University of Bologna - Department of Mathematics

Piazza di Porta S.Donato, 5 - 40127 - Bologna

xcbool: the object composer User's Guide - Version 1.0

G. Casciola G. De Marco

Department of Mathematics
University of Bologna

Bologna 2000

Abstract

This report describes the *xcbool* system. This program performs boolean operations on sculptured solids. The boundary of a solid is represented using trimmed NURBS (Non Uniform Rational B-Splines) surfaces.

G. Casciola G. De Marco
Department of Mathematics, University of Bologna, P.zza di Porta S.Donato 5, Bologna, Italy. E-mail: casciola@dm.unibo.it.

Contents

Chapter 1

What is *xcbool* ?

xcbool is the object composer of the *xcmodel* system [XCMODEL00]. This package makes use of two separate programs, that are independently executable: *xcssi* to compute NURBS surface/surface intersection and *xcdbe* to convert a trimmed surface format (see Data file formats section). *xcbool* also makes use of the *trim* library for real time rendering of solid objects. These packages and the library are distributed booth in the executable version **xcboolusr.tar.gz** and in development version **xcbooldev.tar.gz** and are described in this document and in [TRIM99]. Download and installation instructions are in [XCMODEL00].

The field of solid modeling deals with the design and representation of physical objects. The two major representation schemes used in solid modeling are constructive solid geometry (CSG) and boundary representations (B-rep). Both these representations have different inherent strengths and weaknesses and for most applications both are required [Hof89]. While CSG implicitly represents a solid as an algebraic expression, B-rep explicitly defines an object as a set of surfaces.
Currently, most solid modelers are able to support solids composed of polyhedral models and quadric surfaces (like spheres, cylinders etc.) and their Boolean combinations. At the same time, the field of surface modeling has been developed to model classes of piecewise surfaces based on particular conditions of shape and smoothness. Such models are also referred to as sculptured models. Over the years, there has been considerable effort put into integrating surface and solid modeling. In particular, there is considerable interest in building complete solid representations from spline/NURBS surfaces and their Boolean combinations.
However, the major bottlenecks are in performing robust, efficient and accurate Boolean operations on the sculptured models. The main difficulty is in evaluating and representing the intersection of parametric surface patches and this has hindered the development of solid modelers incorporating parametric surface models.

As a result, most of the current solid modelers use polyhedral approximations to these surfaces and apply existing algorithms to design and manipulate these polyhedral objects. Not only does this approach lead to data proliferation, but the resulting algorithms are inefficient and inaccurate.
xcbool is a system designed to compose sculptured solids using boundary representation through Boolean operations. Starting from primitive solids, the system allows the user to obtain composed solids following the CSG tree of Boolean operations. A primitive object is defined by a single untrimmed NURBS surface. The final and partial objects in the CSG project are defined as a list of trimmed NURBS surfaces. The trimming curves for a surface are computed starting from all the intersection curves between this surface and the others; these are performed by *xcssi* package where a surface/surface intersection algorithm (SSI) specialized for NURBS surfaces has been implemented [CasMor95].

Chapter 2

How to work with *xcbool*

When *xcbool* is set, it opens the window shown in fig.2.1. Note that the GUI of *xcbool* is made with *xtools* [XTOOLS99]. The work area is subdivided into the following sections that we call:

- **Tools** top left;
- **New operation** under the previous, from which it is possible to set a Boolean operation;
- **Object list** under the previous, from which it is possible to select primitive or already composed objects as operands for a next Boolean operation;
- **Buttons** two rows of buttons at the bottom, which control a lot of operations that we will see in detail;
- **View** the large square area on the right, where the objects (their boundary surfaces) and/or the surface domain are visualized;
- **View Selector** bottom center, to select what to visualize;
- **Curve Selector** right of the previous, to select viewing of the trimming domain curves;
- **COP position** right of the previous, to select the camera position;

As soon as this window appears, you will immediately see which actions are possible. The buttons, cursors, textboxes, bars, etc. are accompanied by a label; conventionally, if the label is drawn in black, the related function is 'able', whereas white means 'unable'. Look now at the able buttons only:

2.1 Quit button

This button allows the user to quit *xcbool*.

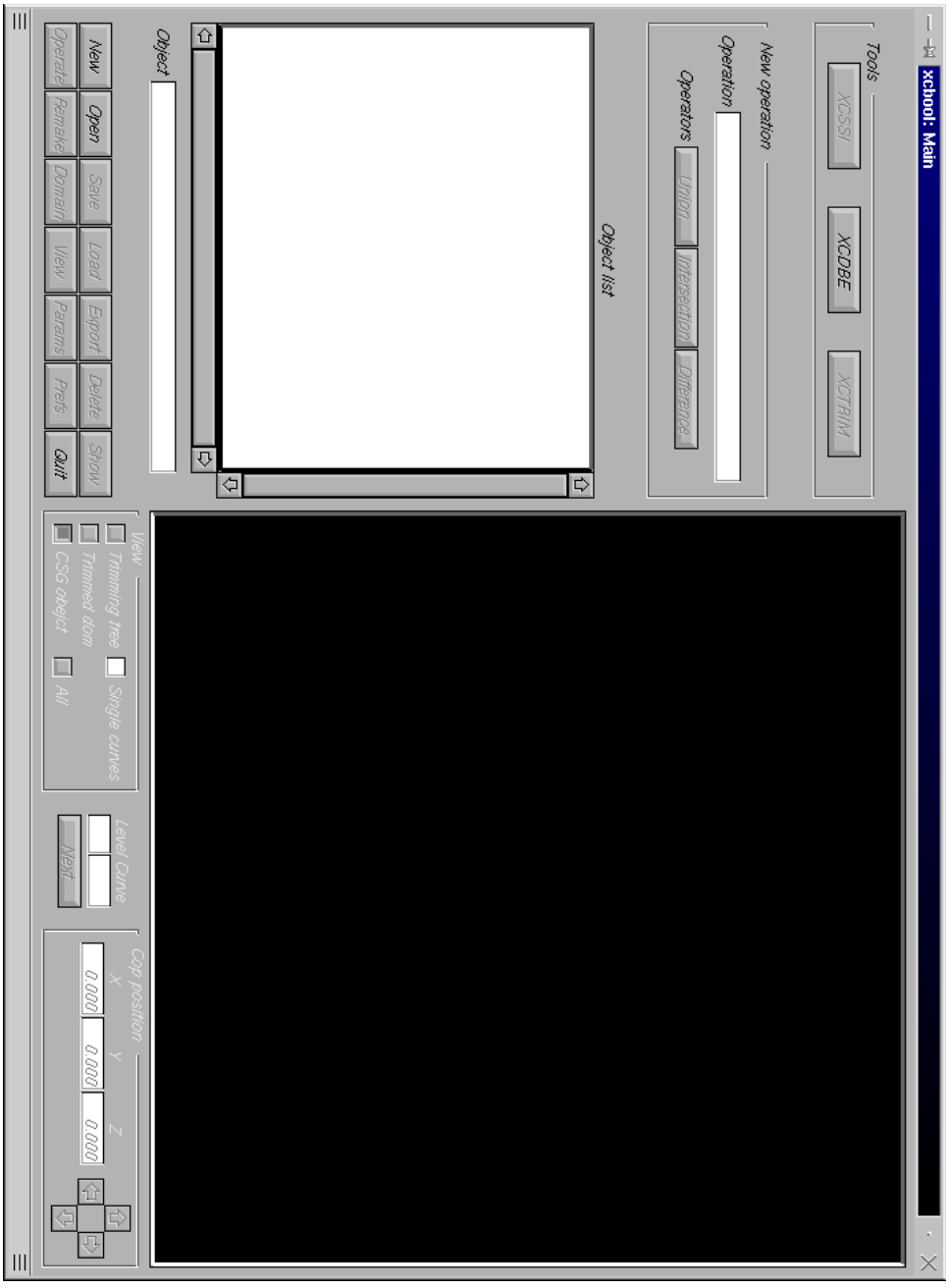

Figure 2.1: *xcbool* main window

2.2 New button

This button opens a file request to create a directory for the new project. Once a directory has been made, a dummy .csg file, with the same directory

name, will be created; this file will be used to contain our project objects. The **Load** button will now be abled (see Other buttons section) and we will be able to load primitive (.db files) and/or temporary (.dbt files) objects as operands for new Boolean operations.

2.3 Open button

This button opens a file request window to load a .csg file, which stores a project previously started and saved. Once the .csg file has been loaded, the usable objects to continue the project are shown in the object list. There are three possible object types:

- *primitive* represented by untrimmed and closed NURBS surfaces (.db file);
- *temporary* obtained from primitive and/or temporary objects, usable for compositions, but not as a final format and therefore not viewable (.dbt file);
- *composed* boundary representation of a final object obtaining from primitive and/or temporary objects (.obj file).

The boundary of a composed solid is represented by trimmed NURBS surfaces, that is, surfaces defined on special regions of the parametric domains that are, in their turn, defined by trimming curves. In the parametric domains of the NURBS surfaces, closed trimming curves are determined that could define either an *active* region (also known as an *island*) or a *non active* region (also known as a *lake*). An alternative organization of the "island" and "lake" curves includes the use of a particular 2D CSG tree known as a *trimming curve tree* . The able regions represent, on the parametric domain, the portions of the entire NURBS surface that form the boundary of the final solid, while the unable regions represent parts that must be excluded. For every composed solid, together with the file *.obj*, the trimmed surfaces representing its boundary are saved (*.tree* format) as well as the related *.dbt* file, in order to use this object as an operand for a following boolean operation.

Chapter 3

Boolean operations

To set up a Boolean composition between two objects, you have to select the first operand from the *object list*, then choose the operator (**Union**, **Intersection** or **Difference**) from the *Operation* box and then select the second operand. Only when an operation has been set up, the button **Operate** for the composition of the two solids will be abled, in such a way that we can compute the intersection of each surface with the others (**Make** or **Make All** buttons in Boolean Operation window). Once all the intersections have been computed, use the **Compute** button to obtain the composed solid.

3.1 Operate button

This opens a window to manage surface/surface intersections (see fig. 3.1), in which the pairs of surfaces to be intersected are presented. Each intersection is associated with a checkbox, indicating whether it has already been calculated or not, and the *Intersection parameters* box shows the suggested tolerance parameters (fields labeled *Default*) and those used (fields labeled *Used*). Before proceeding with the calculation of the intersection it is possible to set suitable tolerance parameters by modifying those in *Used*. The following shows what these parameters are used for:

- *Search Refinement Tolerance* (SRT): defines the quality of the adaptive subdivision of a surface controlling the initial level of linear approximation, and thus influencing the research of the *starting points* on the intersection curves. The SRT must therefore be such as to guarantee at least one starting point for each curve segment of intersection;
- *Curve Refinement Tolerance* (CRT): if two points belonging to an intersection curve are at a distance of less than CRT, the segment joining them is taken as an approximation of the part of the curve separating them; the CRT influences the number of points obtained for each segment curve;

For the first and the second surface, SRT tolerance is indicated with respect to *SRT1* and *SRT2*. At the present time there is no method for calculating a priori the exact combination of these parameters that can respond at the same time to the request of efficiency, robustness and accuracy, neither there is a way of estimating them in order to guarantee to find all the solutions.

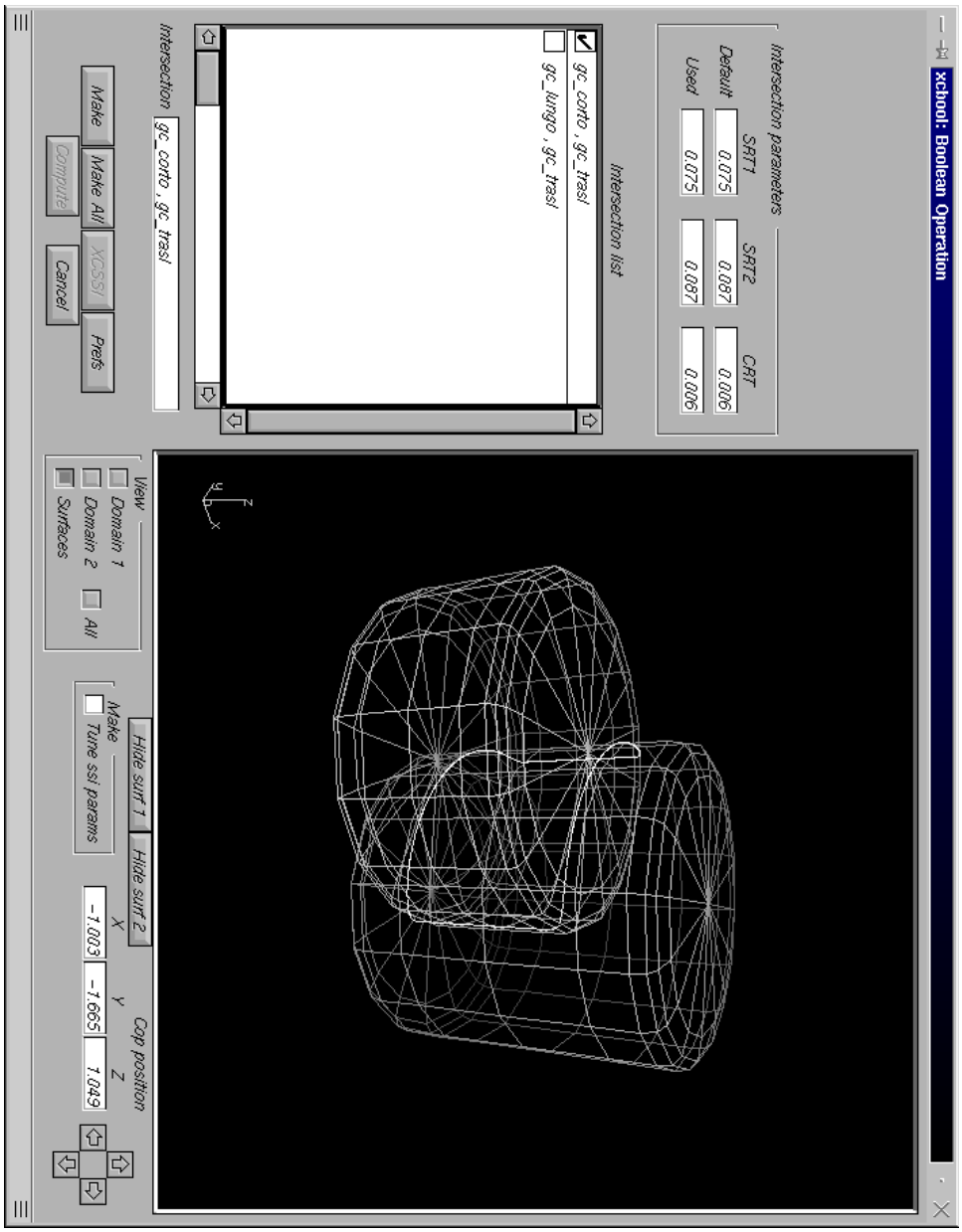

Figure 3.1: Boolean Operation window

3.2 Make button

This button runs the *xcssi* intersection program producing a file *.int* containing intersection curves for the couple of surfaces selected in the *Intersection list*. For this intersection the parameters specified in the textbox **Used** are used. To do this, *xcbool* relies on *xcssi*, that is loaded as a separate process. The checkbox **Tune ssi params** allows the use of *xcssi* in a semi-interactive way, in order to be able to follow the intersection phases step by step and to refine the initial parameters.

Once the intersection curves have been identified, they are superimposed on the drawing of the two surfaces, and visualized in the right section of the window. Here it is possible to observe the two surfaces with their related 3D intersection curves either together or separately, using the **Hide/Show surf** buttons. The checkpoints in the **View** box allow the user to choose the type of visualization:

- **Domain 1** shows the intersection curves in the parametric domain of the first surface;
- **Domain 2** shows the intersection curves in the parametric domain of the second surface;
- **Surfaces** shows the intersection curves in the 3D space together with the surfaces;
- **All** shows, at the same time, all the three previous visualizations;

3.3 Make All button

This button allows the user to create the remaining intersection files (.int files). In this case the intersection routine uses the default **Intersection parameters**.

3.4 Prefs button

This button allows the user to set some parameters regarding the visualization of the two surfaces to intersect (see. fig. 3.2). The checkpoints **Uniform** and **Adaptive** allow the user to choose a uniform or adaptive grid for the surface representation. In the uniform case the user must set the dimension in the U and V of the grid. In the adaptive case, an SRT planar approximation tolerance must be given. The same parameters are applied to both surfaces.

The **Apply** button allows the user to apply the parameters immediately so as to see a preview. The **Ok** button definitively applies the chosen parameter values and updates the view. The **Default** button resets the initial

parameter values. The **Cancel** button cancels the **Prefs** window, resetting the previous parameter values.

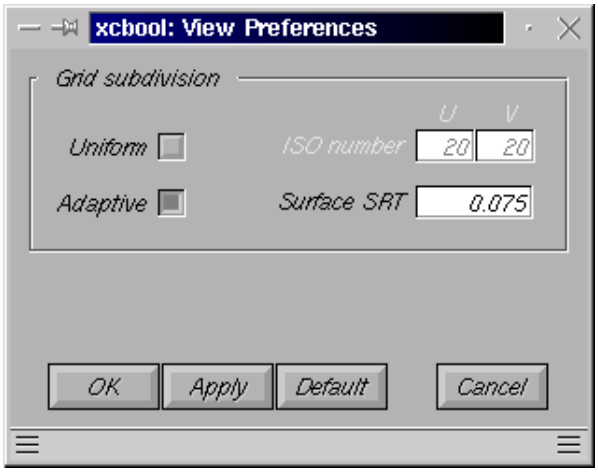

Figure 3.2: View Preferences window

3.5 XCSSI button

XCSSI is the Surface/Surface Intersection program. This button allows the user to execute this program in an interactive and educational way. (not yet available).

3.6 Cancel button

This button stops the Boolean operation procedure and closes the related window; the control returns to the **Main** window of *xcbool*.

3.7 Compute button

Once all the intersections have been calculated, the **Compute** button is abled, allowing for the actual composition of the two solids and closing the Operation window. What is obtained is a new composed solid that is inserted at the end of the object list. A symbolic name is given to this new object, which it is always possible to change, by modifying the name in the textbox **Object**, situated immediately below the *Object list*.

Chapter 4

Parametric domain visualization

After selecting a surface of a composed solid, using the **Domain** button in the main window (see fig.2.1), we can view the trimming tree associated with it. If abled, the checkbox **Single curves** allows you to view the curves one by one; in this case use the **Next** button to go on to the next. The textboxes **Level** and **Curve** show the level reached in the tree and the number of curves at that level. Once all the curves have been viewed, the areas in grey represent the active regions of the trimmed surface on the parametric domain.

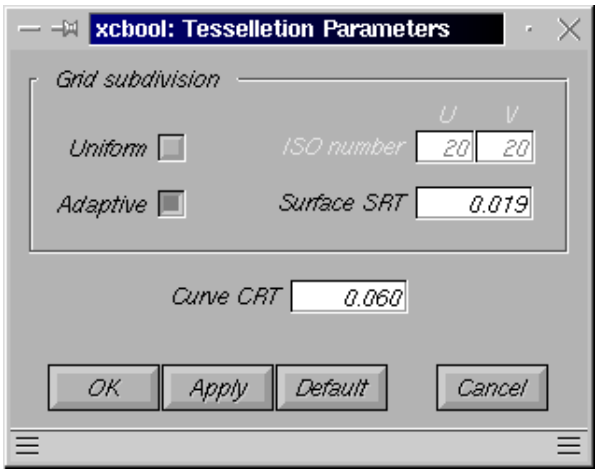

Figure 4.1: Tessellation Parameters window

By selecting **Trimmed dom** checkpoint in the Main window on the other hand, it is possible to view the tessellation operated on the trimmed NURBS domain and required for its representation. The parameters controlling this tessellation are set through the window of fig. 4.1, which is

opened by clicking the **Params** button. It is possible to choose the type of grid (uniform or adaptive) to be used for the surface domain. The related parameters determine the quality of the approximation of the surface. The *Curve CRT* tolerance defines the number of points on the trimming curve to be used for tessellation. Even the untrimmed surfaces, that is, those of the primitive solid or of temporary solids, can be viewed, and so their parameters can also be set for the approximation of the surface; in this case the *Curve CRT* tolerance is not considered.

The **Apply** button allows the user to apply the parameters immediately so as to see a preview. The **Ok** button definitively applies the chosen parameter values and updates the view. The **Default** button resets the initial parameter values. The **Cancel** button cancels the **Prefs** window, resetting the previous parameter values.

Chapter 5

Solid object rendering

Using the **Hide/Show** button in the Main window, it is possible to decide which object has to be visualized by the **View** button. It is also possible to able and unable the single surfaces making up an object. In the case that several objects are able, these are then visualized together, as if they were a single object, so that it is possible to observe their position and to predict the result of the composition in advance.

With the checkpoint **All** it is possible to obtain both a 3D view and a view of a parametric domain previously abled through the **Domain** button (see fig. 5.1).

5.1 Preferences window

The **Preferences** window (see. fig. 5.2) allows the user to choose the rendering method and to set some parameters regarding the object attributes for the shading method.

- *Revolution steps* defines the number of steps in which the camera completes a 180 degree rotation to view the surfaces;
- *Drawing mode* allows the user to choose the rendering method. *xcbool* provides *wire frame*, *depth cueing*, *hidden line* and *shading*;
- *Shading model* allows the user to choose between the *Gouraud* and *Phong* shading models; if the Phong model has been chosen, the user must set the *Exp* parameter for highlighting.
- *Light position* allows the user to set the light source position or direction in spherical coordinates; *Relative* means respect to the viewer;
- *Ambient ref.*, *Diffuse Ref.*, *Specular Ref.* parameters allow the user to define the object attributes; these are able for the Phong model only.

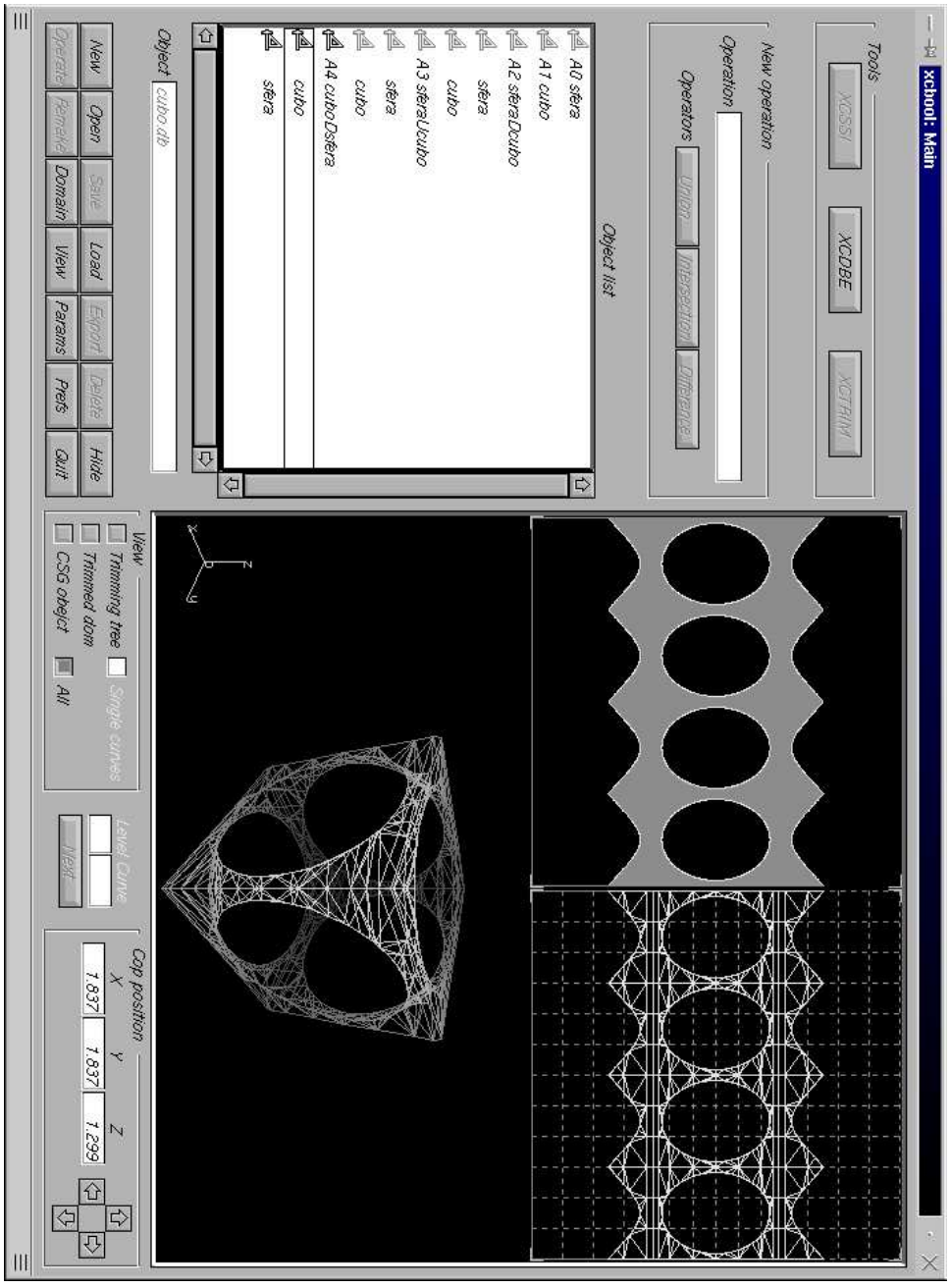

Figure 5.1: Composed object visualization

Fig. 5.3 shows a trimmed surface rendered using the *Phong* shading model.

The **Apply** button allows the user to apply the parameters immediately so as to see a preview. The **Ok** button definitively applies the chosen pa-

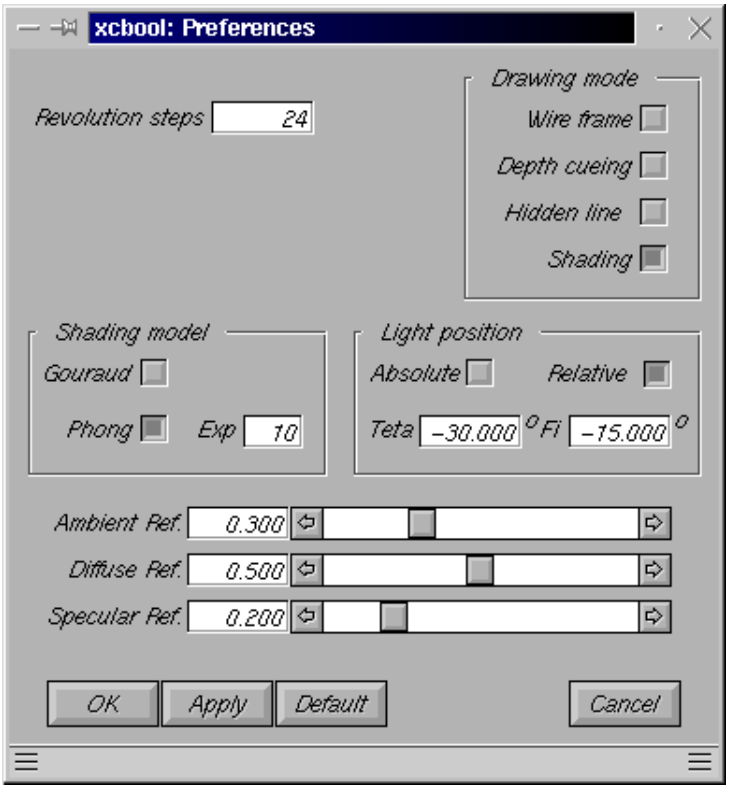

Figure 5.2: Preferences window

rameter values and updates the view. The **Default** button resets the initial parameter values. The **Cancel** button cancels the **Preferences** window, resetting the previous parameter values.

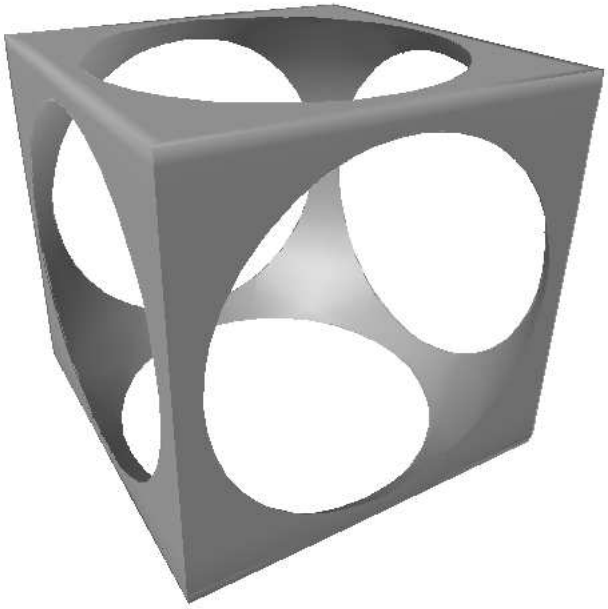

Figure 5.3: Shading Visualization

Chapter 6

Other buttons

6.1 Save button

This button allows the user to save the project by updating the .csg file.

6.2 Load button

This button allows the user to load primitive solids (untrimmed NURBS surfaces) or composed solids (previous results) as operands for a next Boolean operation.

6.3 Export button

In order to use a composed object for the construction of a scene model, we first need to save the trimmed NURBS surfaces representing the boundary in a specific format. The possible formats are *.tree* (*TREE format*) (see section Data file formats), that is used internally by *xcbool*, and the format *.dbe* (*DBE format*) (see section Data file formats) obtained by conversion of the internal format.

Through the window in fig. 6.1, which is opened by the **Export** button, you can specify the name and the directory in which to save the object and the format to be used, *.dbe* or *.tree*. In order to convert from the *TREE* format to the *DBE* format, the system uses the program *xcdbe,* that can also be run from the main window of *xcbool* through the button **XCBDE** (in this version this button is unabled).

6.4 Delete button

This option deletes the selected primitive or object from the Object list. It is not possible to delete a primitive that is still being used for another object.

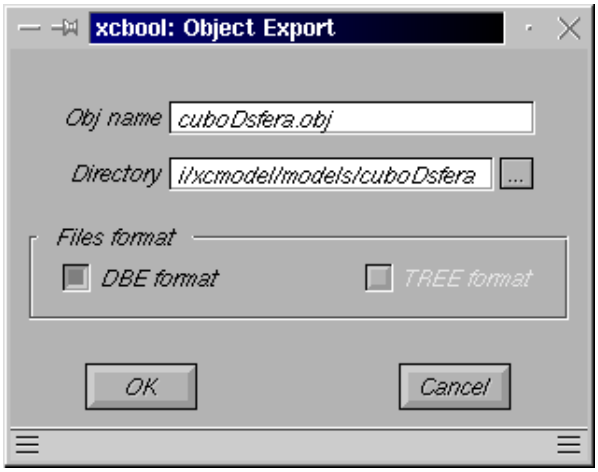

Figure 6.1: Object Export window

This is used in order to remake the object. Instead, it is always possible to delete a composed object, even if it used for another object, because it can be remade from its primitives using the information stored in the .csg file.

6.5 Remake button

The operations that have led to the construction of a composed solid can be repeated using the **Remake** button. This requires all the composition operations that have been used to construct the solid to be recalculated using the information contained in the CSG tree, saved in the *.csg* project file. Starting from the leaves of the CSG tree and going up to its root all the Boolean operations are carried out, one by one, recomputing the intersections relating to the operation with the original tolerance. Before proceeding with the computation of an operation, confirmation is requested, in order to allow the whole procedure to be interrupted.

Chapter 7

Data file formats

In this section the syntax of each file format used by *xcbool* is given and explained. The data files created or used by *xcbool* are stored in the directory `xcmodel/objects` as default. The # character in the following introduces a comment on the data in the file.

7.1 Intersection file

The following example file is:

`xcmodel/objects/cube_sphere/cube_sphere.int`

`cube_sphere` is the directory project, `cube_sphere.int` is the intersection file. The extension .int identifies these files. The format is self-explanatory.

```
Surface Names:
cube.db
sphere.db
Intersection Params SRT1=0.120000 SRT2=0.120000 CRT=0.010000
Number of Curves
6
Number Points for Curve 1
666
Break on Surface Domain 1
84
582
Break on Surface Domain 2
665
Domain Points of Curve 1
8.749998e-01 7.060278e-01 0.000000e+00 2.310833e-01
...
8.749998e-01 7.060278e-01 0.000000e+00 2.310833e-01
```

```
Number Points for Curve 2
666
Break on Surface Domain 1
665
Break on Surface Domain 2
84
582
Domain Points of Curve 2
1.000000e+00 2.078697e-01 1.250000e-01 7.689167e-01
...
1.000000e+00 2.078697e-01 1.250000e-01 7.689167e-01
...
```

7.2 Temporary result for a composed object

The following example file is:

`xcmodel/objects/cube_sphere/cubeDsphere.dbt`

`cube_sphere` is the directory project, `cubeDsphere.dbt` is the composed object file. The extension .dbt identifies these files.

```
L
3                                   # number of surfaces plus 1
0         # 0 means diff. boolean operator (1 inters., 2 union)
FILENAME:sphere.db          # follows the first operand; db file
DEGREE_U_V
         2         2
N.C.P._U_V
         5         9
KNOTS_U_V
         8        12
COORD.C.P.(X,Y,Z,W)
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00
 ...
 0.00000e+00 0.00000e+00-1.00000e+00 1.00000e+00
KNOTS_U
 0.00000e+00
 ...
 1.00000e+00
KNOTS_V
 0.00000e+00
 ...
 1.00000e+00
```

```
Curve:7                              # number of trimming curves
Holes:1          # 1 means that the trimming regions are lakes
FILENAME:out       # follows the first trimming curve as NURBS
DEGREE
1
N.C.P.
669
N.KNOTS
0
COORD.C.P.(X,Y,W)
0.000000e+00 7.689167e-01 1.000000e+00
...
0.000000e+00 7.689167e-01 1.000000e+00
KNOTS
FILENAME:out      # follows the second trimming curve as NURBS
...
FILENAME:cube.db         # follows the second operand; db file
DEGREE_U_V
         1         1
N.C.P._U_V
         4         5
KNOTS_U_V
         6         7
COORD.C.P.(X,Y,Z,W)
 0.00000e+00 0.00000e+00-7.50000e-01 1.00000e+00
 ...
 0.00000e+00 0.00000e+00 7.50000e-01 1.00000e+00
KNOTS_U
 0.00000e+00
 ...
 1.00000e+00
KNOTS_V
 0.00000e+00
 ...
 1.00000e+00
Curve:6                              # number of trimming curves
Holes:1          # 1 means that the trimming regionr are lakes
FILENAME:out       # follows the first trimming curve as NURBS
DEGREE
1
N.C.P.
669
N.KNOTS
0
```

```
COORD.C.P.(X,Y,W)
1.000000e+00 7.921304e-01 1.000000e+00
...
1.000000e+00 7.921304e-01 1.000000e+00
KNOTS
FILENAME:out      # follows the second trimming curve as NURBS
...
3                                  #3 means no boolean operation
```

7.3 Trimmed NURBS surface (.tree format)

The following example file is:

`xcmodel/objects/cube_sphere/cubeDsphere_sphere.tree`

`cube_sphere` is the directory project, `cubeDsphere_sphere.tree` is the trimmed NURBS sphere file.

```
FILENAME:sphere.db                 #follows the sphere .db file
DEGREE_U_V
         2         2
N.C.P._U_V
         5         9
KNOTS_U_V
         8        12
COORD.C.P.(X,Y,Z,W)
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00
 ...
 0.00000e+00 0.00000e+00-1.00000e+00 1.00000e+00
KNOTS_U
 0.00000e+00
 ...
 1.00000e+00
KNOTS_V
 0.00000e+00
 ...
 1.00000e+00
1                          #lake or island flag at the tree root
1                                      #number of root children
6                             #number of current curve children
558                        #number of points for the current curve
0.000000 0.768917           #follows the domain coordinate points
...
0.000000 0.768917
```

```
0                              #number of current curve children
554                         #number of points for the current curve
0.634467 0.496372             #follows the domain coordinate points
...
0.634467 0.496372
...
```

7.4 Trimmed NURBS surface (.dbe format)

The following example file is:

`xcmodel/objects/cube_sphere/cubeDsphere_sphere.dbe`

`cube_sphere` is the directory project, `cubeDsphere_sphere.dbe` is the trimmed NURBS sphere file.

```
FILENAME:cubeDsphere_sphere.dbe    #follows the sphere .db file
DEGREE_U_V
         2         2
N.C.P._U_V
         5         9
KNOTS_U_V
         8        12
COORD.C.P.(X,Y,Z,W)
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00
 ...
 0.00000e+00 0.00000e+00-1.00000e+00 1.00000e+00
KNOTS_U
 0.00000e+00
 ...
 1.00000e+00
KNOTS_V
 0.00000e+00
 ...
 1.00000e+00
1      #1 follows other information, 0 does not follow anything
0   #surface normal:0 uses computed normal, 1 inverse direction
0                          #lake or island flag at the tree root
8                                      #number of root children
0                              #number of current curve children
554                         #number of points for the current curve
0.634467 0.496372             #follows the domain coordinate points
...
0.634467 0.496372
```

```
0                              #number of current curve children
554                         #number of points for the current curve
0.384467 0.496372             #follows the domain coordinate points
...
0.384467 0.496372
...
```

7.5 B-rep Composed object

The following example file is:

xcmodel/objects/cube_sphere/cubeDsphere.obj

cube_sphere is the directory project, cubeDsphere.obj is the trimmed NURBS sphere file.

```
FILENAME:cubeDsphere.obj                          #object file name
2                            #number of trimmed surfaces involved
cubeDsphere.tree                #follow the trimmed surface names
cubeDsphere.tree
```

7.6 CSG representation for objects

The following example file is:

xcmodel/objects/cube_sphere/cube_sphere.csg

cube_sphere is the directory project, cube_sphere.csg is the history project file.

```
FILENAME:cube_sphere.csg                        #project file name
N.NODES: 6                                      #number of objects
NODE N. 0                                            #first object
TYPE: 0                                        #0 means primitive
LABEL: A0                                    #identificative name
NAME: cube.db                                            #file name
NODE N. 1                                          #second object
TYPE: 0
LABEL: A1
NAME: sphere.db
NODE N. 3                                           #third object
TYPE: 2          #final composed object; TYPE 1 means temporary
LABEL: A3
NAME: sphereDcube.obj                             #object file name
OPERATION:                           #boolean operation computed
```

```
3 1 0             #3 means Difference between NODE 1 and NODE 0
N.INTERSECTIONS: 1             #number of surface intersections
PARAMETERS:                  #SRT1, SRT2 and CRT parameters used
1.200000e-01 1.200000e-01 1.000000e-02
NODE N. 4                                          #fourth object
TYPE: 2                                  #final composed object
LABEL: A4
NAME: cubeDsphere.obj
OPERATION:
3 0 1             #3 means Difference between NODE 0 and NODE 1
N.INTERSECTIONS: 1
PARAMETERS:
1.200000e-01 1.200000e-01 1.000000e-02
NODE N. 5
TYPE: 2
LABEL: A5
NAME: sphereIcube.obj
OPERATION:
1 1 0           #1 means Intersection between NODE 1 and NODE 0
N.INTERSECTIONS: 1
PARAMETERS:
1.200000e-01 1.200000e-01 1.000000e-02
NODE N. 6
TYPE: 2
LABEL: A6
NAME: sphereUcube.obj
OPERATION:
2 1 0                  #2 means Union between NODE 1 and NODE 0
N.INTERSECTIONS: 1
PARAMETERS:
1.200000e-01 1.200000e-01 1.000000e-02
```

List of Figures

Bibliography

[CasMor95] G. Casciola, S. Morigi, Il problema SSI nella modellazione solida con superfici NURBS, Atti dell'Accademia delle Scienze dell'Istituto di Bologna, serie V, n.6, (1995).

[Hof89] C.M. Hoffmannn, Geometric and Solid Modeling. An Introduction, Morgan Kaufmann Publishers (1989).

[XCMODEL00] G. Casciola, *xcmodel*: a system to model and render NURBS curves and surfaces, User's Guide - Version 1.0, Progetto MURST: "Analisi Numerica: Metodi e Software Matematico", Ferrara (2000),
http://www.dm.unibo.it/~casciola/html/xcmodel.html

[XTOOLS99] S.Bonetti, G.Casciola, *xtools* library: Programming Guide - Version 1.0, (1999)
http://www.dm.unibo.it/~casciola/html/xcmodel.html

[TRIM99] G.Casciola, G. De Marco, *trim* library: Programming Guide - Version 1.0, (1999)
http://www.dm.unibo.it/~casciola/html/xcmodel.html