Why Bitmapped Screen Dumps used as OHs Sometimes get Ugly

By professor Jacob Palme,
Department of Computer and Systems Sciences
KTH Technical University

Abstract

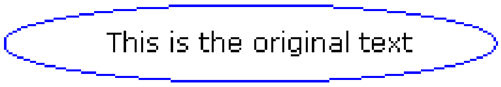
Computer screen dumps will sometimes look very ugly when you view them in manuals and help texts. This document analyzes why and suggest how to avoid this problem.

The Problem

When you dump a computer screen content in a bitmapped file, this file will usually get the same resolution as is used for the computer screen, 72 pixels/inch or 96 pixels/inch.

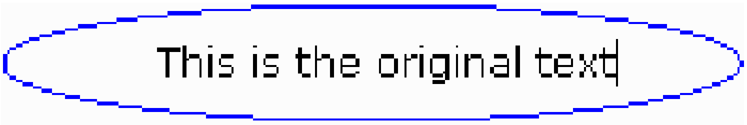
If you then use this bitmapped picture in an overhead or in a document, which is to be read on a computer, the picture looks OK if it is shown at the same resolution as it was produced. If, however, the resolution is changed, the picture can sometimes become very ugly.

If, for example, the original bitmapped picture looked like this:

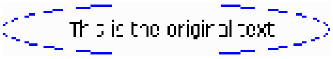


The original bitmapped picture used in all the examples. Verdana text, 12 pt, 72 dots/inch resolution.

It looks OK if shown with the original size as above. If, however, it is shown in reduced or enlarged size, it will look very ugly:



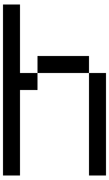
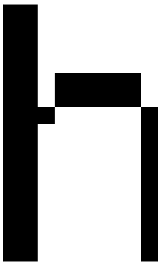
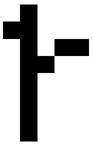
The same image increased to 1,5 times original size.



The same image reduced to 2/3 of original size.

This problem will especially occur if you show the document with a program which allows the increase or reduction of the size of the document on the screen, such as Adobe Acrobat or Microsoft Powerpoint. Powerpoint, using anti-aliasing.

The explanation can be seen if you increase the size of the letter "h" from the original and the two revised pictures. In the table below, the size of these characters has been increased 8 times, so that you can see them more clearly.

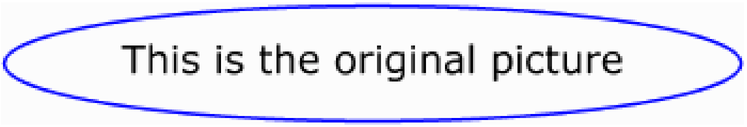
	**Original "h"**	The oriignal "h" was a bitmapped picture 6 pixels wide and 10 pixels high.
	**"h" increased 1,5 times**	As you can see here, there is no good way to increase bitmapped picture by 50 %. The only choices are to either leave the line widths as they are, or double the line widths. The choice between these alternatives seems to have been done rather randomly.
	**"h" reduced to 2/3**	When you reduce by 2/3, it means that you have to remove 1/3 of the pixels. Since the original "h" was composed of lines 1 pixel width, some of them became 0 length after reduction. Not very good either.

Solution 1: Show in Original Size

The simplest solution to this problem is to ensure that the pictures are always shown in their original size. This is the method used by web browsers, and web browsers are, because of that, sometimes better when displaying documents containing screen dumps.

Solution 2a: Produce in very High Resolution

Another solution is to produce the picture in very high resolution. If, for example, the picture in the example above is produced with 600 pixels/inch, and its size is then reduced, it will still look fairly good. Example

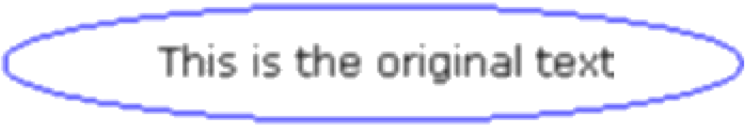


High-resolution bitmapped picture reduced to 72 pixels/inch

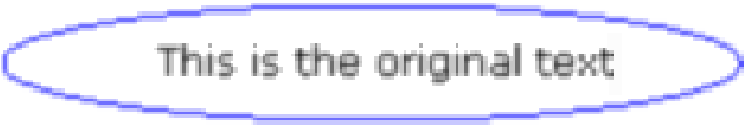
Solution 2b: Convert to very High Resolution and Blur

This is variant of Solution 2a, where you have to start with the original picture in only a low resolution format. You can convert the low resolution format to a high resolution format and then blur the picture, which will make it behave a little better when reduced to low resolution while viewing.

Below is an example, where the original picture used in all the examples above was converted in this way.



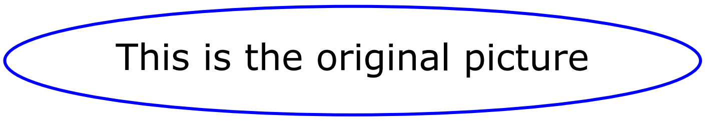
The picture to the left was first converted from 72 to 720 pixels/inch, then blurred, then converted back to 72 pixels/inch, and then actually shown in 1,5 times enlargement:



A simpler variant, the original picture was blurred as is, and then shown in 1,5 times enlargement.

Solution 3: Store Picture in Object Format instead of Bitmapped Format

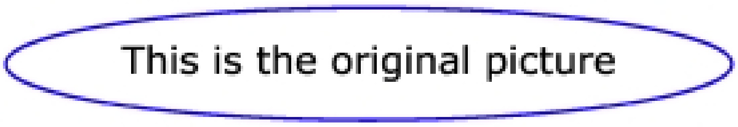
The best solution should be to store the picture in an object oriented format instead of a bitmapped format. With this method, the picture should always get the best possible rendering on the screen. Example of how it should look like (my experients, see "solution 4" below were not so positive, however, when trying to embed PDF and Flash in HTML):



Object picture

Solution 4: Embed objects in other formats handled by plug-ins

I also experimented with inserting the picture using plugins. Before this, I converted the text to paths, so that the result should not rely on the right fonts available. Here are the results:

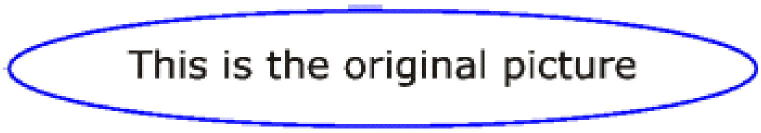


Embedding PDF in web page. This will only work with Explorer, not with Netscape 4.7.

This does not look as neat as it should, probably because the functionality in Explorer for embedding PDF in HTML has some problems, or because I did not understand correctly how to use it.

The HTML code used for the picture above was:

```
<object type="application/pdf" codetype="application/pdf" height=114
width=348 data="original-paths.pdf">
```



Embedding Flash in a web page.

This embedding was done using the following :-) neat and simple code, which Dreamweaver produced automatically for me:

```
<object  classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=5,0,0,0"
width="348"  height="57">
<param  name=movie  value="original-paths-150-perc-inc.swf">
<param  name=quality  value=high>
<embed  src="original-paths-150-perc-inc.swf"  quality=high
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash"
type="application/x-shockwave-flash"  width="348"  height="57">
</embed>
</object>
```