# Huffman-encoded images

1. Encoded Image

Input: ppm images, raw bitmap of images

Output: encoded ppm images and huffman code

Pipeline: Image can be represented as matrix of (R, G, B) tuple, and RGB range in [0,255], so get frequency of an array of 256, and then using huffman coding algorithm, which greedyly get the two integers with lowest frequency and then delete two nodes(integer and frequency representation) and insert an integrated node into tree, and for n-1(n is 256 here) times, which get the huffman tree, and then traverse the tree to get code of each leaf node.

As image processing is easy for python, PIL lib, so I use python here.

As you kindly mentioned, gogol.huffman_code is a little strange, because I forget to delete zero occurrences pixel, which make the tree very long, and leaf many zero occurrences pixels, now I removed these elements for huffman coding, and it seems normal now. Thanks.

As you kindly mentioned, huffman encode and decode really do not need delimitor, I just for enhance decoding efficiency, it is ok now, but decoding need check each prefix code length from 1 to maximum huffman code for finding the correct huffman code for this image. Thanks.

2. Decoded Image

Input: encoded ppm images and huffman code

Output: decoded original image

using hashmap to store the map of huffman code to integer of pixel of each color, dict in python, it can efficiently decode images. And show it as a jpeg image.

3. I change a few bits of encoded images, and it is precisely hard to say which bits flipping will effect the image decode, or can decode but display very different image compared to original image.

   a) If the bits flipping make the huffman decoded failed, that means can not find the prefix huffman code, and can not decode.

b) If the bits flipping make the huffman decoded still success, that means not fatal error, but may display very different from original image, because it will probably effect the behind pixel recovery. Commonly less than 5 bits flipping may probably do not effect huffman decoding process.