big genus or over million edges. Tests prove that the algorithm runs in linear time complexity with average time per face on common hardware being about 13 μm. The tests also suggest the algorithm may be significantly slower in case of certain mesh structures, but this is not caused by inner faces, as solving meshes maximizing the number of inner faces (Fig.2) is not affected significantly.

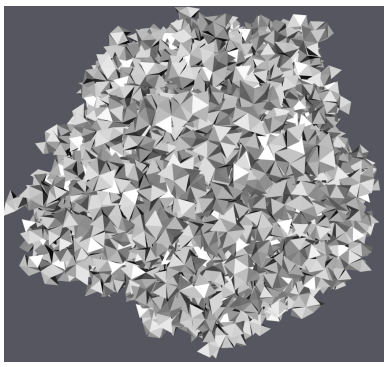

Figure 2: Mesh generated with intention to maximize the number of inner faces.

5 Conclusion

A new algorithm recreating closed 3D edge-manifold non-oriented triangle meshes have been proposed running in O(n). This algorithm most particularly solves the inner face removal problem and seams to be faster then any other similar existing algorithm.

References

Peter A.c. Varley and Pedro P. Company (2010) *A new algorithm for finding faces in wireframes*. Computer-Aided Design, 42(4). 279–309.