University of Bologna - Department of Mathematics

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

xcmodel: a system to model and render NURBS curves and surfaces User's Guide - Version 1.0

G. Casciola

Department of Mathematics
University of Bologna

Bologna 2000

Abstract

xcmodel is an interactive graphics system based on NURBS (Non Uniform Rational B-Splines) with the power of a professional CAD (Computer Aided Design) system, but realized and usable in an academic environment.

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

Contents

Contents i

1 Getting started 1
1.1 System requirements . . . 1
1.2 Download *xcmodel* . . . 2
1.2.1 With imake . . . 3
1.2.2 Without imake . . . 4
1.3 *xcmodel* directory structure . . . 4
1.3.1 *xcmodel* development version . . . 4
1.3.2 *xcmodel* binary version . . . 5
1.4 *xccurv* directory structure . . . 6
1.4.1 *xccurv* development version . . . 6
1.4.2 *xccurv* user version . . . 7
1.5 *xcsurf* directory structure . . . 7
1.5.1 *xcsurf* development version . . . 7
1.6 *xcbool* directory structure . . . 8
1.6.1 *xcbool* development version . . . 8
1.6.2 *xcbool* user version . . . 9
1.7 *xcrayt* directory structure . . . 9
1.7.1 *xcrayt* development version . . . 9
1.7.2 *xcrayt* user version . . . 10
1.8 *xcmodel* data . . . 11
1.9 *xcmodel* documentation . . . 11
1.10 Modifying the environment . . . 12

2 Tutorial: building a telephone 15
2.1 Curve and surface modelling . . . 15
2.2 Solid modelling . . . 17
2.3 Rendering . . . 18

3 Data file formats **21**
3.1 *xccurv* 21
3.2 *xcsurf* 21
3.3 *xcbool* 22
3.4 *xcrayt* and *xmovie* 22
3.5 Data file extensions 23

List of Figures **25**

Bibliography **27**

CHAPTER 1

Getting started

xcmodel is a system based on NURBS to model free form curves and surfaces, to compose solid objects and to render modelled scenes. The *xcmodel* environment integrates four interactive graphics systems:

- ***xccurv*** performs the modelling of 2D NURBS curves;
- ***xcsurf*** performs the modelling of 3D NURBS curves and surfaces;
- ***xcbool*** performs boolean operations on solids. The boundary of a solid is represented using trimmed NURBS surfaces;
- ***xcrayt*** with *xmovie*, performs the description and rendering of a modelled scene by a ray-tracing algorithm and its visualization.

This section of the *xcmodel* User's Guide, contains information specific to Unix implementation of *xcmodel*. It begins with instructions for installing *xcmodel* on your computer (the whole system) or one of its subsystems (*xccurv*, *xcsurf*, *xcbool* and *xcrayt*), along with a discussion of minimum system requirements. Later sections, provide information about what it is possible to do with the system and about the data file formats used.

1.1 System requirements

xcmodel is designed to run on any workstation or PC system running a Unix Operating System with Xwindow System.
xcmodel is able to compile and run in the following environments:

- SUN Sparc, Solaris, with SUN's C compiler "cc" or GNU's "gcc"
- SGI, Irix, with SGI's C compiler "cc"
- Intel, Linux

and we expect compilation problems to be minimized under other environments.

The development version of ***xcmodel*** (the whole system) requires 11 Mbytes of free space on your hard disk for PC-Intel (Red Hat 6.1).

xccurv and ***xcsurf*** do not require particular resolution or colors for the graphics adapter and display.

xcbool and ***xcrayt*** require at least a 24 bit and 1024x768 resolution graphics adapter and display.

Thus ***xcmodel*** (the whole system) requires at least a 24 bit and 1024x768 resolution graphics adapter and display.

1.2 Download *xcmodel*

xcmodel is freely distributed in compressed format files (tarred and gzipped) at the following Web site:

http://www.dm.unibo/~casciola/html/xcmodel.html

You can find *xcmodel* (the whole system), *xccurv*, *xcsurf*, *xcbool* and *xcrayt* (the single subsystems) in the development version (the source codes) or in the user version (the executables only); you can also find *xcmodel* data (curves, surfaces, objects and models as examples) and the documentation. The complete list is:

```
xcmodeldev.tar.gz          development version
xccurvdev.tar.gz
xcsurfdev.tar.gz
xcbooldev.tar.gz
xcraytdev.tar.gz
xcmodelusr.tar.gz          binary/user version
xccurvusr.tar.gz
xcsurfusr.tar.gz
xcboolusr.tar.gz
xcraytusr.tar.gz
curves2d.tar.gz            data
curves3d.tar.gz
surfaces.tar.gz
objects.tar.gz
models.tar.gz
doc.tar.gz                 documentation
```

The following installation procedure uncompresses and untars them in Unix environments in an intrusive manner. This means that a subdirectory `xcmodel` will be created in the directory you are using and all the files

will be included in this directory (to remove all, you simply give the unix command: rm -r xcmodel).

gunzip file.tar.gz
tar xvf file.tar

If you download a development version then you can either compile via "imake" or in a stand-alone way.

1.2.1 With imake

Imakefiles are provided to build specific Makefiles on your computer and operating system.
On Solaris, since the "cc" compiler is no longer part of the OS distribution, a lot of people use "gcc" instead. This is fine, but be aware that the imake tool you get as part of the Xwindow System on a Solaris box is configured for "cc". Therefore the compilation using the generated Makefiles will not succeed, unless you have changed the default configuration.
The compiling procedure can be performed in two different ways:

- By the Makefile contained in the `xcmodel/sources` directory.
  Execute the following command:

  make all

  to compile the whole system or

  make <target>

  to compile a single package, where target can be "xccurv", "xcsurf", "xcbool" or "xcrayt".

- Step by step.
  For each of the source directories, following the order shown in the specific sections, execute the commands:

  xmkmf -a

  or, if this option is not supported by your version of xmkmf:

  xmkmf
  make Makefiles
  make includes
  make depend

  Then simply execute:

make install

which will install the library or the program in `xcmodel/lib` or `xcmodel/bin` directory.

Building the *xcmodel* distribution with imake means that you have imake correctly installed and configured on your system. We have done our best to provide the Imakefiles so they work with as many systems as possible, but there is nothing we can do against wrong imake configurations. So, if your installation fails using imake, get your imake configuration fixed or don't use it.

1.2.2 Without imake

A set of makefiles is provided for those who do not have imake available on their system. However, this is only provided as a convenience, and should be considered as a starting point and not as something ready to use. These makefiles, called Makefile.ok, will most likely require some editing in order to be tweaked to your system.
Once this setting is done, you should be able to compile and install, by executing the following command for each of the source directories in the given order (see the specific sections):

make -f Makefile.ok

1.3 *xcmodel* directory structure

After installation, your `xcmodel` directory will include the following subdirectories containing the various components of the *xcmodel* system.

```
/bin            the xcmodel binary and associated files
/lib                          the xcmodel libraries
/doc                        the xcmodel documentation
/sources                     the xcmodel source codes
/curves2d
/curves3d
/surfaces
/objects
/models
/textures
```

1.3.1 *xcmodel* development version

The development version contains an almost empty `bin` and `lib` directories. The `bin` directory contains:

```
.xccurvrc
.xcmodelrc
.xcsurfrc
seal.ppm
```

The `lib` directory contains:

```
camera.xclib
refractionindex.xclib
/textures
```

If you list the directory `sources` you find:

```
/xtools
/matrix
/descriptor
/xcmodel
/xccurv
/xcsurf
/trim
/xcssi
/xcdbe
/xcbool
/hrayt
/xcrayt
```

Note that this is the order to follow for the compilation procedure.

1.3.2 *xcmodel* binary version

The `bin` directory contains the executable files. If you list this directory you find:

```
.xccurvrc
.xcmodelrc
.xcsurfrc
hrayt
seal.ppm
xcbool
xccurv
xcdbe
xcmodel
xcrayt
xcssi
xcsurf
```

```
xctrim
xframe
xhrayt
xmovie
```

If you list the `lib` directory you find:

```
camera.xclib
refractionindex.xclib
/textures
```

The binary version contains an almost empty `sources` directory. In fact, you only find:

```
/matrix
/descriptor
/include/descriptor/descriptor.h
/include/descriptor/types.h
/include/matrix/matrix.h
/include/hrayt/vec\_macros.h
/include/hrayt/types.h
/include/globals/define.h
```

In order to create new models you need to apply the compilation procedure for the following directories:

```
/matrix
/descriptor
```

1.4 *xccurv* directory structure

After installation, your `xcmodel` directory will include the following subdirectories containing the various components of the *xccurv* system.

```
/bin              the xccurv binary and associated files
/lib
/doc                              the xccurv documentation
/sources                            the xccurv source code
/curves2d
```

1.4.1 *xccurv* development version

The development version contains an almost empty `bin` directory. It contains:

```
.xccurvrc
.xcmodelrc
seal.ppm
```

If you list the `sources` directory you find:

```
/xtools
/xcmodel
/xccurv
```

Note that this is the order to follow for the compilation procedure.

1.4.2 *xccurv* user version

The `bin` directory contains the executable files. If you list this directory you find:

```
.xccurvrc
.xcmodelrc
seal.ppm
xccurv
xcmodel
```

The binary version contains empty `sources` and `lib` directories.

1.5 *xcsurf* directory structure

After installation, your `xcmodel` directory will include the following subdirectories containing the various components of the *xcsurf* system.

```
/bin                the xcsurf binary and associated files
/lib
/doc                              the xcsurf documentation
/sources                            the xcsurf source code
/curves2d
/curves3d
/surfaces
```

1.5.1 *xcsurf* development version

The development version contains an almost empty `bin` directory. It contains:

```
.xcsurfrc
.xcmodelrc
seal.ppm
```

If you list the `sources` directory you find:

```
/xtools
/xcmodel
/xcsurf
```

Note that this is the order to follow for the compilation procedure.

xcsurf user version

The `bin` directory contains the executable library files. If you list this directory you find:

```
.xcmodelrc
.xcsurfrc
seal.ppm
xcmodel
xcsurf
```

The binary version contains empty `sources` and `lib` directories.

1.6 *xcbool* directory structure

After installation, your `xcmodel` directory will include the following subdirectories containing the various components of the *xcbool* system.

```
/bin              the xcbool binary and associated files
/lib
/doc                            the xcbool documentation
/sources                          the xcbool source code
/objects
```

1.6.1 *xcbool* development version

The development version contains an almost empty `bin` directory. It contains:

```
.xcmodelrc
seal.ppm
```

If you list the `sources` directory you find:

```
/xtools
/xcmodel
/trim
/xcssi
/xcdbe
/xcbool
```

Note that this is the order to follow for the compilation procedure.

1.6.2 *xcbool* user version

The `bin` directory contains the executable files. If you list this directory you find:

```
.xcmodelrc
seal.ppm
xcmodel
xcbool
xcssi
xcdbe
xctrim
```

The binary version contains empty `sources` and `lib` directories.

1.7 *xcrayt* directory structure

After installation, your `xcmodel` directory will include the following subdirectories containing the various components of the *xcrayt* system.

```
/bin              the xcrayt binary and associated files
/lib
/textures
/doc                            the xcrayt documentation
/sources                          the xcrayt source code
/models
```

1.7.1 *xcrayt* development version

The development version contains an almost empty `bin` and `lib` directories. The `bin` directory contains:

```
.xcmodelrc
seal.ppm
```

The `lib` directory contains:

```
camera.xclib
refractionindex.xclib
/textures
```

If you list the `sources` directory you find:

```
/xtools
/matrix
/descriptor
/xcmodel
/hrayt
/xcrayt
```

Note that this is the order to follow for the compilation procedure.

1.7.2 *xcrayt* user version

The `bin` directory contains the executable files. If you list this directory you find:

```
.xcmodelrc
seal.ppm
xcmodel
xcrayt
hrayt
xframe
xhrayt
xmovie
```

If you list the `lib` directory you find:

```
camera.xclib
refractionindex.xclib
/textures
```

The binary version contains an almost empty `sources` directory. In fact, you only find:

```
/matrix
/descriptor
/include/descriptor/descriptor.h
/include/descriptor/types.h
/include/matrix/matrix.h
/include/hrayt/vec_macros.h
/include/hrayt/types.h
/include/globals/define.h
```

In order to create new models you need to apply the compilation procedure for the following directories:

```
/matrix
/descriptor
```

1.8 *xcmodel* data

xcmodel system and its subsystems use and exchange data files organized into six directories:

```
/curves2d
/curves3d
/surfaces
/objects
/models
/textures
```

In the tarred and gzipped files you can find, rispectively, a lot of already modelled curves, surfaces, objects, models and textures as examples.
If you download a subsystem you find the relative data directory enclosed with one example only. For completeness we summarize the subsystems and their data directories, referring to the "Data file formats" section to explain who creates a data file and who only uses it:

xccurv: curves2d
xcsurf: curves3d and surfaces
xcbool: objects
xcrayt: models and textures

1.9 *xcmodel* documentation

xcmodel system and its subsystems are all documented. The documentation is in PostScript format. You can find this documentation in the tarred and gzipped file `doc.tar.gz`. It consists of:

```
/doc/xcmodel.ps       this document!
/doc/xccurv.ps
/doc/xcsurf.ps
/doc/xcbool.ps
/doc/xcrayt.ps
/doc/xtools.ps
/doc/trim.ps
/doc/descriptor.ps
/doc/matrix.ps
```

1.10 Modifying the environment

xcmodel system (as all of its subsystems) runs giving the Unix command:

./xcmodel &

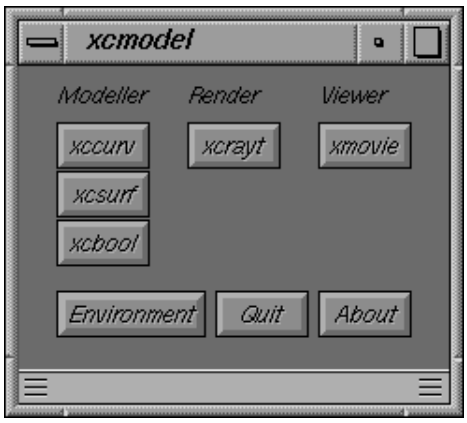


Figure 1.1: *xcmodel* console window

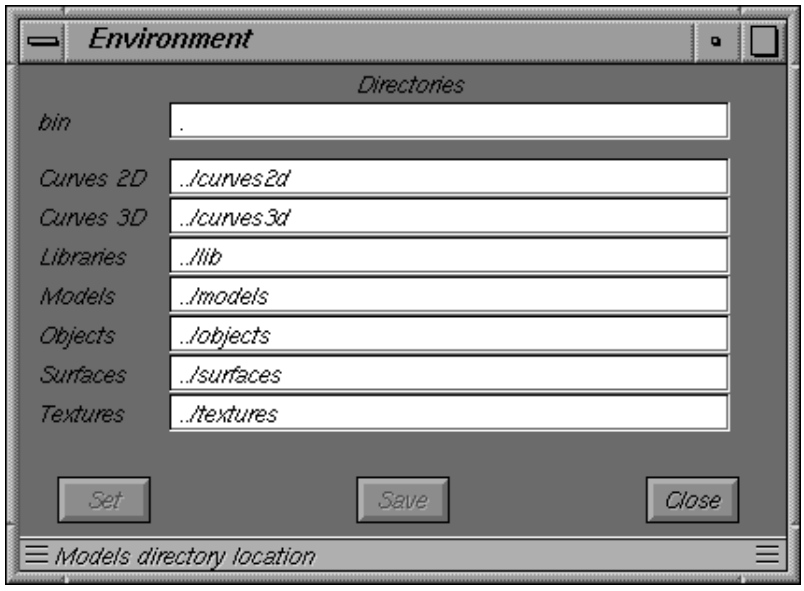


Figure 1.2: *xcmodel* environment setting

This program opens the console window in fig. 1.1 that allows each subsystem to also run at the same time and allows the user to set his/her

environment. In fact, the **Environment** button opens the window in fig. 1.2, from which it is possible to modify the directories site where *xcmodel* looks for the binary code, libraries and data files. This can be very useful for the user to personalize the system.
The **Set** button modifies the environment for the actual work session, while the **Set** and **Save** buttons modify the environment ever for successive work sessions (the .xcmodelrc file in the `bin` directory will be saved). The **Close** button closes this window.

CHAPTER 2

Tutorial: building a telephone

This tutorial will focus on building a sculptured object using curve and surface modelling techniques, boolean operations and rendering techniques. We start by modelling the different parts of our object.

2.1 Curve and surface modelling

- telephone base:
  - model some 2D section curves by *xccurv*;
  - position these curves in the 3D space loading them with *xcsurf* (see Fig. 2.1);

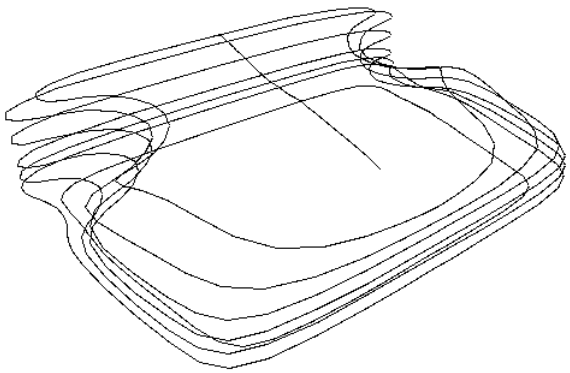


Figure 2.1: section curves

  - interpolate by skinning technique using *xcsurf* (see Fig. 2.2);
- receiver:

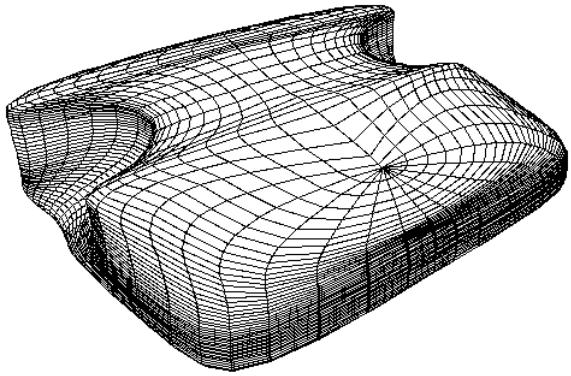
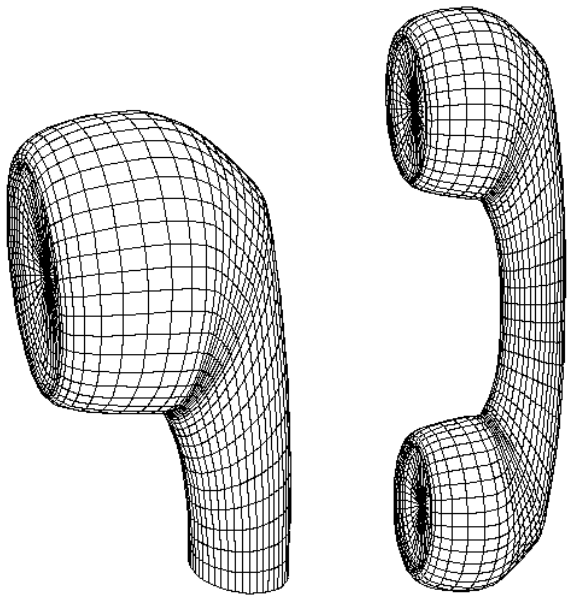


Figure 2.2: telephone base surface

- model interactively by *xcsurf* just half of the whole receiver (see Fig. 2.3a);
- create the second half by duplicating and rotating (see Fig. 2.3b);

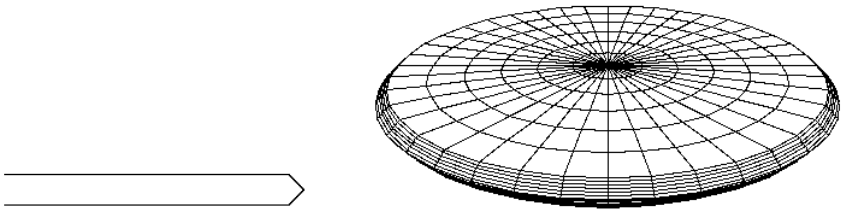


(a) (b)

Figure 2.3: receiver surface

- disk:
  - model the profile curve by *xccurv* (see Fig. 2.4a);

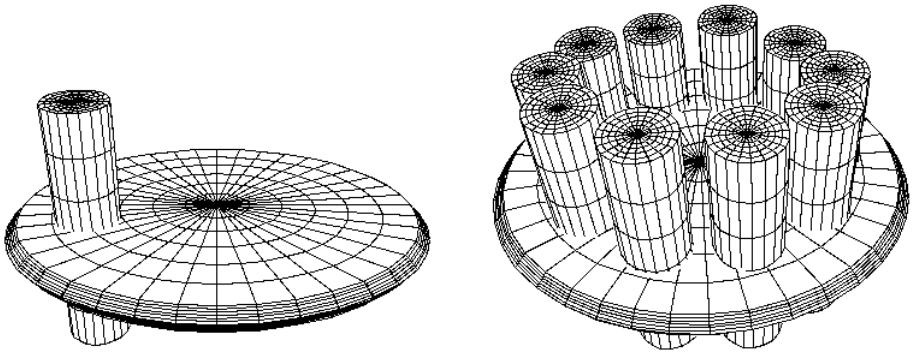
– model the disk by revolution using *xcsurf* (see Fig. 2.4b);



(a) (b)

Figure 2.4: profile curve and disk surface

– load a cylinder primitive with *xcsurf* and locate it with respect to the disk (see Fig. 2.5a);
– duplicate the cylinder and rotate it around the "disk axis" by 36 degree;
– repeat for the other cylinders (see Fig. 2.5b);

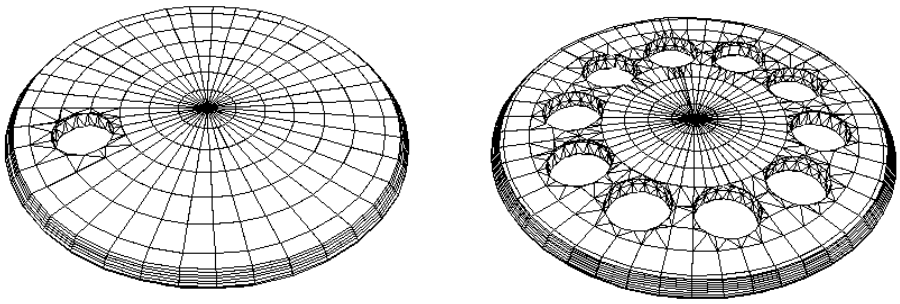


(a) (b)

Figure 2.5: disk and cylinder surfaces

2.2 Solid modelling

- disk:
  – compose the primitives disk and cylinder by the boolean operation difference in *xcbool* (see Fig. 2.6a);
  – compose, by the same operation, the resulting object with the other cylinders (see Fig. 2.6b);



(a) (b)

Figure 2.6: disk and cylinder difference

2.3 Rendering

Once you have created an object, you will want to look at it as a rendered image. This means to describe a virtual scene by defining the background, setting some light sources, defining the material peculiarities of the object, applying eventually a texture and then positioning the camera and setting the view parameters (see Fig. 2.7, 2.8 and 2.9). All these capabilities are provided by *xcrayt* system.

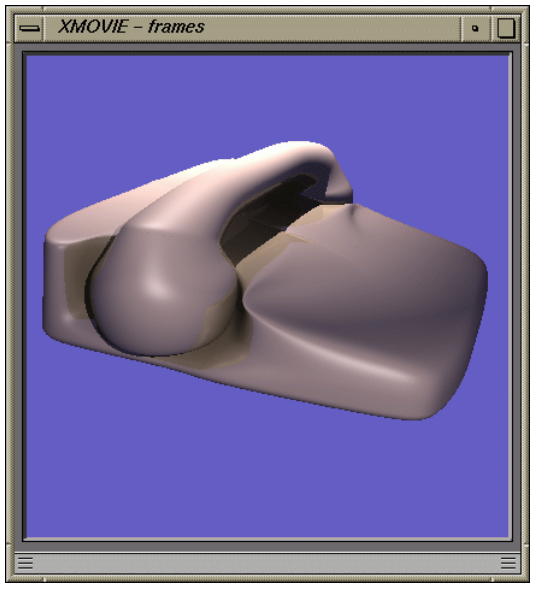


Figure 2.7: Telephone base and receiver image

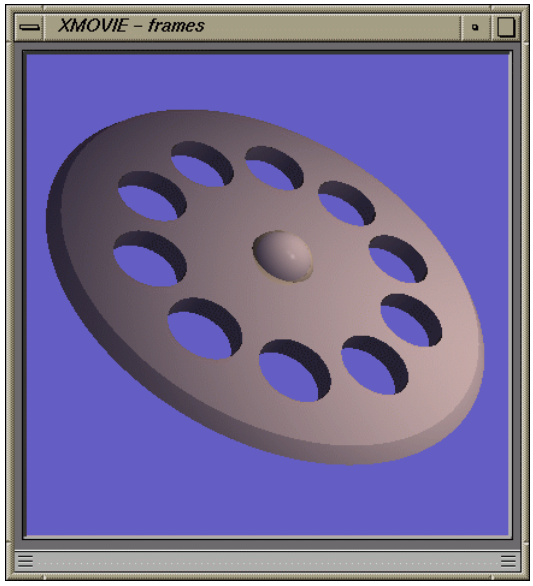


Figure 2.8: disk image

Figure 2.9: final telephone image

Chapter 3

Data file formats

This section, very important for a developer and less for a user, gives some information about the data files used in *xcmodel* and its subsystems. For each, we explain what data files are created and what data files are only used. Then we say what the data file extensions mean, referring to [XCCURV00] (xccurv.ps), [XCSURF00] (xcsurf.ps), [XCBOOL00] (xcbool.ps) and [XCRAYT00] (xcrayt.ps) documents for the file format syntax. This documentation can be found in doc.tar.gz.

3.1 *xccurv*

Data files made and used:

```
curves2d/
         .db
         .cp
         .ip
```

3.2 *xcsurf*

Data files made and used:

```
curves3d/
         <user_object>/
                        .db
                        .obj
surfaces/
         .cp
         .ip
         <user_object>/
                        .db
                        .dbe
```

```
                        .obj
```

Data files only used:

```
curves2d/
         .db
```

3.3 *xcbool*

Data files made and used:

```
objects/
        <user_object>/
                       .dbt
                       .int
                       .tree
                       .dbe
                       .obj
                       .csg
```

Data files only used:

```
surfaces/
         <user_object>/
                        .db
                        .dbe
                        .obj
```

3.4 *xcrayt* and *xmovie*

Data files made and used:

```
models/
       <user_model>/
                     .md
                     .vw
                     .arg
                     .sta
                     .hr
                     .ppm
                     .hra
```

Data files only used:

```
surfaces/
         <user_object>/
                        .db
                        .dbe
                        .obj
objects/
        <user_object>/
                       .dbe
                       .obj
textures/
         .hr
         .ppm
```

3.5 Data file extensions

- A **.db** (acronym of **data base**) file contains all the information to define a NURBS. It represents a NURBS 2d curve, 3d curve or surface, depending on the directory in which it is.

- A **.dbe** (acronym of **data base extended**) file contains all the information to define a trimmed NURBS surface. The trimming regions must be disjoint and defined by a list of 2d curves in the parametric domain.

- A **.tree** (stands for **tree**) file contains all the information to define a trimmed NURBS surface. The trimming regions are defined by a tree of 2d curves in the parametric domain.

- A **.obj** (stands for **object**) file contains a list of file names with the extensions .db, .dbe or .tree. We have used object instead of list, because we adopt the B-rep (boundary representation) for a solid object, which is defined by a list of trimmed NURBS surfaces .

- A **.csg** (acronym of **constructive solid geometry**) file contains the CSG tree for a composed object.

- A **.cp** (acronym of **control points**) file contains an array of control points and weights for a NURBS. It represents the control points for a 2d curve, a 3d curve or a surface, depending on the directory in which it is.

- A **.ip** (acronym of **interpolation points** but also for approximation points) file contains an array of points. It represents the points for a 2d curve, a 3d curve or a surface, depending on the directory in which it is.

- A **.int** (stands for **intersection**) file contains the results of a surface/surface intersection.
- A **.dbt** (acronym of **data base trace**) file contains the partial result of a boolean composition of NURBS solid primitives.
- A **.md** (stands for **model**) file contains all the information to define the geometric and light model of a 3D scene.
- A **.vw** (stands for **view**) file contains all the information to define the camera parameters to view a 3D scene.
- A **.arg** (stands for **arguments**) file contains all the parameters for rendering the scene with the ray-tracer hrayt.
- A **.sta** (stands for **statistics**) contains all the information about the execution of the ray-tracer hrayt.
- A **.hr** (stands for **hrayt**) file contains a compressed image, produced by hrayt, that is the rendered scene.
- A **.ppm** (Portable Bitmap Utilities) is a standard graphics format in Xwindow-Unix environment) file contains an image in binary form with an ASCII header.
- A **.hra** (stands for **hrayt animation**) file contains a list of file name with extension .hr or .ppm. It represents an animation.

List of Figures

1.1 *xcmodel* console window 12
1.2 *xcmodel* environment setting 12

2.1 section curves 15
2.2 telephone base surface 16
2.3 receiver surface 16
2.4 profile curve and disk surface 17
2.5 disk and cylinder surfaces 17
2.6 disk and cylinder difference 18
2.7 Telephone base and receiver image 18
2.8 disk image 19
2.9 final telephone image 19

Bibliography

[XCCURV00] G.Casciola, *xccurv*: the 2D modeller; 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

[XCSURF00] G.Casciola, *xcsurf*: the 3D modeller; 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

[XCBOOL00] G.Casciola, G.De Marco, *xcbool*: the object composer; 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

[XCRAYT00] G.Casciola, *xcrayt*: the scene descriptor; 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