Invisible Watermarking Algorithm

Embedding Algorithm

- Get the host image to be watermarked
- Get the binary watermark image
- Get the private key for encoding of data
- Encode and hash the binary watermark with the user key
- Divide the host image into 8 x 8 blocks
- Convert each RGB block to its corresponding YCbCr representation
- DCT transform the Y component
- Mark the low order DCT coefficients as

  Coeff = Coeff * (1 + alpha * val)

  Where

  alpha is the embedding factor

  val = +1 if watermark bit is 1

  = -1 if watermark bit is 0.
- Take inverse DCT transform of each block
- Convert the YCbCr representation to corresponding RGB representation.

Extraction Algorithm

- Get the suspect image and the original cover image.
- Get the watermark image and key that is to be tested.
- Divide both suspect and original image into 8 x 8 blocks
- Convert from RGB space to YCbCr representation
- DCT transform blocks from both images
- Compare corresponding image blocks

- If the summation of the DCT coefficients in the suspect image block is larger than the corresponding coefficients in the original image block then watermark bit is 1, else it is 0.
- Compare the extracted sequence with the encoded binary watermark to make a decision whether the image is authentic or not.

This algorithm is implemented by **Rajan Sheth, Adrain Pinto and Nitesh Chawada.**
http://www.geocities.com/gwatermarker

The original algorithm was only for grayscale images. It was first modified by **Adrain Pinto** and then implemented for color images.

<u>Modification includes</u>

- Using a private key for encoding the binary data using **blowfish algorithm** and hashing it using **RC4 algorithm.**
- converting the image from **RGB to YCbCR** model and then performing the DCT transform and other manipulation.