Computer Vision Homework 2

Part 1:
Q1.1 Filter Bank

There are 4 classes of filter. Each one of these filters pick up different properties. Each one of these can be run at 5 different scales, and across 3 channels . Here they are as well as their respective properties that they pick up from the image:

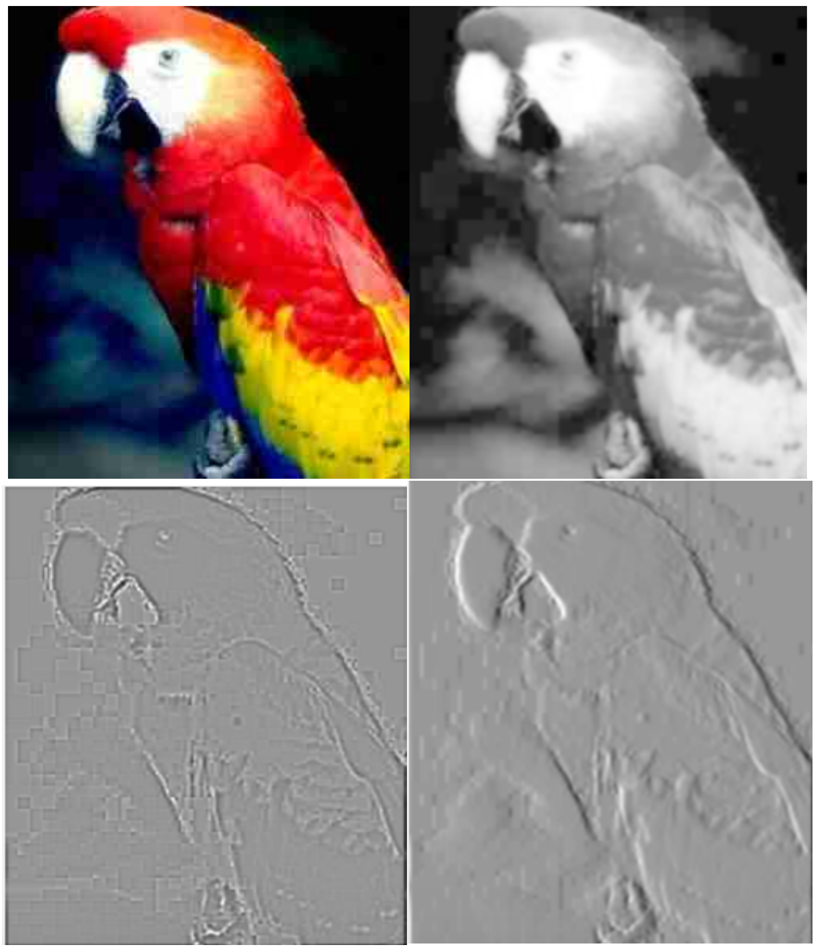
- Gaussian Scale
    - De-noising by removal of high frequency information in the image. The result of this de-noising is a blur which gets rid of noise but also preserves some information which is highly suitable for Gaussian pyramids, where coherent resolution is kept as close to default despite compression of an image.

- Log Scale
    - Refers to the Laplacian transform of the image. This Laplacian filter comes about from the difference between a unit filter and a Gaussian to show the high frequency information that is lost post Gaussian blurring.

- dX (Derrivative wrt X)
    - Reacts well to vertical lines because horizontal gradient distribution can precisely identify vertical lines. Is a derivative of the image with respect to the x-direction.

- dY (Derrivative wrt Y)
    - Reacts well to horizontal lines because vertical gradient distribution can precisely identify horizontal lines. Is a derivative of the image with respect to the y-direction.

Q1.2
I love parrots and I picked one for this test ☺ Here it is originally in the top left, along with a Gaussian filter transformation in the top right, a Laplace Transform in the Bottom Left, and a dX transformation in the bottom right! These are 3 of the 4 filter responses (omitted dY for no interesting reason) :



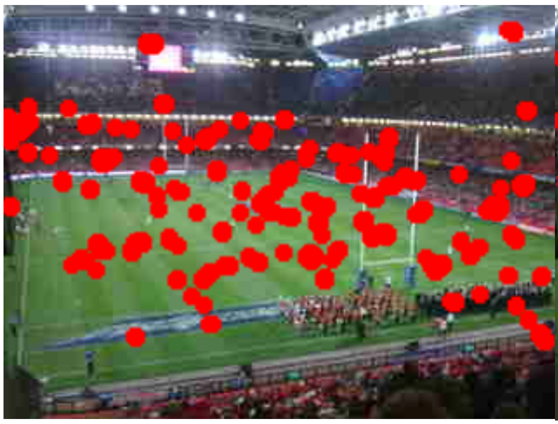
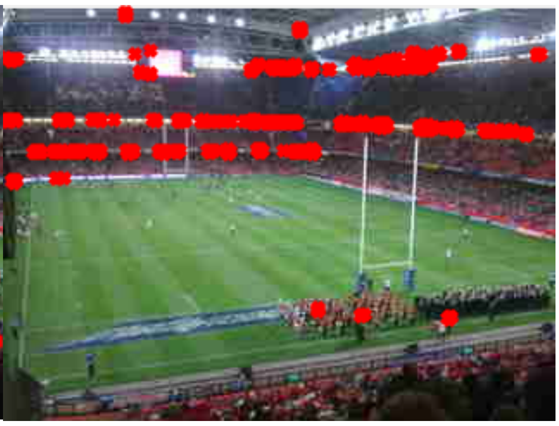
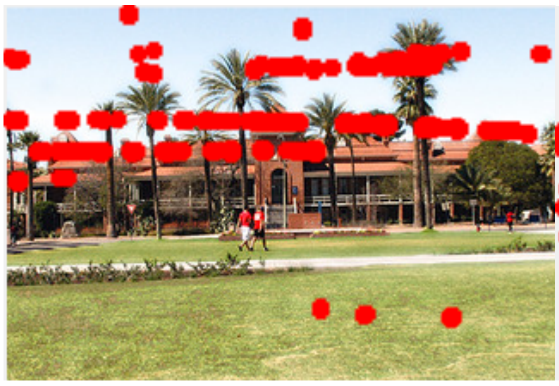
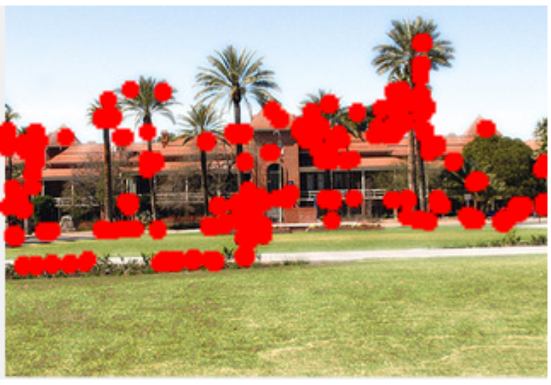
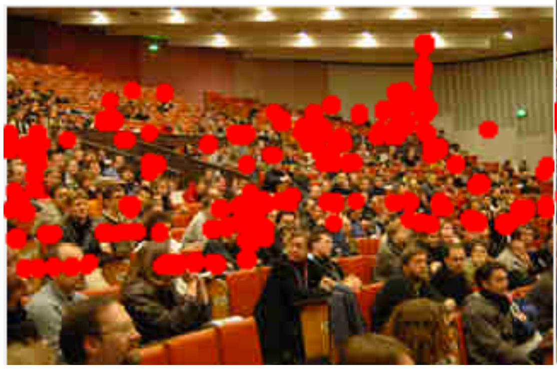
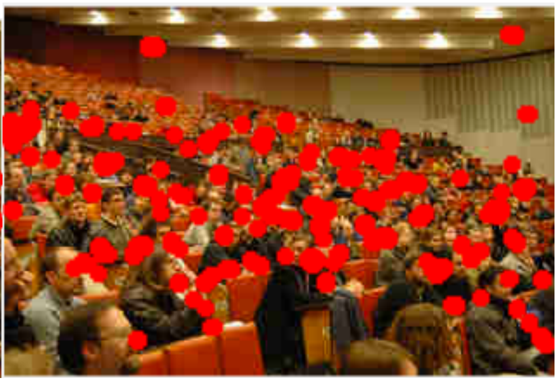
Artefacts that appear in these filtered images are described below:

Gaussian: Smoothed edges, loss of high frequency detail in the image (it seems detail is less granular post Gaussian transform)
Laplace (log): Demarcation of edges across multiple hierarchies, with greater prominence for the contour of the bird rather than higher hierarchical contours like those of the wing
dX: Edge imprint with high preponderance of vertical or quasi vertical edges. The deeper the edges, the more vertical they are etc…

The CIE Lab colour space is different from what I am use to on OpenCV as the channels are not distinct R, G, B channels but rather 2 of those channels are a blue → yellow and green → red, and the third one is a lightness channel (how bright the light is in the image or how dim etc). It is useful because it is device independent, which means that image processing should yield the same output for all devices irrespective of how colour is defined on one machine. It is almost a version control for how images are defined by colour, ensuring there is synergy and alignment on all devices that process those images.

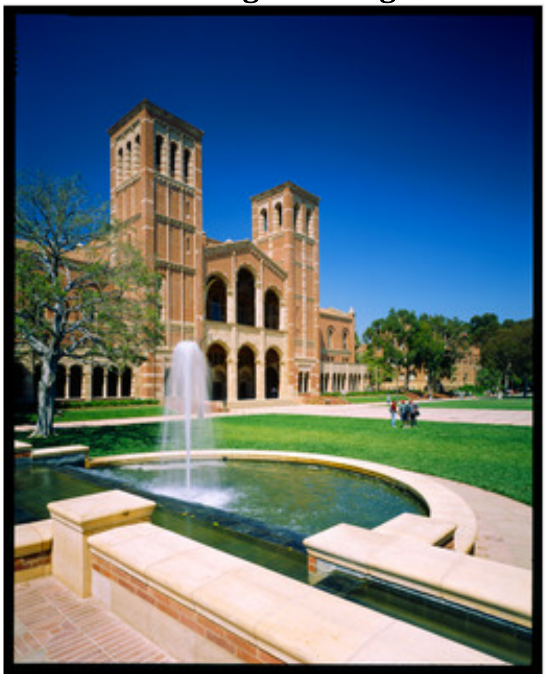
Q1.3

The left side represents the images with getRandom(), the right represents the images with getHarrisPoints(). The parameters for alpha and k were 500 and 0.05 respectively.

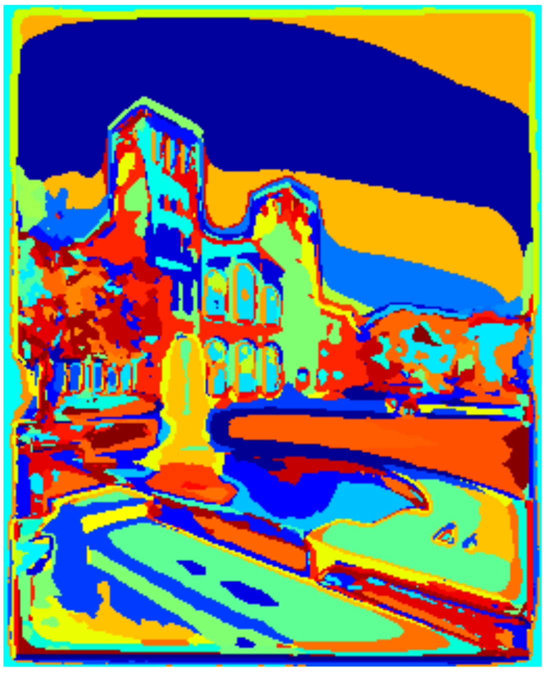







Q2.1
The two classes are campus and landscape. Here is the first image belonging to the campus class.

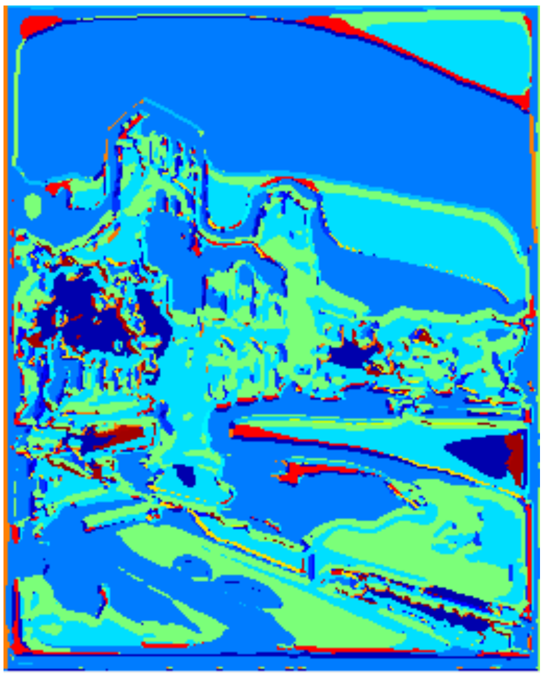
Original Image



Random Points

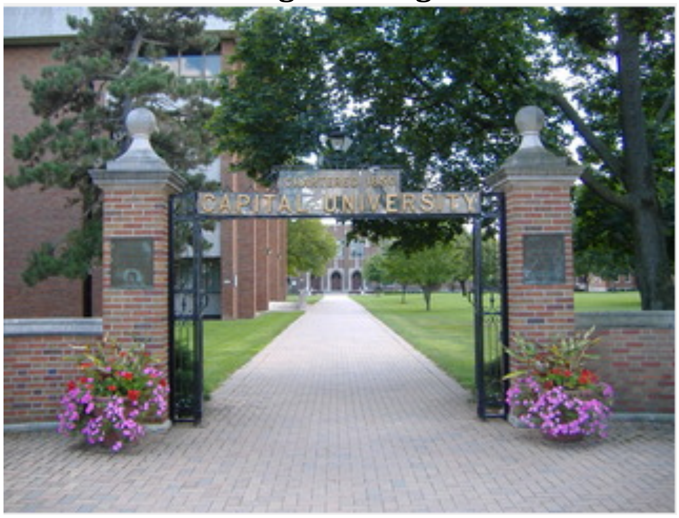


Harris Points

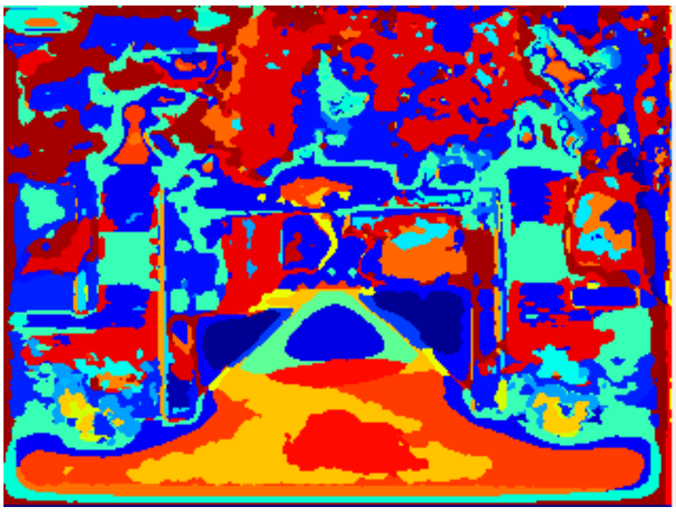


Here is the second image belonging to the campus class.

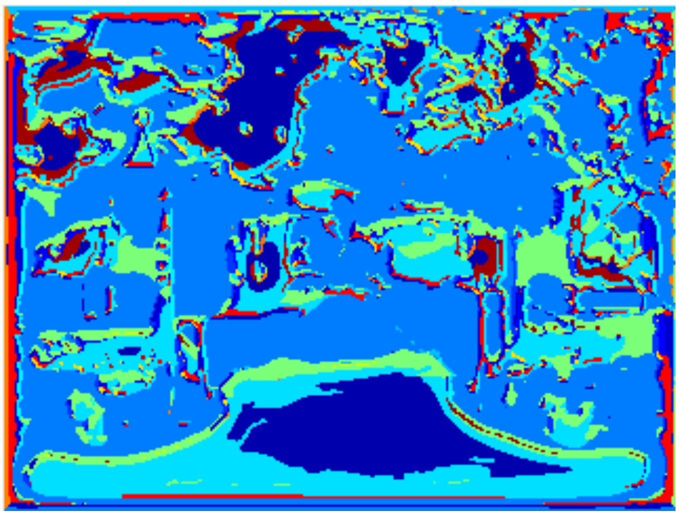
Original Image



Random Points

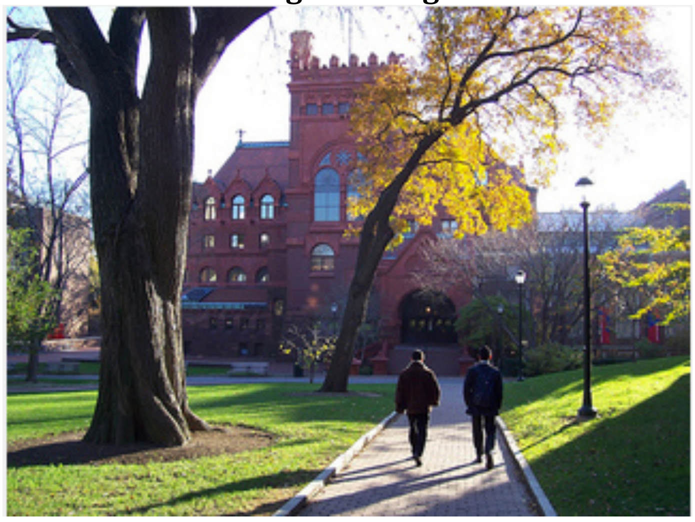


Harris Points

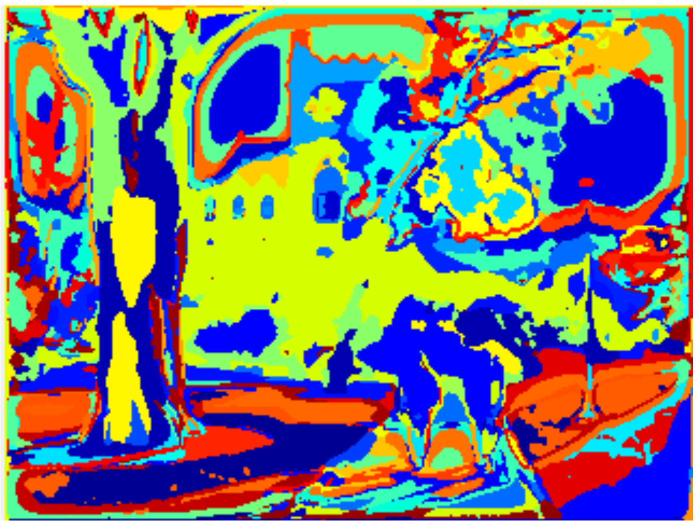


Here is the third image belonging to the campus class.

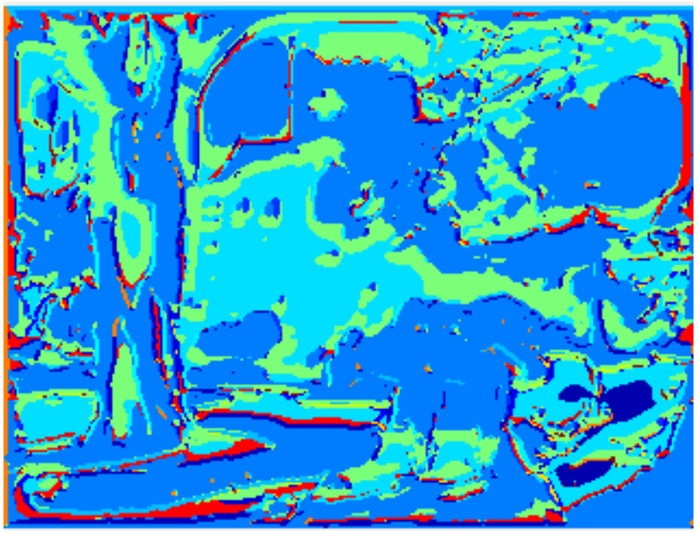
Original Image



Random Points

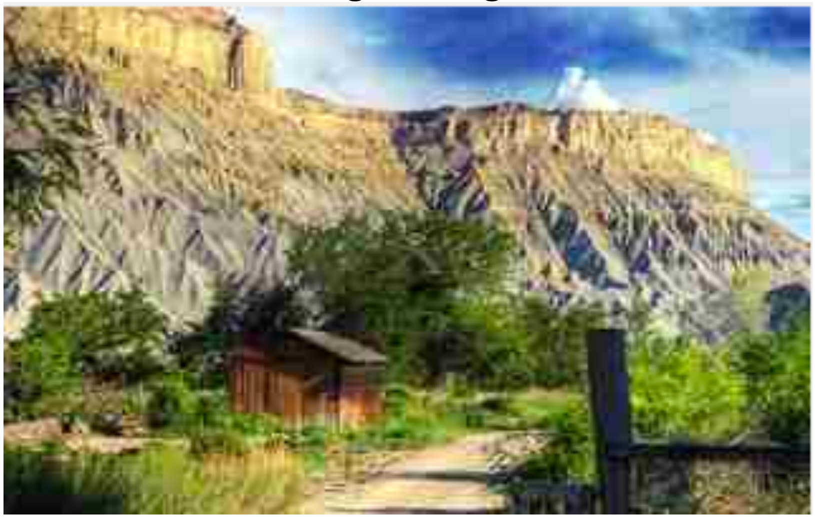


Harris Points

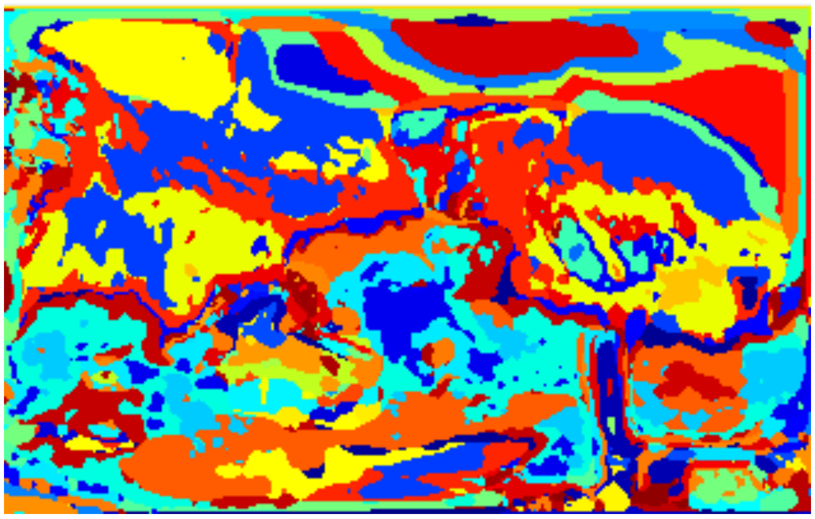


Here is the first image belonging to the landscape class.

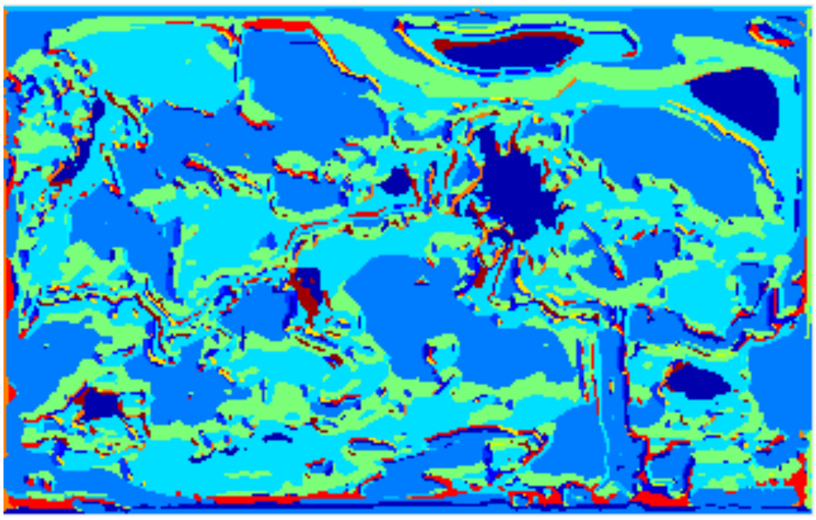
Original Image



Random Points



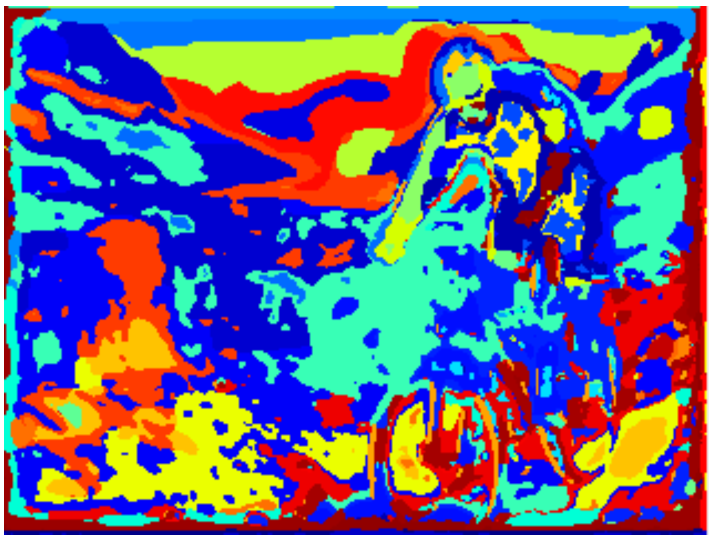
Harris Points



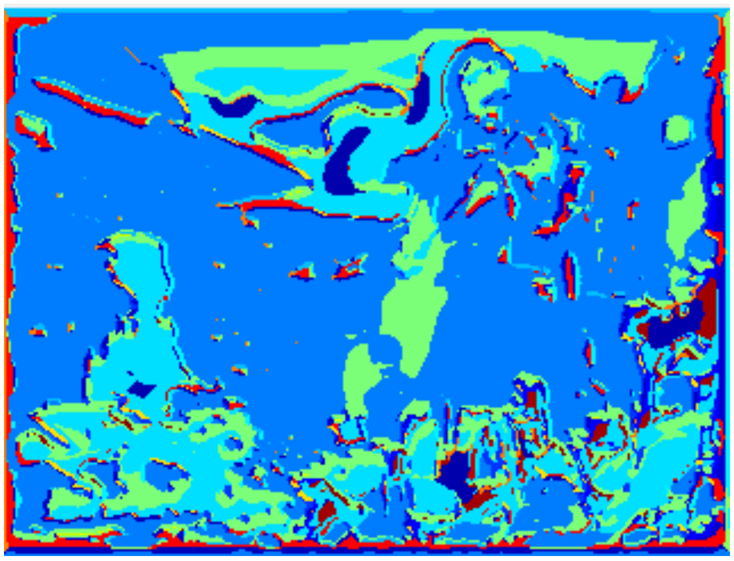
Here is the second image belonging to the landscape class.

Original Image

Random Points



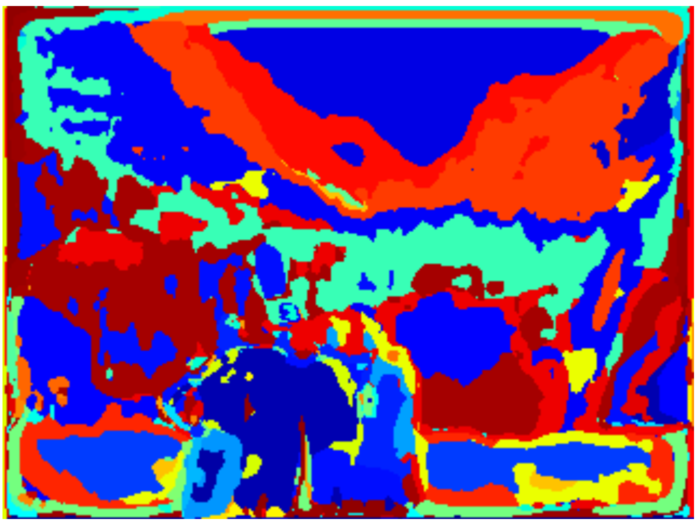
Harris Points



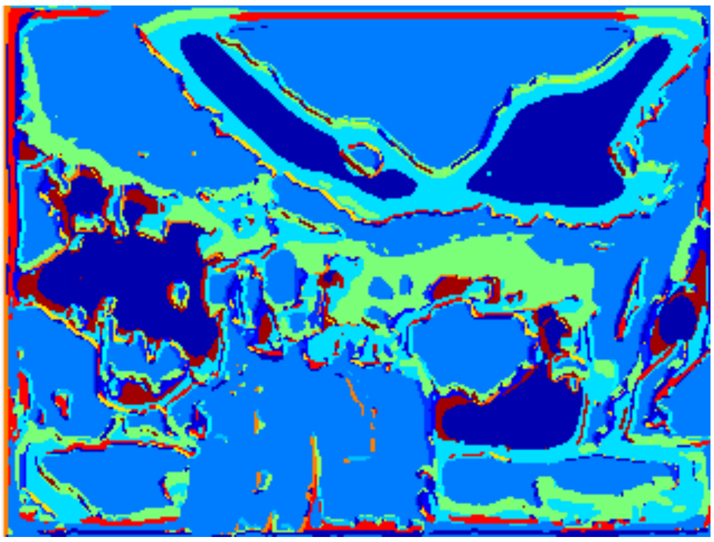
Here is the third image belonging to the landscape class.

Original Image

Random Points



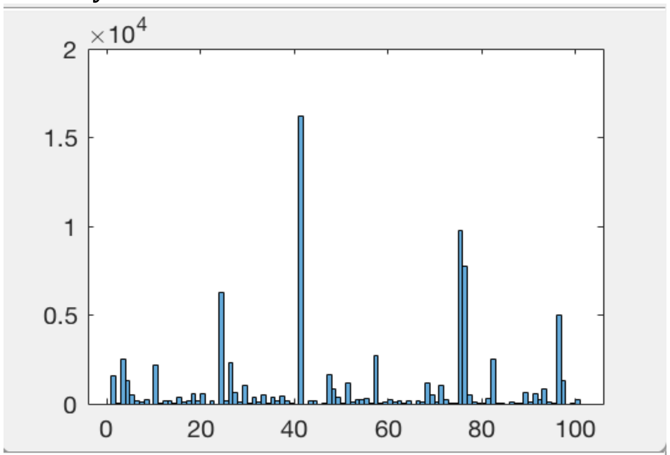
Harris Points

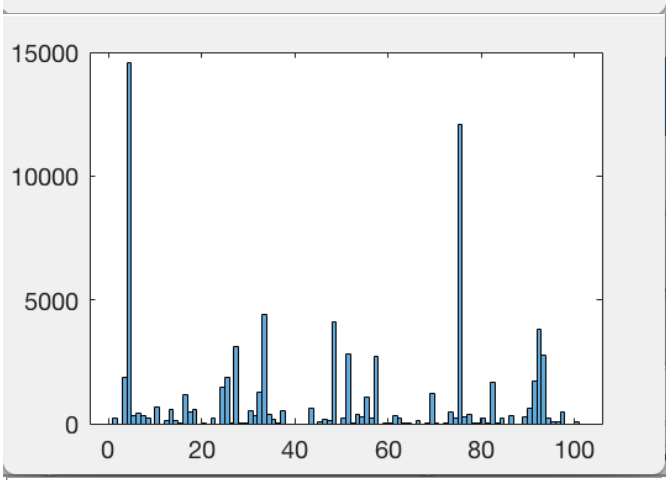


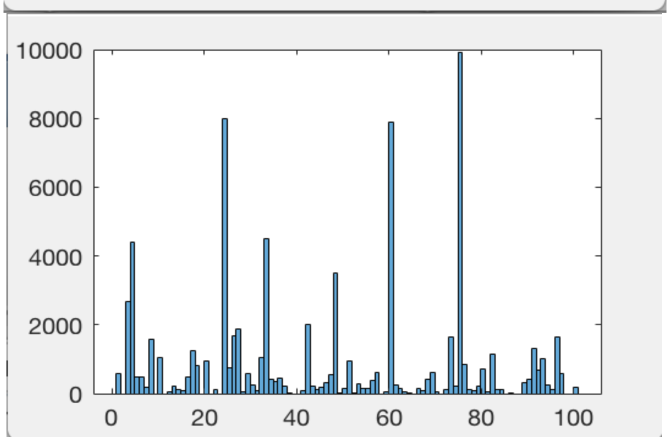
It looks like the harris point based method of sampling points did not provide the intented visual word map. The Harris Point based method seems to be capturing the semantic meaning less well than the random point based method. In reality it should perform better because features are detected which yield higher information gains, as these features are usually corners or vertexes, which are situated in the semantic demarcations of the image. Random points, samples randomly along the image, and in many cases, a lot of points may be located in the middle of surfaces that themselves yield nothing much as far as disambiguating semantic meaning of the image.

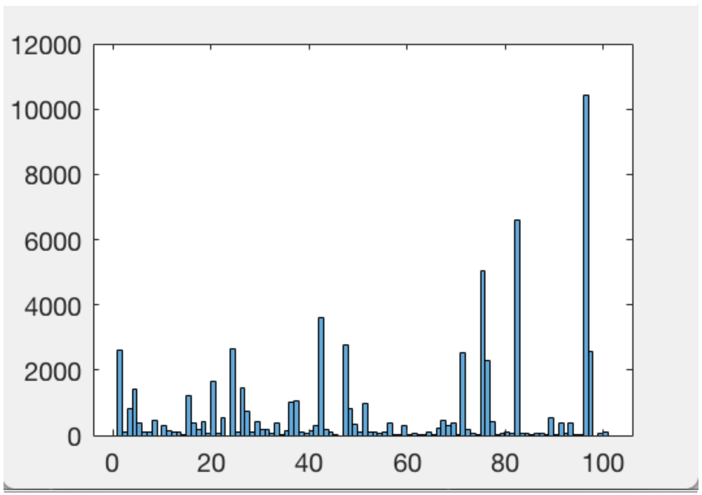
Q2.2

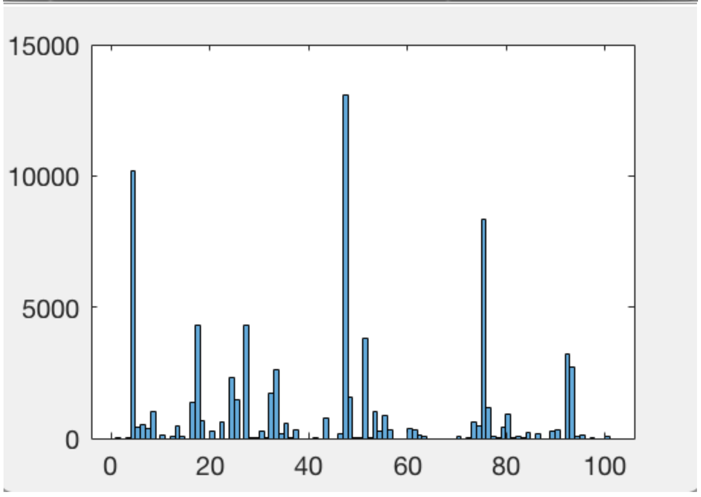
These 6 histograms belong to the 6 images from question 2.1, but based on Harris points dictionary.

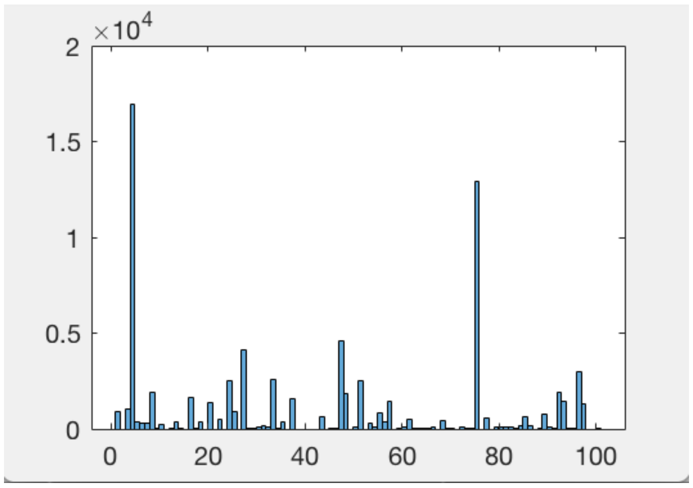








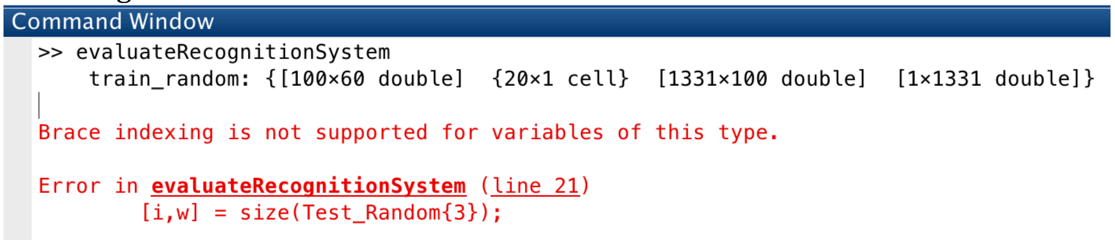




<u>Q2.3</u>
I would just like to clarify that the output of this code has been transformed to 4 separate structs that represent the training, testing, for random and harris each distinctly. I do not know how I should combine all of these into 2 separate containers, were we suppose to concatenate the elements for each element in the struct as we go from train to test (so as to merge these together)? Well I just stuck with outputting 4 separate structs. Hope that is ok.

Q3.2

I have been unable to access any of the data in my struct, which is passed on to this evaluateRecognitionSystem.m. When displaying these structs they read the following:



```
>> evaluateRecognitionSystem
    train_random: {[100×60 double]  {20×1 cell}  [1331×100 double]  [1×1331 double]}

Brace indexing is not supported for variables of this type.

Error in evaluateRecognitionSystem (line 21)
        [i,w] = size(Test_Random{3});
```

I have not been able to get around this, and was left with not enough time to complete this. However, I have written the rest of the code, as if we bypassed this issue to produce a histogram.