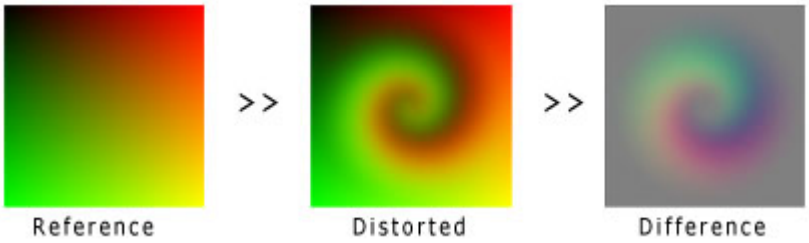
Calculating displacement maps based on reference maps

I already blogged about this, but I saw that some items could use some more explanation. Please read “Getting the displacement you need” first for the basic idea.

In short the principle is:

- you have a reference map, say A
- you apply a distortion effect to the reference map created a distorted version of A say A’
- you calculate the displacement map that could transform A into A’ by calculating the difference in pixel value between A and A’



Some important points before we start:

Use a **lossless** image format (eg bmp). If you use lossy compression anywhere in the pipeline, you are throwing information away. This missing information results in artifacts in the displacement map. So, save all images as BMP for example. Import them into flash and set the properties to lossless.

We will now describe some steps in my previous post in some more detail.

Creating the reference map

Option 1 – Creating the reference map in photoshop

————————————————————-

- create a new image 256 x 256
- add another layer
- linear gradient fill the bottom layer from left to right, black to red
  (while filling keep shift pressed to make sure the gradients are vertical/horizontal &
  zoom in a bit to make sure you fill the whole area)
- linear gradient fill the top layer from top to bottom, black to green
- set the top layer to difference, or screen, or lighten (doesn't matter in this case)
- verify the top left pixel has RGB value (0,0,0) and the bottom right has (255,255,0), if this is not the case, start over

Option 2 – Creating the reference map through code in Flash
————————————————————-

Copy & paste in imaging programming and save.

Creating the transformed map/image

- Open up your base reference map
- Apply a distortion filter, eg Spherize
- Save the image

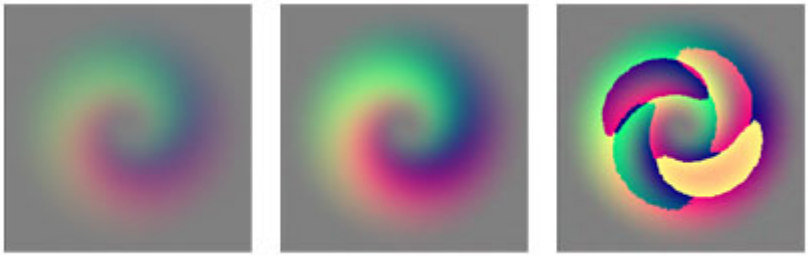
Calculating / deducting the displacement map

- import both the base reference map A and the transformed reference map A' into Flash
- set the properties to lossless
- export them as base and transformed
- run the difference calculation code below

Scaling the displacement map / Preventing artifacts
To prevent confusing there are two ways of scaling, one is to scale the displacements, and two is to resize the displacement map itself. This technique applies to both types.
As you scale the displacement map, you might see artifacts

appearing. How bad these artifacts are depends on how you are scaling the map and how much you are scaling it. Since the displacement map can only displace pixels from -128 to 128 without scaling, the feature has its limits.

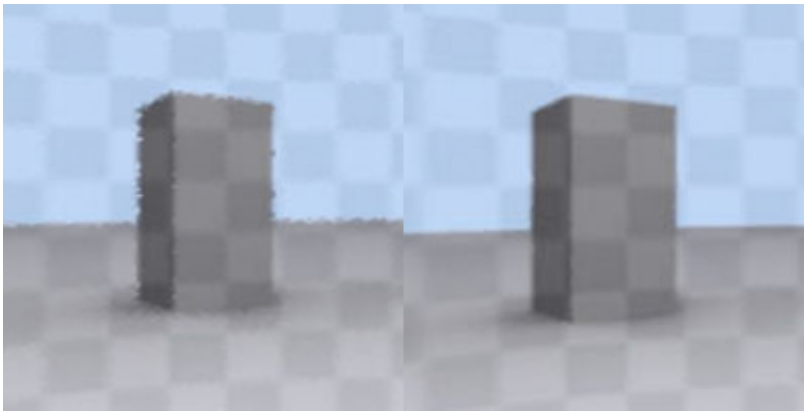
What might help in preventing artifacts is maximizing the pixel difference, so that less scaling in the filter in needed. How far you can maximize it depends on the image itself. Below is an example of multiplying the difference until it breaks:



The second image is useable, but the 3rd is not (unless you want some glass like special effect, then it is very nice).

Another thing that can help preventing artifacts is blurring. Basicly if you resize the displacement map, any lines that you see in the map, will be visible in the result as well. Blurring those lines will improve the result.

The image on the left shows artifacts due to displacement map scaling, the image on the right uses the same map, but blurred.



The code for this post can be downloaded here: BitmapDifferenceExample (485).

Note that if you generate your displacement maps through code a lot of this information does not apply:). The next post will go deeper into the panoramic displacement mapping.