TERM PROJECT

JPEG COMPRESSION

By: Fola Aluko

CS 543 SP 23

Report

This project covered the JPEG compression algorithm and how it can be used for lossy compression, while keeping major information related to the image as clear as possible. For the intermediate analysis below, I used “alu.png” as my input and a quality scale factor of 30.

VALUE CHANGES OF 8x8 PIXEL BLOCKS DURING COMPRESSION

```
Original Image Output
   148   145   157   182   211   194   161   174
   167   145   144   152   172   175   162   161
   178   167   159   150   156   162   155   150
   163   166   170   162   156   158   155   149
   115   123   153   166   165   161   157   168
   125   126   152   169   175   175   174   173
   192   174   172   181   188   204   200   180
   182   171   176   180   181   194   194   176
```

After YCbCr conversion: this shows luminance and chrominance channels.

```
YCbCr Image Output

(:,:,1) =

  136.0390  129.7280  134.1480  161.9570  171.4940  154.5400  135.7200  141.1400
  140.5630  120.5980  116.3040  131.8550  144.1950  136.4180  125.3560  125.7580
  146.9010  127.6320  125.5990  112.4730  123.4310  122.9910  119.0400  120.1210
  155.0510  141.1590  129.8000  124.8490  109.8160  110.0550  118.2080  121.2240
  144.4660  130.0800  132.5760  115.8380  113.4360  114.5880  124.2200  129.8570
  147.5700  128.9710  127.2390  117.0770  130.7420  135.4550  139.5100  134.8910
  159.4020  142.1490  128.7680  119.8500  139.5700  152.8340  150.5950  133.1200
  143.1730  125.5510  124.3390  122.8450  132.0800  147.0060  145.1650  130.3910


(:,:,2) =

   43.9316   49.7504   45.5630   25.9191   13.2006   25.0257   40.7255   36.5381
   38.5568   50.3880   53.3756   42.9067   32.5568   34.6882   43.1882   43.5257
   35.5443   48.6758   48.6944   53.8445   46.5319   47.3445   46.1882   46.7069
   38.2815   45.5568   50.8383   49.6820   53.6509   54.6447   48.3507   47.2131
   42.5622   48.9876   50.4005   53.6384   53.3010   52.6509   42.7006   39.5195
   41.3748   51.3064   51.7193   54.6322   44.6633   39.1820   35.2006   33.8569
   31.8756   43.8694   49.1633   51.9386   37.9884   25.9884   24.9946   33.1633
   36.5195   48.1571   49.9697   49.1198   41.6509   32.6633   27.4946   35.8321


(:,:,3) =

    9.0316   11.3933   16.7998   14.7962   28.6784   28.6457   18.5316   23.9382
   19.3569   17.9054   20.2549   14.8690   20.3326   28.0195   26.6372   25.6372
   22.6821   28.5802   24.3241   27.2671   23.7307   28.3241   26.1493   21.8120
    6.1700   18.2185   29.1736   26.9989   33.4418   34.6979   26.7428   20.3120
  -20.5169   -4.5497   15.0680   36.2792   37.2792   33.6045   23.8811   27.7064
  -15.5983   -1.6189   18.1615   37.5353   32.0680   28.7064   25.1008   27.6821
   23.7513   23.2185   31.3363   44.1166   35.0438   36.9952   35.7391   33.9382
   28.1942   32.9176   37.3484   41.2671   35.3933   34.0195   35.3326   33.0316
```

After Subsampling:

```
Chroma Subsampling Image Output

(:,:,1) =

  136.0390  129.7280  134.1480  161.9570  171.4940  154.5400  135.7200  141.1400
  140.5630  120.5980  116.3040  131.8550  144.1950  136.4180  125.3560  125.7580
  146.9010  127.6320  125.5990  112.4730  123.4310  122.9910  119.0400  120.1210
  155.0510  141.1590  129.8000  124.8490  109.8160  110.0550  118.2080  121.2240
  144.4660  130.0800  132.5760  115.8380  113.4360  114.5880  124.2200  129.8570
  147.5700  128.9710  127.2390  117.0770  130.7420  135.4550  139.5100  134.8910
  159.4020  142.1490  128.7680  119.8500  139.5700  152.8340  150.5950  133.1200
  143.1730  125.5510  124.3390  122.8450  132.0800  147.0060  145.1650  130.3910


(:,:,2) =

   43.9316   43.9316   45.5630   45.5630   13.2006   13.2006   40.7255   40.7255
   38.5568   38.5568   53.3756   53.3756   32.5568   32.5568   43.1882   43.1882
   35.5443   35.5443   48.6944   48.6944   46.5319   46.5319   46.1882   46.1882
   38.2815   38.2815   50.8383   50.8383   53.6509   53.6509   48.3507   48.3507
   42.5622   42.5622   50.4005   50.4005   53.3010   53.3010   42.7006   42.7006
   41.3748   41.3748   51.7193   51.7193   44.6633   44.6633   35.2006   35.2006
   31.8756   31.8756   49.1633   49.1633   37.9884   37.9884   24.9946   24.9946
   36.5195   36.5195   49.9697   49.9697   41.6509   41.6509   27.4946   27.4946


(:,:,3) =

    9.0316    9.0316   16.7998   16.7998   28.6784   28.6784   18.5316   18.5316
   19.3569   19.3569   20.2549   20.2549   20.3326   20.3326   26.6372   26.6372
   22.6821   22.6821   24.3241   24.3241   23.7307   23.7307   26.1493   26.1493
    6.1700    6.1700   29.1736   29.1736   33.4418   33.4418   26.7428   26.7428
  -20.5169  -20.5169   15.0680   15.0680   37.2792   37.2792   23.8811   23.8811
  -15.5983  -15.5983   18.1615   18.1615   32.0680   32.0680   25.1008   25.1008
   23.7513   23.7513   31.3363   31.3363   35.0438   35.0438   35.7391   35.7391
   28.1942   28.1942   37.3484   37.3484   35.3933   35.3933   35.3326   35.3326
```

After 2D DCT:

```
Warning: Image size adjusted to [392, 568]
DCT Transform Image Output

(:,:,1) =

   1.0e+03 *

    1.0597    0.0145    0.0233    0.0358    0.0106    0.0051    0.0075    0.0078
   -0.0022    0.0046   -0.0295   -0.0090    0.0230   -0.0070   -0.0017   -0.0049
    0.0432   -0.0346   -0.0347    0.0219    0.0057   -0.0037   -0.0018   -0.0039
    0.0273   -0.0148   -0.0126   -0.0027    0.0056   -0.0037   -0.0013    0.0012
    0.0029    0.0047   -0.0069   -0.0174    0.0036    0.0041   -0.0018   -0.0057
    0.0226    0.0098    0.0006    0.0024   -0.0029   -0.0018   -0.0035   -0.0028
   -0.0027   -0.0060   -0.0038   -0.0032   -0.0004   -0.0010    0.0038    0.0033
    0.0014   -0.0010    0.0010   -0.0002   -0.0045   -0.0032    0.0039    0.0045


(:,:,2) =

  335.1893   10.0460  -24.4316  -20.7121    0.0000   13.8394   10.1199   -1.9983
    3.6533   -6.6102   25.4092  -15.3252   -0.0000   10.2400  -10.5248    1.3149
  -31.6386   20.2367   10.2902  -16.1035   -0.0000   10.7600   -4.2623   -4.0253
   -7.0332   13.0886   10.7392   -5.6909   -0.0000    3.8026   -4.4483   -2.6035
    4.3812    0.5308    6.8740    2.1732    0.0000   -1.4521   -2.8473   -0.1056
   -9.6918   -1.0102    1.9094   -1.7127   -0.0000    1.1444   -0.7909    0.2009
    1.5201    0.4610    4.6753    1.6411   -0.0000   -1.0966   -1.9366   -0.0917
   -5.0608   -0.4442    0.2022    2.0139   -0.0000   -1.3457   -0.0838    0.0883


(:,:,3) =

  182.4050  -53.5664  -34.0101   -1.7865    0.0000    1.1937   14.0874   10.6550
  -23.4300   12.3903    9.3127    3.8980   -0.0000   -2.6045   -3.8574   -2.4646
   26.5224   28.1024   20.3668    1.3447         0   -0.8985   -8.4362   -5.5899
  -36.5814  -30.6781  -19.9416    5.6070    0.0000   -3.7465    8.2601    6.1023
   -2.1302   -8.3843  -14.8704    3.1984    0.0000   -2.1371    6.1595    1.6677
   15.4721   10.8000    2.3631    0.8666         0   -0.5790   -0.9788   -2.1483
  -10.3807   -2.5048   -5.1640    0.2306         0   -0.1540    2.1390    0.4982
    0.4715    0.1200    2.3095    4.6578    0.0000   -3.1123   -0.9566   -0.0239
```

After Quantization:

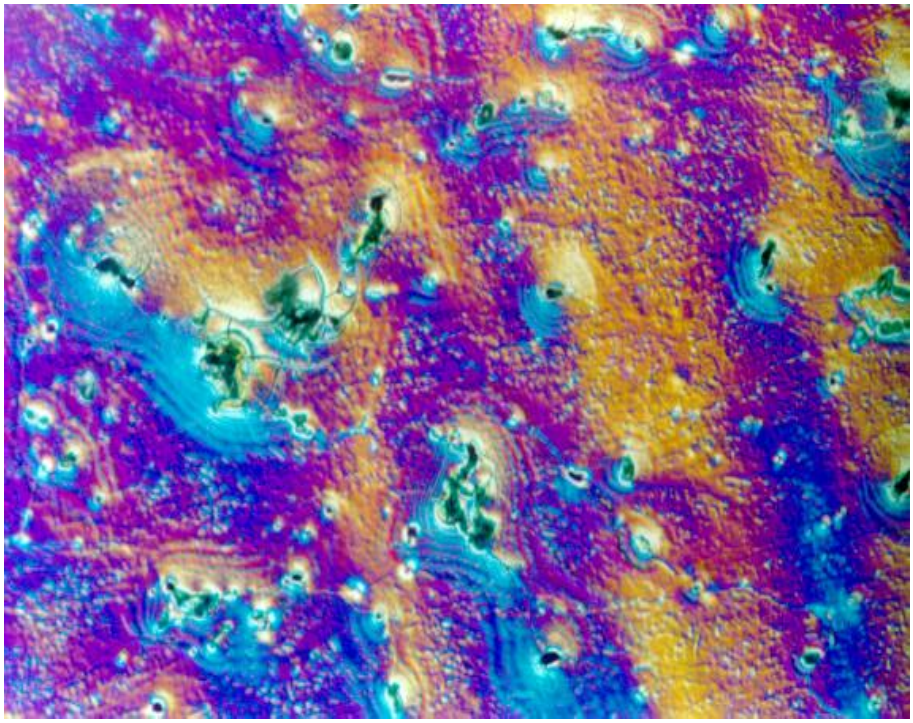
```
Quantized Image Output

(:,:,1) =

    41     1     1     1     0     0     0     0
     0     0    -1     0     1     0     0     0
     2    -2    -1     1     0     0     0     0
     1    -1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0


(:,:,2) =

    12     0    -1     0     0     0     0     0
     0     0     1     0     0     0     0     0
    -1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0


(:,:,3) =

     7    -2    -1     0     0     0     0     0
    -1     0     0     0     0     0     0     0
     1     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
```

- This process highlights the mathematical shape the images take based on the techniques shown. First, we convert them to YCbCr image format as that's the best for image processing, given that it can transfer very meaningful information over just the Y-channel, and we then manipulate the Cb- and Cr- channels by subsampling them, losing some information, but keeping the data visually the same.
- After this, we perform DCT on it using 8x8 blocks, given that the eye can't really see any significant changes over that span, to remove the AC components of the image. Then we quantize, based on a scaling factor that determines how lossy the compression should be. In the example above, we use qf = 30.

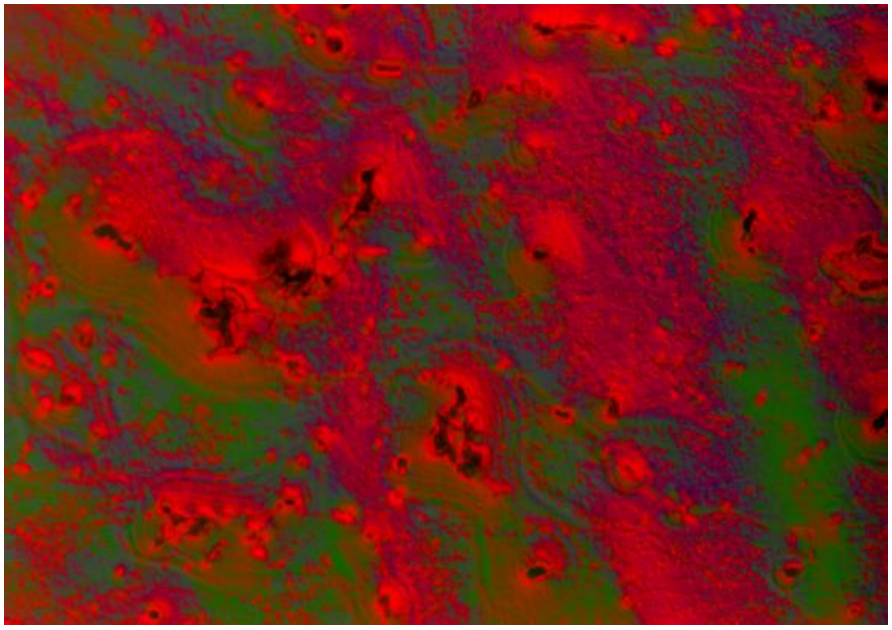
OUTPUT IMAGES AT EVERY STAGE OF EXECUTION

In the below pictures, I used “alu.png” and qf = 30 for our reference.

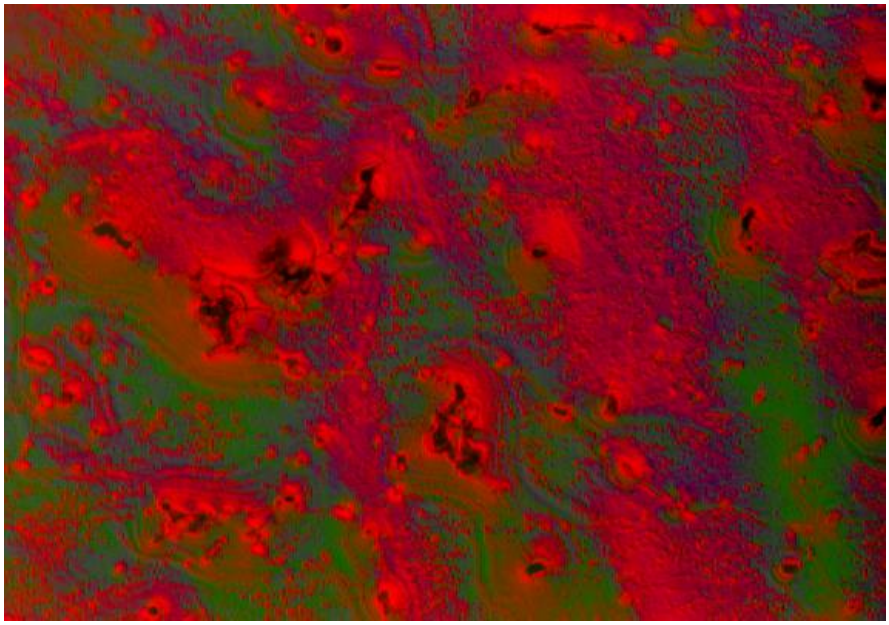
Original Image:



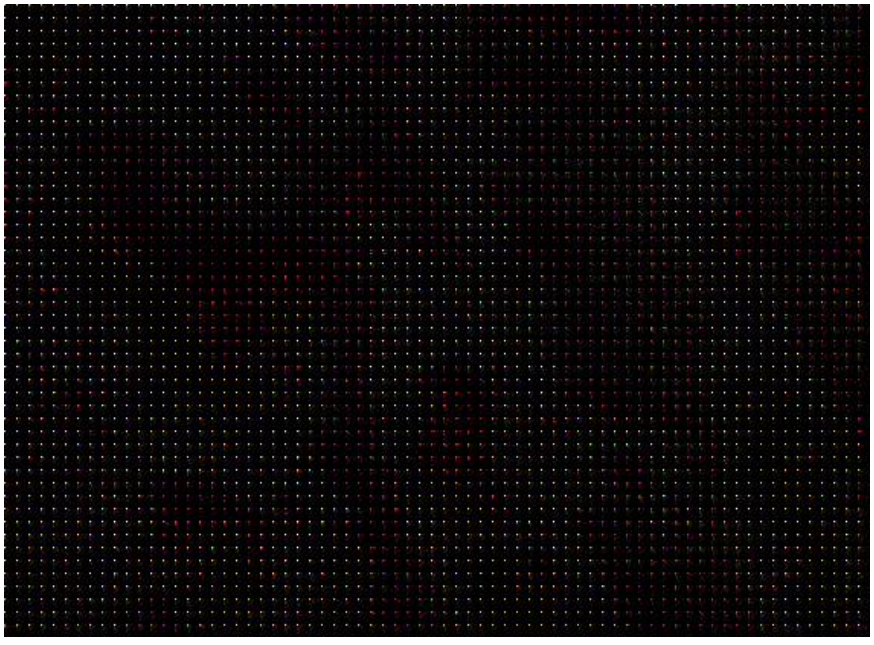
After Converting to YCbCr:



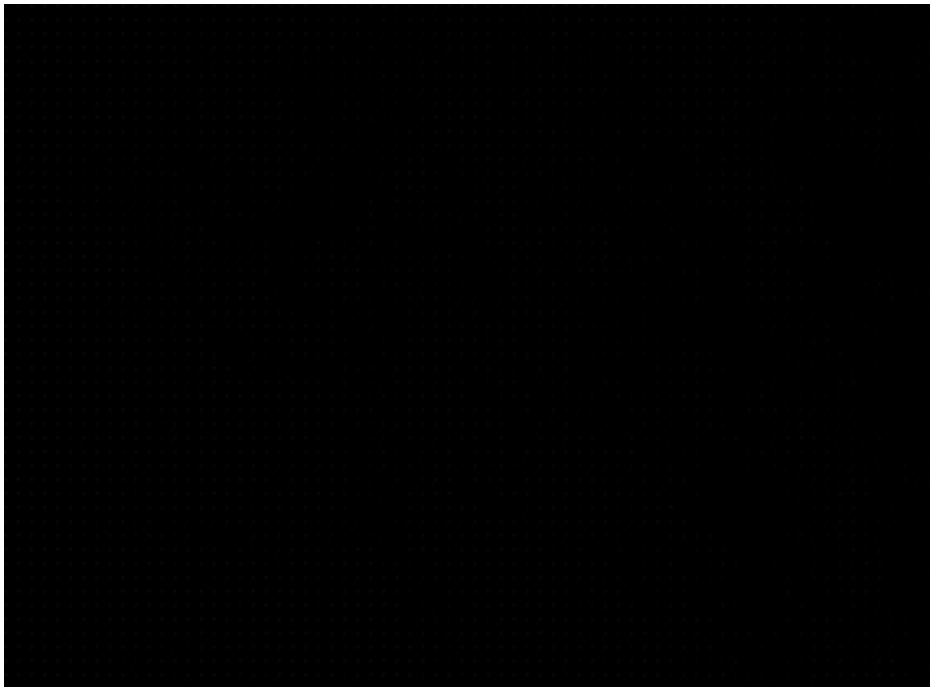
After 4:2:2 SubSampling:



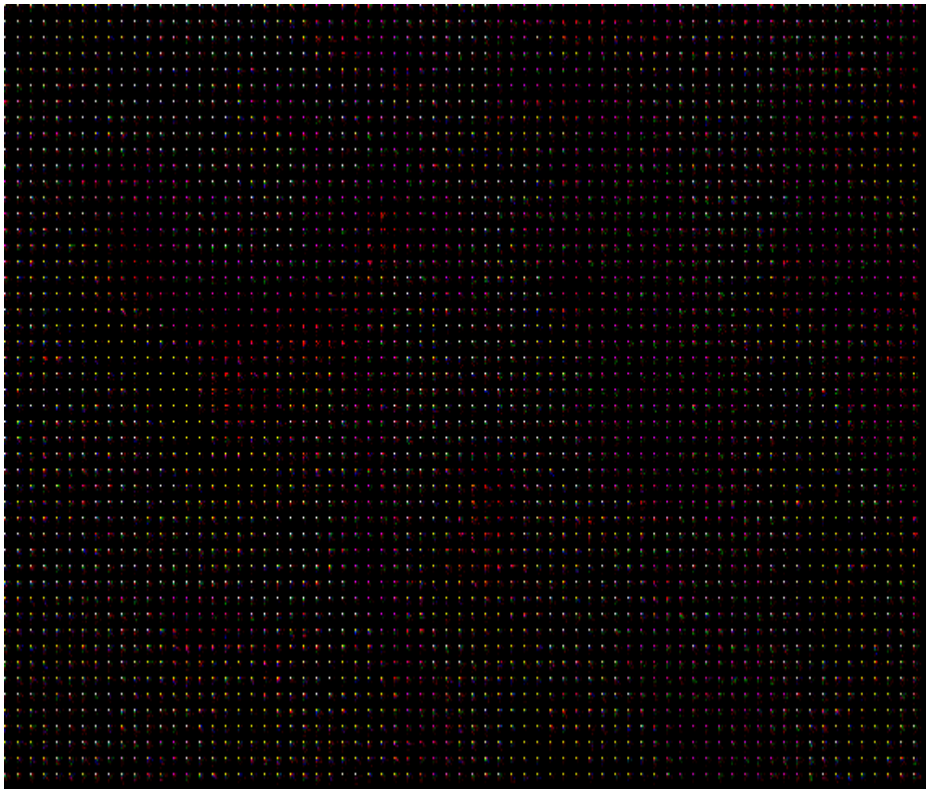
After 2D DCT:



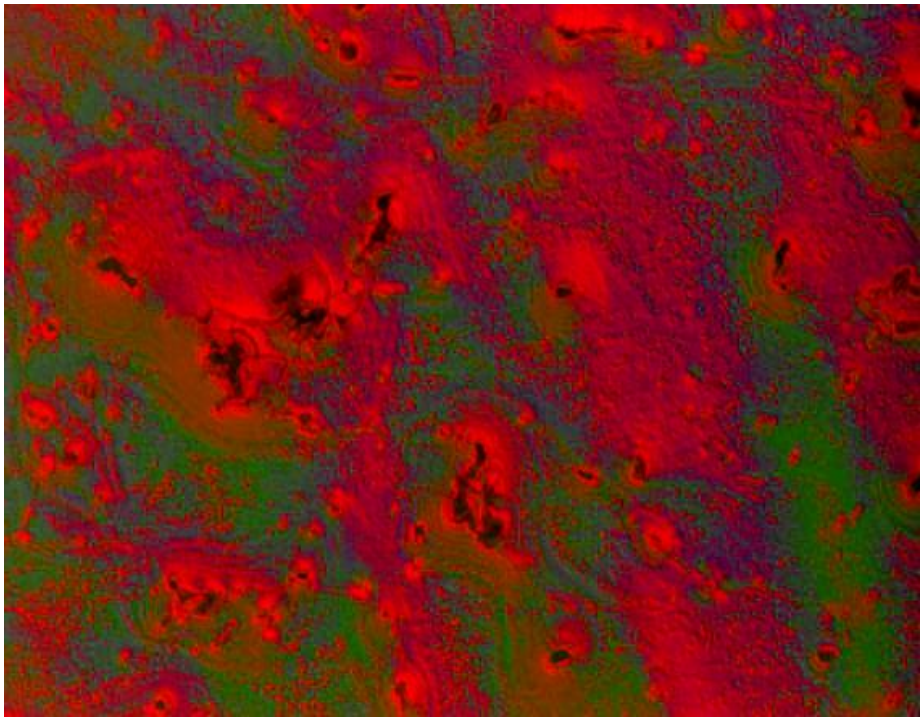
After Quantization:



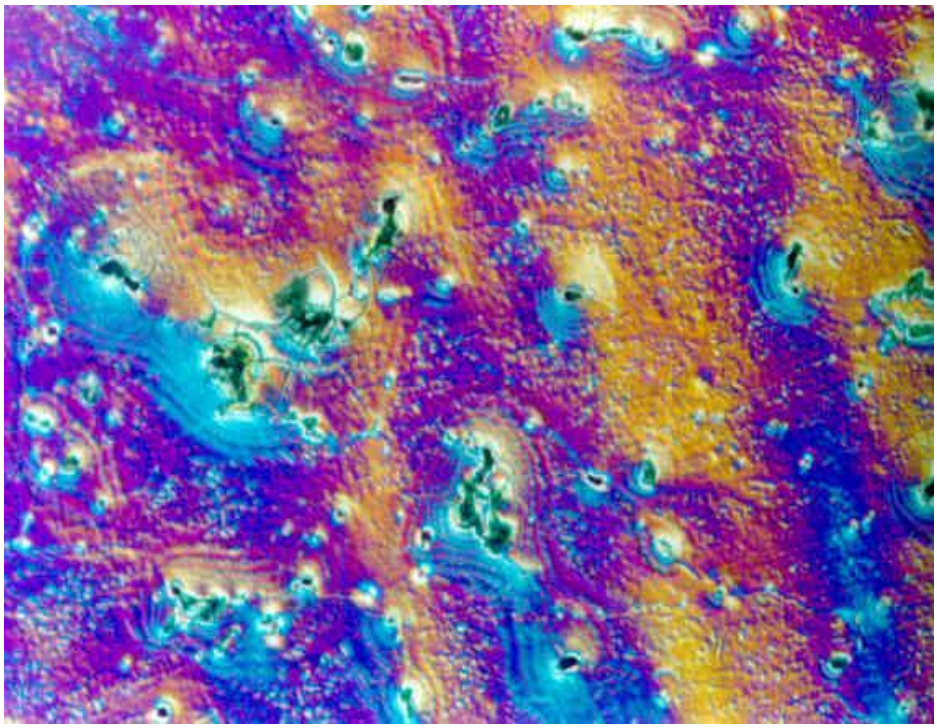
After Dequantization:



After 2D IDCT:



Final RGB Output:



- Just like with the pixel values, the whole image has gone through a lossy compression in which some of its data has been lost, but it remains the bulk of the information, such that interpretation of the image stays the same to eyes.
- It's very fascinating what happens when we quantize the 2D DCT coefficient, as it feels like all the information regarding the image is gone, but dequantizing shows us that it's just about the same to the naked eye, all the changes have occurred over tiny blocks of pixel values, such that it doesn't change the overall semantic of the image.
- Finally, it's worth noting that the lower the quality scale factor, the more lossy the compression. At qf = 100, it feels like the image even became clearer, or barely anything happened to it, while at qf = 3, the image looks blurred and very lossy. Below is an example of "pills.png" that went through the JPEG compression:

Qf = 3:

Qf = 100:

GUI SCREENSHOT