Solidi Geodetici di I e II classe e loro duali

PolyhedronLibrary
contains output of
uses
1
uses
Gedim

Utils

void invertiValori(int& p, int& q)
vector<int> CalcoloVEFPoliedro(const int q, const int b, const int c)
vector<int> CalcoloDuplicato(const int q, const int b, const int c, const vector<int>& dimension)
vector<int> CalcoloDimensione2(const int b, const int q)
void RimuoviVerticiDuplicati(PolyhedralMesh& meshTriangulated)
void RimuoviLatiDuplicati(PolyhedralMesh& meshTriangulated)
void NewMesh(const PolyhedralMesh& meshTriangulated, PolyhedralMesh& meshFinal, const vector<int>& dimension)

void generaTetraedro(PolyhedralMesh& mesh)
void generaOttaedro(PolyhedralMesh& mesh)
void generaIcosaedro(PolyhedralMesh& mesh)

void Triangolazione(const int q, const int b, const int c, PolyhedralMesh& mesh, PolyhedralMesh& meshFinal)
void TriangolazioneDuale(const int q, const int b, const int c, PolyhedralMesh& mesh, PolyhedralMesh& meshFinal)
void Triangolazione2(const int q, const int b, PolyhedralMesh& mesh, PolyhedralMesh& meshFinal)
void PopulateCell3D(PolyhedralMesh& meshTriangulated)

void triangulateAndStore(PolyhedralMesh& mesh, PolyhedralMesh& meshTriangulated, const unsigned int b, const unsigned int c, const vector<int>& dimension)
void triangulateAndStore2(PolyhedralMesh& mesh, PolyhedralMesh& meshTriangulated, const vector<int>& dimension, map<pair<unsigned int, unsigned int>, vector<unsigned int>> edgeToFacesMap)

void FindAddEdge(const unsigned int a, const unsigned int b, PolyhedralMesh& meshTriangulated, unsigned int& edgeID, const unsigned int triangleID)
unsigned int FindAddVertice(const Vector3d& coord, PolyhedralMesh& meshTriangulated, unsigned int& k1)
unsigned int FindAddEdge2(const unsigned int a, const unsigned int b, PolyhedralMesh& meshTriangulated, unsigned int& k2)
Vector3d FindNearBarycenter(const PolyhedralMesh& meshTriangulated, const unsigned int edgeId, const unsigned int currentFaceId, map<pair<unsigned int, unsigned int>, vector<unsigned int>> edgeToFacesMap)
void FindAddFace(const vector<unsigned int>& new_face_vertices, const vector<unsigned int>& new_face_edges, PolyhedralMesh& meshTriangulated, unsigned int& k3)
vector<unsigned int> get_cyclic_normalized(const vector<unsigned int>& current_edges)
vector<unsigned int> NormalizeFaceEdges(const vector<unsigned int>& face_edges)

void CalculateDual(PolyhedralMesh& meshTriangulated, PolyhedralMesh& meshDual)
Vector3d getFaceBarycenter(const PolyhedralMesh& meshTriangulated, const unsigned int faceId)
map<pair<unsigned int, unsigned int>, vector<unsigned int>> buildEdgeToFacesMap(const PolyhedralMesh& meshTriangulated)
map<unsigned int, vector<unsigned int>> buildVertexToFacesMap(const PolyhedralMesh& meshTriangulated)
map<unsigned int, vector<unsigned int>> buildVertexToEdgesMap(const PolyhedralMesh& meshTriangulated)
void ProjectMeshToUnitSphere(PolyhedralMesh& mesh)

void WriteCell0Ds(const PolyhedralMesh& mesh)
void WriteCell1Ds(const PolyhedralMesh& mesh)
void WriteCell2Ds(const PolyhedralMesh& mesh)
void WriteCell3Ds(const PolyhedralMesh& mesh)

double calculateDistanceById(const PolyhedralMesh& mesh, const map<unsigned int, unsigned int>& vertexIdToIndexMap, const unsigned int id1, const unsigned int id2)
MatrixXi calculateAdjacencyMatrix(const PolyhedralMesh& mesh)
ShortestPathResult findShortestPathDijkstra(PolyhedralMesh& mesh, const MatrixXi& adjMatrix, const unsigned int startVertexId_real, const unsigned int endVertexId_real)

TESTUTILS

TestComputePolyhedronVEF
TestCalculateDuplicated
TestTriangulationTetrahedron
TestOrderedEdges
TestNotNullArea
TestNotNullEdges
DualTest

PolyhedralMesh

Cell0DsId : std::vector<unsigned int>
Cell0DsCoordinates : Eigen::MatrixXd
Cell0DsFlag : std::vector<std::vector<unsigned int>>
Cell0DsMarker : std::vector<unsigned int>

Cell1DsId : std::vector<unsigned int>
Cell1DsExtrema : Eigen::MatrixXi
Cell1DsFlag : std::vector<unsigned int>
Cell1DsMarker : std::vector<unsigned int>
Cell1DOriginalFlag : std::vector<bool>

Cell2DsId : std::vector<unsigned int>
Cell2DsVertices : std::vector<std::vector<unsigned int>>
Cell2DsEdges : std::vector<std::vector<unsigned int>>

Cell3DsId : unsigned int
NumCells0Ds : unsigned int
NumCells1Ds : unsigned int
NumCells2Ds : unsigned int
Cell3DsVertices : std::vector<unsigned int>
Cell3DsEdges : std::vector<unsigned int>
Cell3DsFaces : std::vector<unsigned int>

ShortestPathResult

numEdges : unsigned int
totalLength : double
verticesInPath : std::vector<bool>
edgesInPath : std::vector<bool>

ShortestPathResult(nEdges: unsigned int = 0, len: double = 0.0)

ExportParaview

void ExportParaview(const PolyhedralMesh& meshTriangulated)
void printMeshTriangulated(const PolyhedralMesh& meshTriangulated