BoardCAD developer guide

Jonas Hörnstein
2007-01-17

Introduction

Optain the source code

We use SVN (Subversion) to manage the changes to the source code. SVN stores not only the current version of the code, but also keeps track of all the changes and who made them. You need an SVN client to access the code. If you use Linux you are likely to have SVN installed, otherwise you can download the official SVN client at http://subversion.tigris.org (available for Linux, Mac OS X, and Windows). Windows users can alternatively use TortoiseSVN (http://tortoisesvn.net) which is integrated into Windows Explorer and very easy to use.

Our Subversion repository can be checked out through SVN with the following instruction set:
svn co https://boardcad.svn.sourceforge.net/svnroot/boardcad boardcad

If you are using TortoiseSVN, simply create a folder named boardcad, right click on the created folder and choose SVN checkout and simply type in the URL of the repository (see screenshots below).

Everyone are allowed to read the code, but to be able to write you have to be a member of the boardcad project at Sourceforge. If you want to be part of the developers you have to create an account at sourceforge and mail us your username so we can include you in the project.

Install JDK and Java 3D

To compile the code you need to have the Java SE Development Kit (JDK) installed:
http://java.sun.com/javase/downloads/index.jsp

You also have to install Java 3D:
http://java.sun.com/products/java-media/3D/index.jsp

Compilation

To compile the program simply go to the folder where you put the source code, e.g. c:\boardcad and type:

```
javac BoardCAD.java
```

You will get a warning that we use a deprecated API. Don't worry, this is normal... You can now run the program by typing:

```
java BoardCAD
```

Program structure

This is a very short description of the structure of the program. I will try to find time to make a more complete description and even include some UML diagrams, but for now I only give some short descriptions of the most important classes.

BoardCAD
This is the main class. BoardCAD is actually implemented as an applet, but is mainly intended to be run as a stand alone application. The only thing this class does is to create the user interface.

UserInterface
The user interface is designed to be light weight, i.e. we try to do as little processing as possible at the interface to make it easy to change. The main purpose of the interface is to collect user input and send it to the BoardHandler.

The current UserInterface has a menu on top, then a large design panel in the middele (which contains four views that can either be displayed all at the same time or once at a time), and finally a status panel in the bottom.

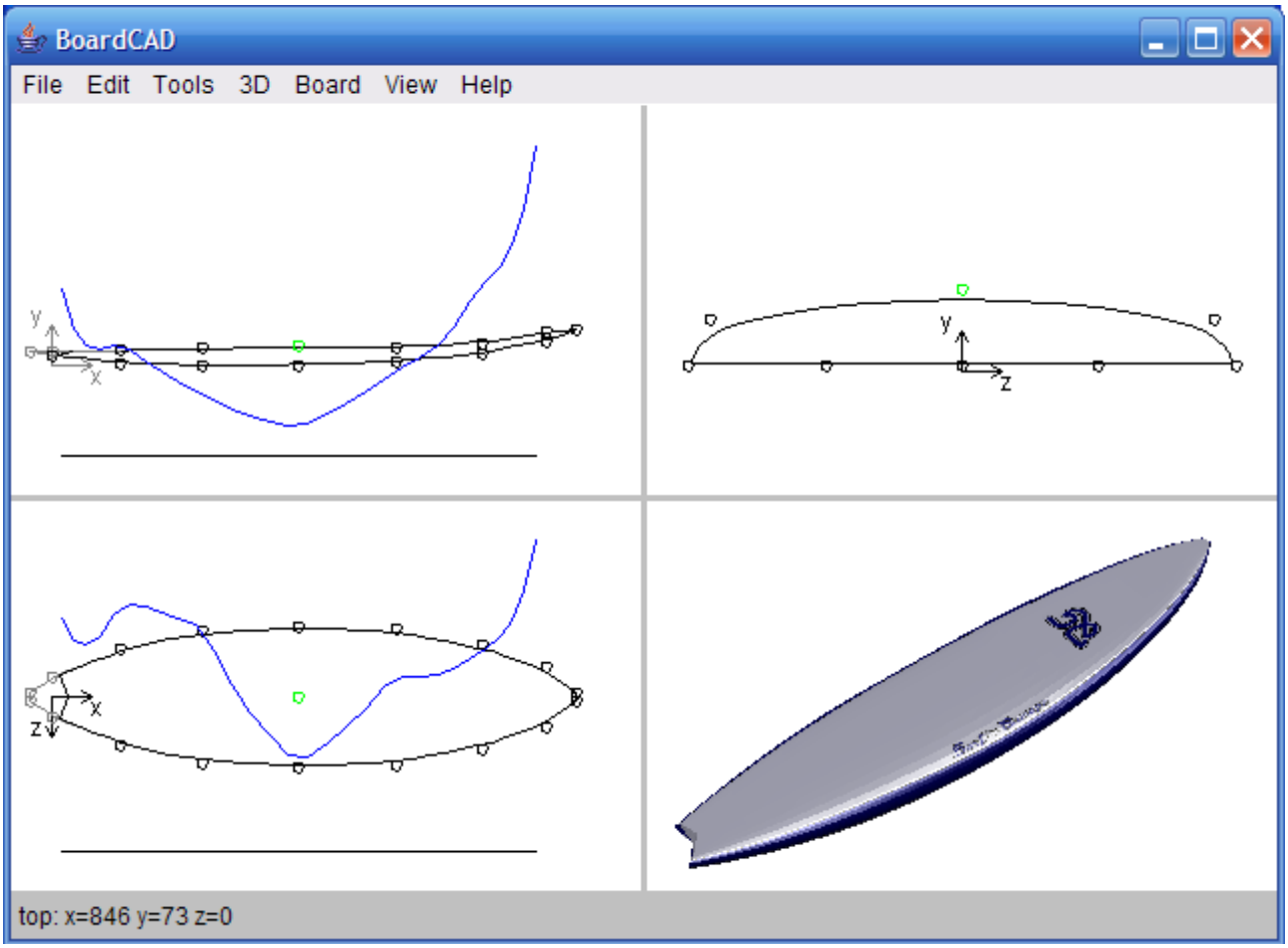

Figure 1. The user interface

The following classes are only used by the UserInterface and are not described in detail for the moment:

- **StatusPanel**
- **DesignPanel**
- **View (abstract class)**
- **RockerView**
- **EdgeView**
- **OutlineView**
- **ThreeDView**

The UserInterface also has a number of Dialogs classes that it can show in order to collect more input from the user. These are:

demodialog – used to input help point, i.e. points that the user have measured from another board and wants to use as a guide when modelling the current board.

Taildialog – used to set the form of the tail. Currently two types of tails can be used: swallow tail (which can also be used to create squash tails) and pin tails (which can also be used to create a perfectly round tail).

Print2DtoStream – sets up a printer

BrowserControl – opens the project web page for showing help files (when those are ready)

AboutDialog – should show some information about the current version of the program.

BoardHandler
The BoardHandler contains a list of all boards that are currently open and make sure that the user inputs reach the right board. All the communication between the board and the user interface have to pass through the BoardHandler. Currently most of the information is just passed on to the active board, but the BoardHandler will allow things like showing none active boards in the background if user wishes so. It is also the BoardHandler that maintains the list of all help points so these can be shown for any board.

Board
The Board class contains the mathematical definition of the board and is also responsible for loading and saving the definition from or to a file, exporting the definition as a STEP-file, and drawing the board, either on a printer or on a graphical panel provided by the UserInterface through the BoardHandler.

Mathematically the board is defined by two NURBS surfaces, one for the deck and one for the bottom. Currently these surfaces are of fixed size, but we might choose to change that... Another limitation in the current model is that we always use a tripple point in the center of the bottom surface. This creates a V-form for the bottom (also flat bottoms and dubble concaves are possible). In order to create a single concave we have to remove this limitation... On top of the NURBS model we have the tail designer. If we use a swallow tail we simply cut the surfaces one segment from the tail at a given angle, resulting in either a swallow tail or a squash tail. If we use a pintail we also cut the tail at the same segment, but use Bezier-curves to get a smooth transition to the deck surface.

There is a lot more to be said about the board model and will be added in the future...

The mathematical definition use the following classes:

- **Surface –** implements a NURBS surface
- **Point -** implements a point in 3D