PolyCubes Optimization
Generating Coarse Quad-Layouts via Smart Polycube Quantization

Supervisor
Prof. Riccardo Scateni

M.Sc. Candidate
Gianmarco Cherchi
Goal

Original model

Polycube

Optimized Polycube

Quad-Layout
Quad-mesh

In 3D Computer Graphics a *Polygon-Mesh* is a collection of vertices, edges and faces of a polyhedral object.

Quad-meshes are a type of polygonal meshes in which faces are quadrilateral.

**Singularities**
Quad-Layout

The mesh domain subdivision called Quad-Layout is obtained by connecting the mesh singularities through chart boundaries.

Having a good quality quad-layout is very important for many applications.
Quad-Layout

Semi-structured Quad (re)meshing

High-order meshing

Texturing
PolyCubes are orthogonal polyhedra made up of:

- axis-aligned faces
- only 90° dihedral angles
- planar faces.

The most important property of a polycube is the ability to represent the original shape in a simple way.
The pipeline

The original model and its polycube

The quad-layout in the original polycube

The quad-layout in the optimized polycube

Gianmarco Cherchi

PolyCubes Optimization
A Possible Solution

The principle on which our approach is based is:

“to align polycube corners to remove the largest number of misalignments”

Mathematical Model
The Mathematical Model – Objective Function

\[
\min e = \alpha \cdot E_{\text{shape}} + \beta \cdot E_{\text{align}}
\]

\[
E_{\text{shape}} = \sum_{i \in V} \left[ (x_i - \bar{x}_i)^2 + (y_i - \bar{y}_i)^2 + (z_i - \bar{z}_i)^2 \right]
\]

\[
E_{\text{align}} = \sum_{(i,j) \in A_x} (x_i - x_j)^2 + \sum_{(i,j) \in A_y} (y_i - y_j)^2 + \sum_{(i,j) \in A_z} (z_i - z_j)^2
\]
The Mathematical Model – Objective Function

Gianmarco Cherchi
The Mathematical Model – Constraints

- Collinearity of the end-points
- Keep vertices in their halfspaces
- Minimum length of edges
- Integer coordinates
- Preserve already aligned vertices
- Avoid shape collapse
Dummy vertices and edges
The Algorithm

1. **Load Polycube**
2. **Compute Voronoi Diagram**
3. **Compute Adjacencies**
4. **Create Model & Optimize**
5. **Final Optimization & Quad-Mesh**
The Interactive Tool

Demo

Gianmarco Cherchi

Poly Cubes Optimization
Results - Bunny

Num. quads: 310 → 156  (49.68% of reduction) 0.24 sec
Results - Dragon

Num. quads: 396 $\rightarrow$ 210 (46.97 % of reduction) 0.49 sec
Results - Armadillo

Num. quads: 952 → 438  (54.00% of reduction)  2.36 sec
Results - Homer

Num. quads: 260 → 202 (22.31 % of reduction) 14.75 sec
Conclusions

- Our approach generates an optimized polycube that can be transformed into an optimized quad-layout.

Percentage of reduction of the quads’ number

Total time (in second)
Future Work

We would like to test our algorithm (with the appropriate changes) in the **hex-meshing** field.

Use optimized polycube for hex-mesh generation.

\[ Q \left( \begin{array}{c} \text{optimized polycube} \\ \text{original polycube} \end{array} \right) > Q \left( \begin{array}{c} \text{original polycube} \\ \text{optimized polycube} \end{array} \right) \]

\[ Q = \text{quality of the hex-layout} \]
Thanks!