Overview documentation, compositemodel

Vibeke Skytt

June 17, 2010

1 Brep topology

A CAD model can normally not be represented by one geometry entity like the ones in gotools-core. The need for sets of entities, adjacency analysis and representation of adjacency arises. There is a need for topological structures. Figure 1 shows the current topology classes in GoTools.

Body is a boundary represented solid model. It is a virtual volume limited by one or more shells, one outer and possibly a number of inner ones. The shells are represented by the class SurfaceModel which is also the entity used to represent a face set. In addition to being a topological entity, SurfaceModel provides an interface which views the set of surfaces as one entity. Thus, an operation like closest point computation can be performed without caring about which surface is closest to the point, see also section 2.3.

The surface model consists of a set of faces represented by the class ftSurface. The face represents the abstract idea of a bounded surface, but it also has a pointer to the geometric representation of this surface. The geometric surface is a ParamSurface, it may be a NURBS surface, an elementary surface or a trimmed version thereof. The face is bounded by one or more loops, one outer and possibly a number of inner ones. The inner loops represents trimming. In that case the geometric surface will be a BoundedSurface. All faces have an outer loop which either represents an outer trimming curve in the case of a BoundedSurface or the surface boundary.

A loop consists of a number of edges represented by ftEdge. The edges have a geometric representation by a ParamCurve. This is the level where adjacency information related to faces or surfaces is stored. An ftEdge is a half edge. It knows the face to which it belongs and the geometry of the curve it corresponds to. This curve is normally a CurveOnSurface curve. Then,

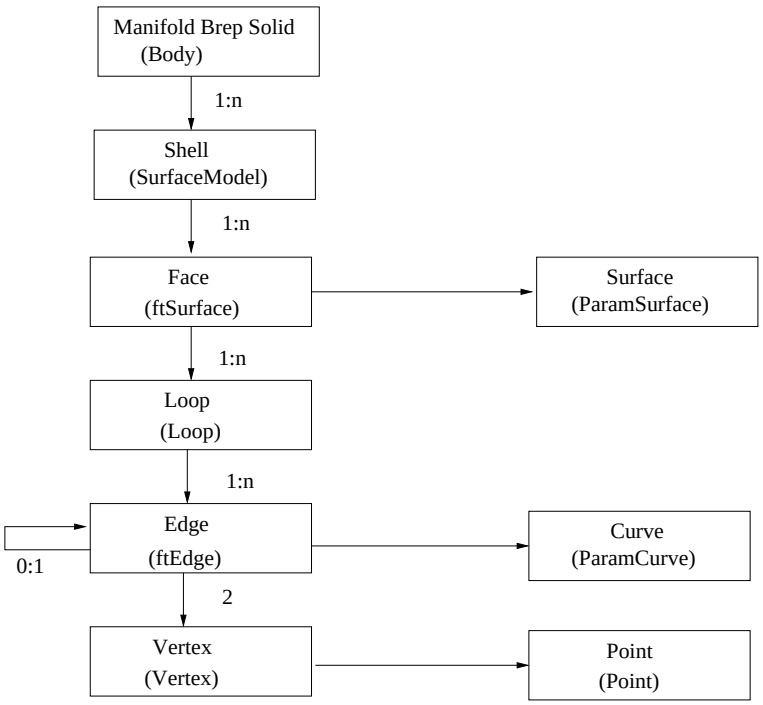

Figure 1: Topology structures for a boundary represented solid

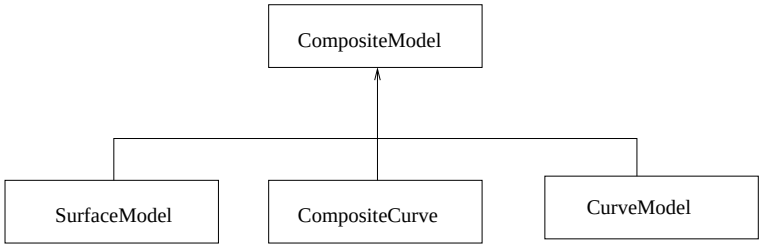

Figure 2: The composite model class hierarchy

both parameter and geometry information are available. The ftEdge has a pointer to a twin edge. This edge belongs to the adjacent ftSurface meeting the current one along the boundary represented by the current ftEdge. This construction is a very flexible one, but it does not automatically ensure C^0 continuity as the curves pointed at by the two twin edges, are normally not the same.

An edge is limited by two vertices which has a geometric representation as a point. A vertex has knowledge about all edges meeting in this point.

2 CompositeModel

SurfaceModel represents the shell in a boundary represented geometry model, but it is also the entity that is used to view a surface set as one unity. In this context, it inherits the abstract superclass CompositeModel that defines an interface for a unity of geometry entities of the same type. All composite models can

- Report how many entities it consist of
- Evaluate a given entity in a given parameter or parameter pair
- Compute the closest point from a given pont to the entity set
- Compute the extremal point of the entity set in a given direction
- Intersect itself with a plane
- Intersect itself with a line
- Compute the bounding box surrounding the entity set

- Check whether one specified entity is degenerate
- Turn parameter directions corresponding to one or all entities
- Tesselate itself
- Clone itself

Tesselation is performed in the gotools-core submodule tesselator. The methods here expect information about the tesselation resulution. For each parameter direction in the object, the number of tringles or line strips must be specified. The resolution is defined from the composite model. This can either be done setting a resolution for all entities or by a density parameter that governs the resolution. The composite model tesselation routines return a vector of shared pointers to a GeneralMesh. The type of meshes used for the concrete tesselation results are LineStrip, RegularMesh and GenericTriMesh.

The control polygon of the composite model or a selected subset of the model may be tesselated. In that case the output mesh is represented as a LineCloud. LineCloud is an entity in gotools-core/geometry.

2.1 CompositeCurve

A composite curve is an ordered collection of curves. The class expects a set of curves and will orient them order them in a sequence. The curve set should preferably be continuous, but also discontinuous set may be represented.

A composite curve will contain a number of ParamCurves, ordering information and continuity information. The curve sequence will be parameterized as a unity. Curve indices follow the indices for the curves given as input to the object. The class has the following type specific functionality:

- Fetch an individual curve
- Fetch the parameter interval of the composite curve
- Given a parameter value of one individual curve, get the corresponding parameter value in the composite curve, and vice versa. Given a parameter value of the composite curve, get the index and parameter value of the individual curve.
- Evaluate the composite curve as one unity

- Hit test. Compute the closest point in the composite curve to a given point along a given line
- Append a new curve to the composite curve

All the curves in the composite curve can be tesselated with respect to a default value, or a value given by the application, or a given subset of the composite curve can be tesselated with respect to the defined resolution. It is also possible to set the number of line segments for each curve relative to a given density parameter. Then the length of each line segment should roughly approximate this density. The density should be set according to the total size of the model. An upper bound of the tesselation size exists, but a large model combined with a small density will still lead to a heavy visualization model.

2.2 CurveModel

A curve model is an incomplete class inheriting a composite model. Not all the functionality defined in CompositeModel is implemented in this class. Its main purpose is to take a set of curves, for instance read from an IGES file, compute the topology of the curve set, use the topology information to order the curves into sequences, and return these sequences as composite curves.

Nevertheless, a curve model may be evaluated, tesselated and we can fetch a particular curve in the set by index or find the associated index given a curve. The index corresponds to the order of the input curves to this class. It is also possible to compute the bounding box surronding the curve set and the curve model may copy itself.

2.3 SurfaceModel

A surface model represents a surface set or a shell. It contains a number of parametric surfaces and topology information representing adjacency between surfaces in the set. The input surfaces to a surface model may be given as parametric surfaces or faces, i.e. ftSurface. In the latter case, the adjacency information may be given or not. A surface model may consist of several disjoint surface sets.

SurfaceModel has some specific functionality:

- Reset topology tolerances and recompute adjacency information, see section 3

- Return a specified surface or face
- Perform intersection with a line or a plane and represent the output in a surface specific manner
- Intersect the surface set with a plane, and trim the result with respect to the orientation of the plane
- Hit test. Compute the closest point in the surface set to a given point along a given line
- Append one or more new faces to the surface set
- Append another surface set to the current one
- Return information about the number of boundary loops in the surface set. One boundary loop will correspond to either a continuous subset of surface or a hole in the surface set.
- Return informations of gaps between surfaces in the set
- Return informations of kinks between surfaces in the set
- Return informations of sharp edges between surfaces in the set
- Triangulate the surface set with respect to a density parameter, and join the individual triangulations for each surface into one triangulation.
- Fetch information about neighbouring surfaces with inconsistent direction of the surface normals
- Fetch all vertices in the surface set
- Fetch vertices lying at the boundaries of the surface set

Some functionality is special for isogeometric models where each surface is expected to be a non-trimmed spline surface and the surfaces should lie in a corner-to-corner configuration

- Check if all surfaces are splines
- Check if the surface configuration is corner-to-corner

- Ensure a corner-to-corner configuration if this is not the case. The function applies only to spline surfaces
- Ensure that the spline surfaces in the model shares the spline space at common boundaries

All the surfaces in the surface model can be tesselated with respect to a default value or a pair of values given by the application, or a given subset of the surface model can be tesselated with respect to this resolution. The application can also specify the approximate number of quads in the tesselation. Each quad are divided into two triangles. Then a long and thin surface will get more triangles in the long direction. It is also possible to set the number of triangles in each parameter direction for each surface relative to a given density parameter. Then the length of each triangle should roughly approximate this density. The density should be set according to the total size of the model. An upper bound of the tesselation size exists, but a large model combined with a small density will still lead to a heavy visualization model.

3 Tolerances

The constructor of sub classes to a composite model requires a set of tolerances to be defined. With the exception of *approxtol*, these tolerances are used to compute adjacency between entities and to check the continuity between adjacent entities.

The tolerances are:

approxtol This tolerance is only required by SurfaceModel. The tolerance restricts the approximation error in functions where an approximation is performed. However, the current version of SurfaceModel does not perform approximation. Thus, the tolerance is superflous and not used. In principle, any value could be given. More appropriate would be to give values in the range $[1.0e^{-4}, 1.0e^{-1}]$ depending on the geometric size of the surface model.

gap CAD models are seen as continuous if they are continuous within this tolerance. If the distance between two points are less than this tolerance, they are viewed as identitical. The tolerance given in an IGES file corresponds to this tolerance. Note, however, that the data in the

file does not necessarily satisfy the gap tolerance and this can create problems for instance when reading trimmed surfaces. A reasonable tolerance is in the specter $[1.0e^{-6}, 1.0e^{-2}]$, but it depends on the expected quality of a model.

neighbour This tolerance is used in adjacency analysis and it represents the maximum distance between neighbouring surfaces or curves. If two curves or surfaces lie more distant than this tolerance, the entities are not found to be adjacent. The neighbour tolerance should be larger than the gap tolerance. If nothing specific is known, a factor of 10 makes sense, but if the gap tolerance is really small, a larger factor should be used. Surfaces that lie closer to each other than the neighbouring tolerance is found to be adjacent, but if the distance between the surface somewhere is larger than the gap tolerance, the surface set contains gaps. This is an error in the model.

bend If two surfaces meet along a common boundary and corresponding surface normals form an angle which is larger than this tolerance, it is assumed that there is an intential sharp edge between the surfaces. Similarly, if two curves in a composite curve meet with an angle larger than this tolerance, there is an intentional corner.

kink If two adjacent curves or surfaces meet with an angle less than this tolerance, they are seen as G^1 continuous. It the angle is larger than this tolerance, but less than the bend tolerance, the intended G^1 continuity is broken and it is an error in the model. The tolerance depends on the continuity requirements of the application. One suggestion is $1.0e-2$. The bend tolerance must be larger than the kink tolerance, for instance by a factor of 10. Both angular tolerances must be given in radians.

4 Intersection Results

Intersections performed in the composite model class either returns the intersection results in a format particular to the concrete composite model sub class or the intersection results are stored in a sub class to IntResultsModel. In the latter case, intersections can be performed without caring about the composite model type.

IntResultsModel stores the intersection results and the entities involved in the intersection and has the following functionality:

- Report upon the existence and number of intersection points and intersection curve segments
- It is able to tesselate itself

The actual geometry of the intersection results must be fetched from the sub classes IntResultsSfModel and IntResultsCompCv. The intersection results classes are very lean, but has the potential to get a rich set of functionality operating on intersection results.

SurfaceModel returns intersection results either as IntResultsSfModel or as ftCurve and a vector of ftPoint. It is also possible to fetch intersection results from IntResultSfModel as ftCurve and ftPoint.

A ftCurve is composed of a number of ftCurveSegment. A ftCurveSegment represenst a piece of an intersection curve. The segment is represented by a spatial curve and a curve in the parameter domain of the parametric entities involved in the intersection, that will be one or two. Thus, the curve segments distinguish between the individual surfaces in a surface set while ftCurve itself does not. ftCurve keeps track on the continuity between the segments it is composed of. A ftCurve is able to tesselate itself.

A ftPoint relates to one face, i.e. ftSurface, and has a geometric representation and a parameter domain representation. Each intersection point, for instance in an intersection between a surface model and a line, is represented as a ftPoint. The application has access to the position of this point, to the face with which it is associated and the parameter values in this face.

A composite curve represents its intersection results as an instance of IntResultsCompCv. Intersection points, which are the expected result in this case, are represented as PointOnCurve. Intersection curve segments are represented as pairs of PointOnCurve. The two points limit the extension of the intersection curve segment. The class PointOnCurve lies in gotools-core/geometry and holds a parametric curve, a parameter value and the geometric representation of the point. This content is accessible from the public user interface.

5 Composite Model Factory

Composite models of any type are created in the composite model factory. To open this factory, the tolerances described in section 3 are required. These tolerances are distributed to the composite models created in the factory.

The factory creates models from two sources. Either the data of a model is read from an IGES or g2 file or the model is created specifically from user input. The various possibilites are:

- Create one or more models from an IGES file. In the first case, either a composite model or a surface model is made depending on which entity dominates the file. In the second case, both composite curves and a surface madel may be generated.
- Create one or more models from a g2 file.
- Create a composite model from SISL curves or surfaces.
- Create a surface model representing a box.
- Create a sphere or a sphere segment. The output is a surface model consisting of one surface.
- Create a surface model representing a cylinder. The model consists of one surface.
- Create s surface model by interpolating a set of curves. The model consists of one surface.
- Create a composite curve with one segment from a circular arc, an elliptic arc or a line segment.

6 The Topological Face

The topological face (ftSurface) has access to the geometric representation of this face and information about neighbouring faces. In addition, it has knowledge about the solid (Body) it belongs to, if any.

The geometric surface is of type ParamSurface and described in the documentation of gotools-core.

The face is limited by a number of Loops. The geometric surface corresponding to a face corresponds to the trimmed face, so the information present in the boundary loops of the face, does also exist in the surface description.

ftSurface has a rich set of functionality of different types:

- Access functionality
  - Fetch the corresponding surface
  - Fetch boundary loops
  - Fetch edges belonging to the boundary loops
  - Fetch neighbouring faces
  - Fetch all vertices or subsets of vertices
  - Fetch the body owning this face
  - Fetch all bodies meeting in this face, maximum 2
  - Fetch the coincident face in a volume model
- Quality checking
  - Check for discontinuities in the surface
  - Check the distance between a face and corresponding edges and vertices
  - Check the orientation of loops belonging to the face
  - Check if the face has a narrow region
  - Check for acute angles in the boundary loops
- Functionality related to an isogeometric model
  - Check if the corresponding surface is a spline
  - Check if this face and the neighbouring face has corresponding spline spaces
  - Ensure that this face and the neighbouring face has corresponding spline spaces
  - Check if this face and the neighbouring face satisfies a corner-to-corner condition

  - Ensurat that this face and the neighbouring face satisfies a corner-to-corner condition
  - Fetch adjacency information between this face and a neighbouring face
  - Fetch information about boundaries where the face has no neighbouring face
  - Fetch information about coefficient enumerations for spline surface belonging to adjacent face and free boundaries

- Other queries
  - Evaluation
  - Closest point
  - Perform smoothing on the current face
  - Compute bounding box

7 The Topological Edge

The topological edge is implemented in the class ftEdge and it is a half edge. Thus, it contains information related to one face only. However, it has access to the corresponding half edge when the edge represents a boundary between two adjacent faces. The topological edge has a geometric representation as a ParamCurve. The edge may represent a restriction of the curve. This restriction is implemented using limiting parameters in the parameter interval of the curve.

The main functionality is:

- Fetch the curve representing the geometry description of the edge
- Fetch the parameters limiting the edge compared to the corresponding curve
- Fetch the face to which edge belongs
- Given an edge parameter, compute the corresponding face parameter
- Access to geometry information like bounding box, results of evaluation, closest point and an estimate of the curve length

- Information about the vertices limiting the edge, i.e. access to vertices and the edge parameter corresponding to a vertex
- Access to the radial edge if this entity exist. A radial edge is defined in volume model and the entity is described in the documentation in trivariatemodel.
- Get all surfaces meeting in this edge, a maximum of two