Inserting an *ebm* shape in MGED

Making an image with GIMP

First, we must make an image with GIMP. This image contains the text, or a two dimensional shape that should be loaded into MGED model as *ebm*.

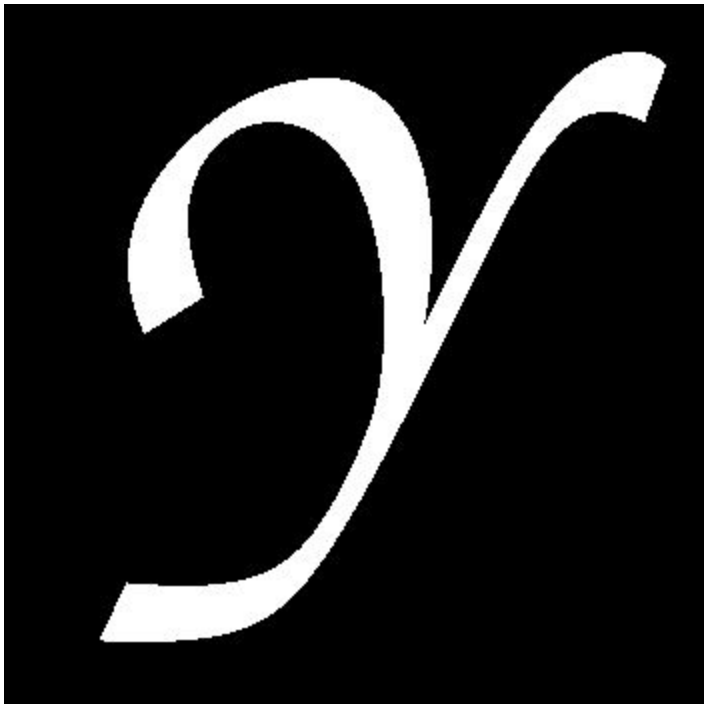
The image must have a **black** background and a **white** foreground.

To get a nice *ebm*, the new GIMP image should have at least (?) 350 x 350 pixels (width and height).

In GIMP we must change the Image Mode to Indexed Colors with 'Black-white, using the (1 bit) palette' option.

We must save this image in PNG format. In the Save dialog box we must check the 'Save with Background' option.

We must write down somewhere the dimensions in pixels of this image (e.g., 350x350), as this information will be needed when the *ebm* is created in MGED.



The GIMP Image

Converting a PNG file into a bitmap file

This can be done with the following command at the shell prompt in an XTERM window on GNU/Linux systems. If BRL-CAD had been installed and added to PATH, then this command will work on Windows as well.

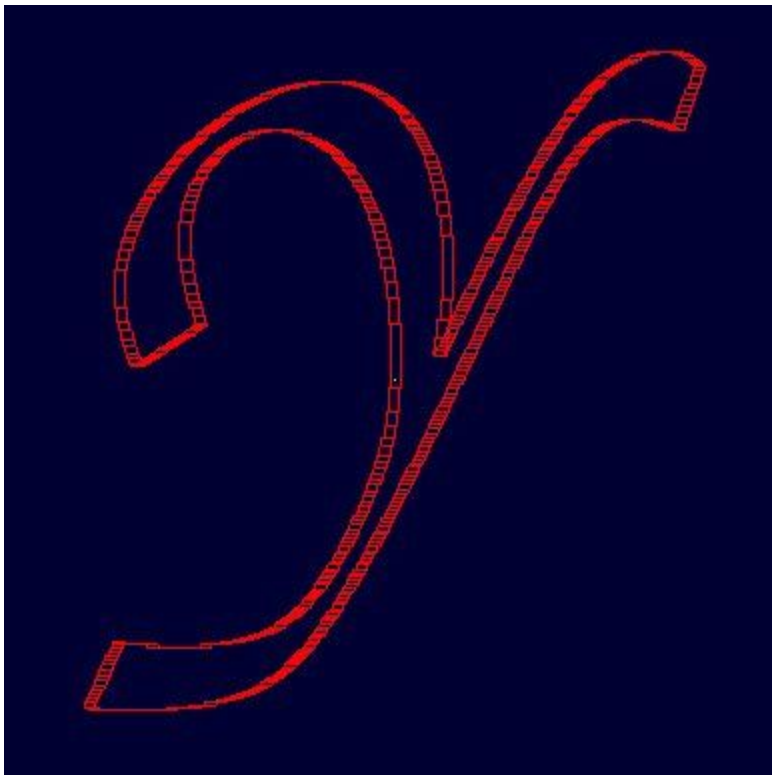
```
$ png-bw image_name.png > image_name.bw
Warning: color image being converted to B/W!!!
```

We get the warning shown above and this is good :)

Inserting *ebm* in MGED

```
mged> in text1.s ebm f image_name.bw 350 350 20
```

If we don't type the dimensions (in pixels) of the file image_name.png exactly as it was created, the *ebm* shape that we get will not be the desired result.



The Wireframe of the *ebm*

If we made a region of this *ebm* and raytraced it, then we can see something like the picture below.

Importing .bw data to a database object

An alternative method to create an *ebm* is to use a binary database object, which can store a .bw file's data, rather than directly use a .bw file as the data source. While the results will be the same, the advantage of this is that the *ebm* does not need to rely on an external file, meaning that it will still function even without the presence of the original .bw file. This can be done through the following steps:

The first step is to actually create the binary database object that will store the .bw file data. This can be done through the 'bo' command.

```
mged> bo -i -u C dbobj image_name.bw
```

This uses the contents of the file image_name.bw to create a binary database object of unsigned characters called dbobj. Note that ‘C’ is used to denote the unsigned char type, as this is what *ebm* reads.

Creating an *ebm* using the binary database object as the data source is similar to before, with a few exceptions.

```
mged> in text2.s ebm o dbobj 350 350 20
```

In this case, there is an ‘o’ indicating an object source instead of an ‘f’ and the name of the source is now the name of the binary database object rather than the name of a file.

Once the EBM has been created, the original .bw file can be removed as its data will still be stored internally within the binary database object.