Image Processing Tutorial

DCT

The calculation of the DCT components are:

$$F(u,v) = \frac{1}{4} C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right]$$

where $C(z) = \frac{1}{\sqrt{2}}$ if $z = 0$

or $= 1$ if $z \neq 0$

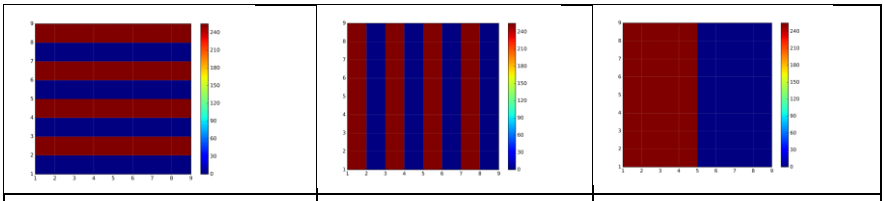
The input intensity pixel bock is:

[[16, 11, 10, 16, 24, 40, 51, 61], [12, 12, 14, 19, 26, 58, 60, 55], [14, 13, 16, 24, 40, 57, 69, 56], [14, 17, 22, 29, 51, 87, 80, 62], [18, 22, 37, 56, 68, 109, 103, 77], [24, 35, 55, 64, 81, 104, 113, 92], [49, 64, 78, 87, 103, 121, 120, 101], [72, 92, 95, 98, 112, 100, 103, 99]]

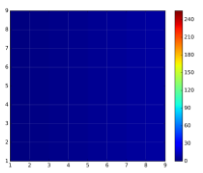
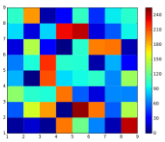
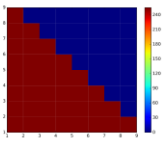
prove that the F(0,0) value is 461:

```
[[ 461   -168   -14   30   -31   8    1    -2]
[-194    -1     38    4    6    3    5    -5]
[         43    10   -22   15   -10  -5    4]
[        --27   -1    0    -2   7    4    -3]
[         11    0     1    4    -4   1    -2]
[         -3    2     2    0    -2   0    2]
[        -6     4    -8    7    -1   -6   7]
[         -1    -3    1    2    3    -1   -1]]
```

For the following, match the pixel blocks to the DCT arrays:



(a)	(b)	(c)

		
(d)	(e)	(f)

Pixel blocks:

1. [[0, 0, 0, 0, 0, 0, 0, 0], [255, 255, 255, 255, 255, 255, 255, 255], [0, 0, 0, 0, 0, 0, 0, 0], [255, 255, 255, 255, 255, 255, 255, 255], [0, 0, 0, 0, 0, 0, 0, 0], [255, 255, 255, 255, 255, 255, 255, 255], [0, 0, 0, 0, 0, 0, 0, 0], [255, 255, 255, 255, 255, 255, 255, 255]]

2. [[255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0], [255, 0, 255, 0, 255, 0, 255, 0]]

3. [[255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0], [255, 255, 255, 255, 0, 0, 0, 0]]

4. [[0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7], [0, 1, 2, 3, 4, 5, 6, 7]]

5. [[6, 18, 4, 200, 123, 65, 11, 241], [54, 154, 190, 2, 251, 201, 55, 143], [133, 102, 99, 201, 55, 21, 66, 65], [77, 1, 211, 87, 95, 103, 66, 137], [63, 98, 220, 102, 100, 9, 76, 32], [21, 144, 29, 1, 102, 198, 201, 11], [87, 21, 85, 230, 241, 21, 54, 94], [103, 191, 9, 32, 105, 43, 91, 102]]

6. [[255,255,255,255,255,255,255,255],[255,255,255,255,255,255,255,0], [255,255,255,255,255,255,0,0], [255,255,255,255,255,0,0,0], [255,255,255,255,0,0,0,0], [255,255,255,0,0,0,0,0], [255,255,0,0,0,0,0,0], [255,0,0,0,0,0,0,0]]

Check your answers here: http://asecuritysite.com/calculators/dct2

Face Processing

We can use thresholding techniques to be able to detect faces, and features within a face. Implement the code at:

https://asecuritysite.com/comms/face

And download a photograph of the face of the people defined below, and see if you can detect their face, smile and eyes. Outline if you have any difficulties in the processing, and the possible reason why the feature is not properly detected:

David Cameron: David Beckham: Kim Kardashian:

JPEG Processing

The JPEG file starts with the hex sequence of "FF FD". There are also other sequences using the file, such as:

- FF F8 tag – Start of file
- FF DB tag (Quantization Table).
- FF C4 tag (Huffman Table).
- FF C0 tag (Start Of Frame (Baseline DCT)).
- FF DA tag (Start Of Scan).
- FF 00 stuffed FF (Likely Huffman Coding).

Write a Python program which reads in a JPEG file, and displays these codes. An example program to detect the start of the file is given here:

```
import binascii

with open('index.jpg', 'rb') as f:
  byte_s = f.read(1)
  hex_byte = binascii.hexlify(byte_s)
  while byte_s:
    if (hex_byte=="ff"):
      byte_s = f.read(1)
      hex_byte = binascii.hexlify(byte_s)
      if (hex_byte=="d8"): print ("FF D8: Start of file")
    else:
      byte_s = f.read(1)
      hex_byte = binascii.hexlify(byte_s)
```

Now download the following file and process it with your program:

http://asecuritysite.com/log/file04.jpg

Check your results against:

http://asecuritysite.com/forensics/jpeg?file=file04

Did you program identify the correct tags?