Pages / William Collins's Home / CS8674 - Masters Project - Investigation of RAW Log RGB Space for Color Constancy

Histogram Based Models

Created by Unknown User (barrelchester), last modified on Apr 29, 2023

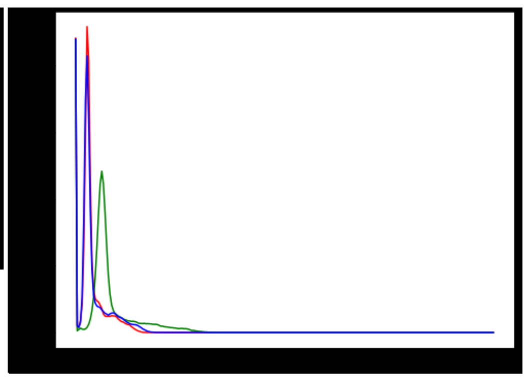
This page details experiments with histogram based models, comparing linear and log histograms. Several types of histograms were used; 1D RGB histograms, 2D histograms of projected 3D RGB points, and raw 3D histograms (in progress for ongoing work).

1D Histograms

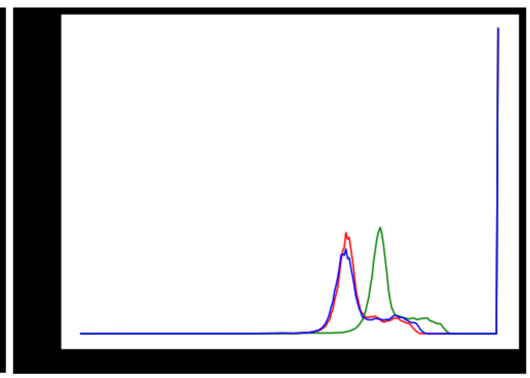
The range of the linear 1D histograms is (0,1) and of the log histograms is (-9.5705, 0). These histograms contain RGB data in isolation from each other, so are not informative enough to encode the 3D structures of the RGB color space and yield strong results.

Example

Without illuminant applied:



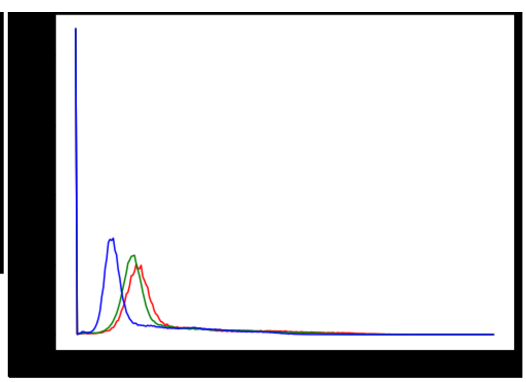
linear histogram



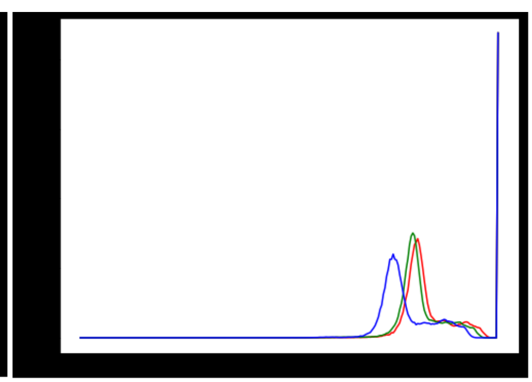
log histogram

With illuminant applied:

with illuminant applied



linear histogram



log histogram

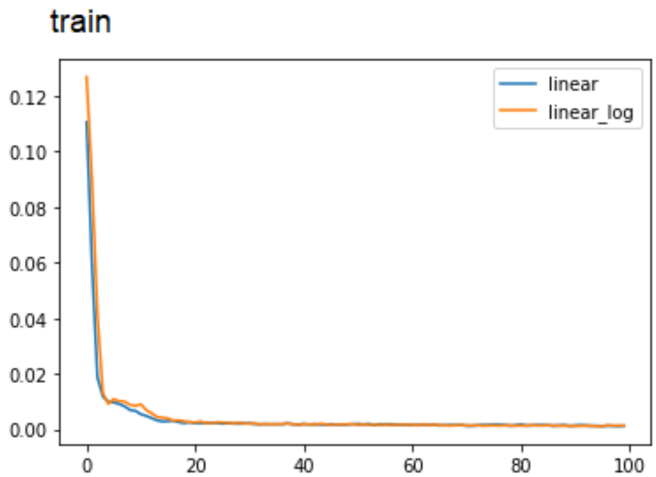
Simple Linear Model

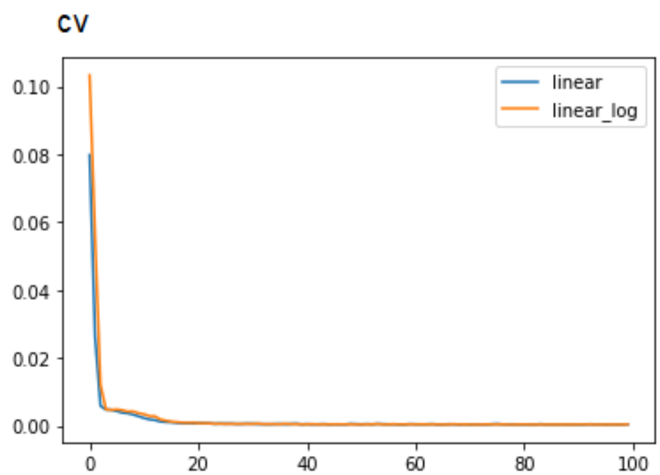
A simple linear model was trained on the concatenated 256 bin 1D RGB histograms.

```
LinearHistogramModel(
  (flat): Flatten(start_dim=1, end_dim=-1)
  (fc1): Linear(in_features=768, out_features=256, bias=True)
  (fc2): Linear(in_features=256, out_features=128, bias=True)
  (fc3): Linear(in_features=128, out_features=3, bias=True)
  (dropout): Dropout(p=0.25, inplace=False)
)
```

Training & CV Curves for 100 Epochs

Plots of training and CV curves for linear and log.





As can be seen, both linear and log performed similarly, but did not score well on the test set.

Results

For comparison, good models score around Mean 1.5.

linear Mean: 9.902493, Median: 5.893889, TriMean: 9.952528, Best25: 0.650706, Worst25: 25.636860
log Mean: 9.881635, Median: 5.453310, TriMean: 9.798018, Best25: 0.607381, Worst25: 25.768649

Histograms of 3D RGB Points Projected to 2D

To better capture the structures in the linear and log RGB space, i.e. BIDR cylinders, the 3D RGB points for each image were projected onto several randomly oriented 2D surfaces. Histograms were then taken of the resulting 2D pixels and passed to a relatively simple convolutional neural network.

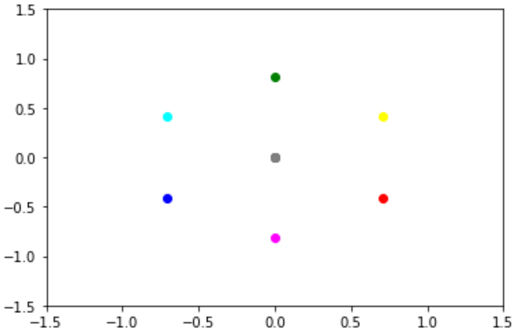
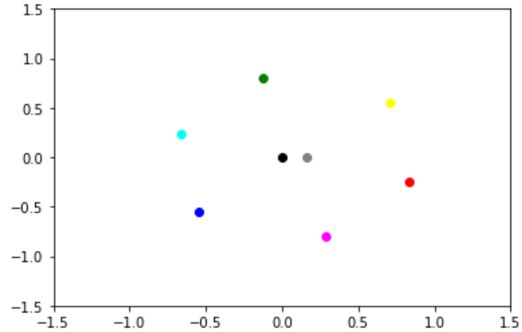
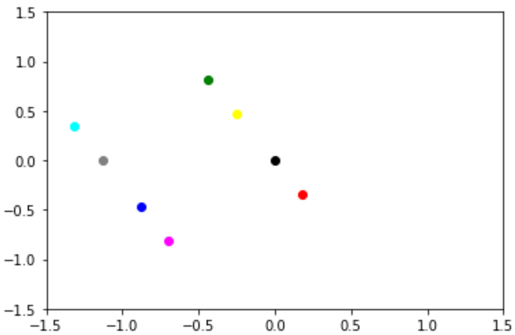
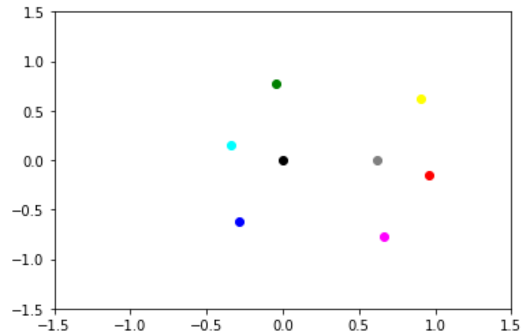
Example Set of RGB Point Projections

The following RGB pixels (scaled to 0..1) describe the vertices of a cube in color space:

[0,0,0],[1,0,0],[0,1,0],[0,0,1],[1,1,0],[1,0,1],[0,1,1],[1,1,1]

being the colors ['black', 'red', 'green', 'blue', 'yellow', 'magenta', 'cyan', 'white'] respectively.

Below are 4 random projections into 2D space (white colored gray for visibility):

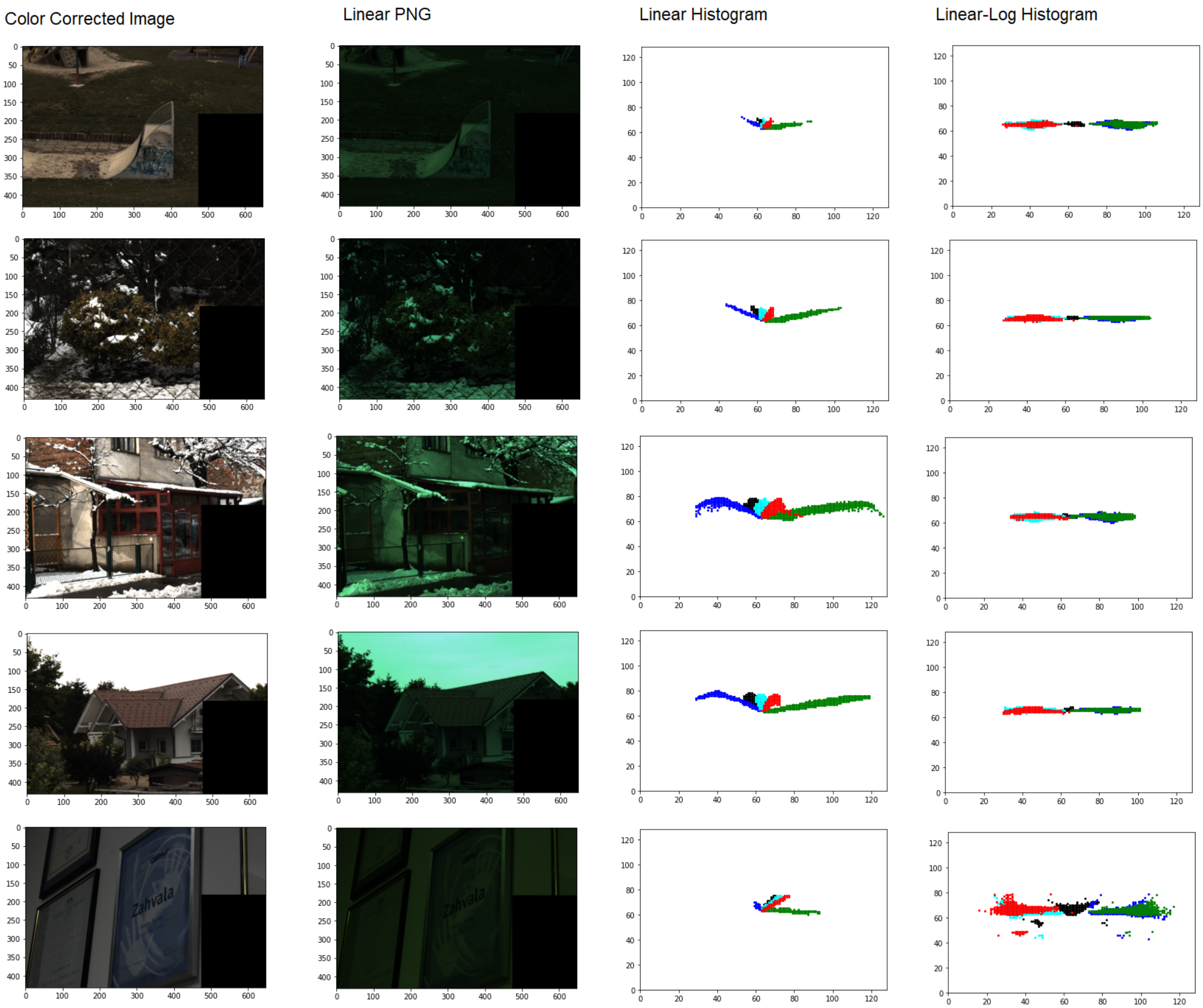





Resulting 2D Histograms for Linear and Log Versions of Several Images

Below are several sample images and their resulting 2D histograms. The linear PNG images are the data used to derive the histograms, the color corrected images are just shown for comparison. The "linear-log" version of the data first is linearized, and then log is applied.

The 5 different colors in the histogram are the 5 different projections of the same image (5 chosen arbitrarily as enough to capture the 3D structure). These 5 projection planes were

128 bins were used. The histograms tend to be very sparse, with 0.18 % bins filled for some examples. The projected points ranged from -1.5 to 1.5, and the linear-log points ranged from -15 to 15. These ranges were used to fit the points to the histograms properly.



Simple CNN Model

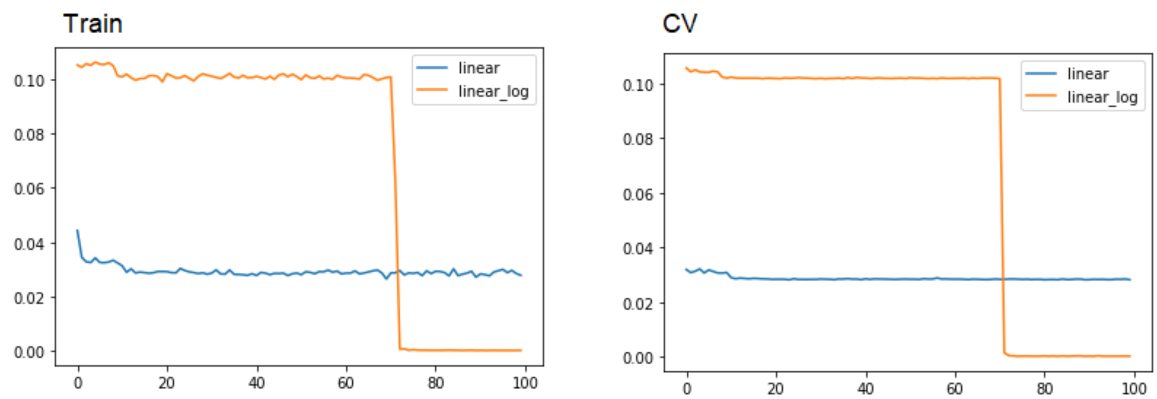
The set of 5 projections form the channels of the 128 x 128 histograms, giving shape (5,128,128) for each image.

```
CNNHistogramModel(
  (do): Dropout(p=0.25, inplace=False)
  (conv1): Conv2d(5, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv2): Conv2d(32, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (conv3): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (conv4): Conv2d(128, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (conv5): Conv2d(64, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (conv6): Conv2d(32, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
  (flat): Flatten(start_dim=1, end_dim=-1)
  (fc1): Linear(in_features=256, out_features=128, bias=True)
  (fc2): Linear(in_features=128, out_features=3, bias=True)
)
```

Training & CV Curves for 100 Epochs

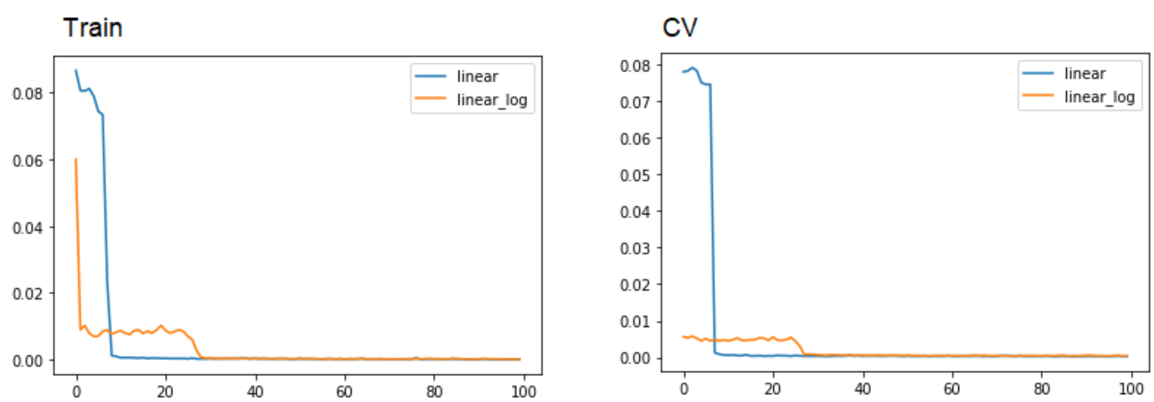
The training sessions vary in unexpected ways. In general, a model will abruptly converge at some random epoch or it will fail to learn and get very bad scores on the test set.

First run



Linear: Mean: 31.008539, Median: 32.724630, TriMean: 31.821227, Best25: 22.448203, Worst25: 36.179763
Log: Mean: 1.857605, Median: 1.132198, TriMean: 1.631050, Best25: 0.380400, Worst25: 4.605093

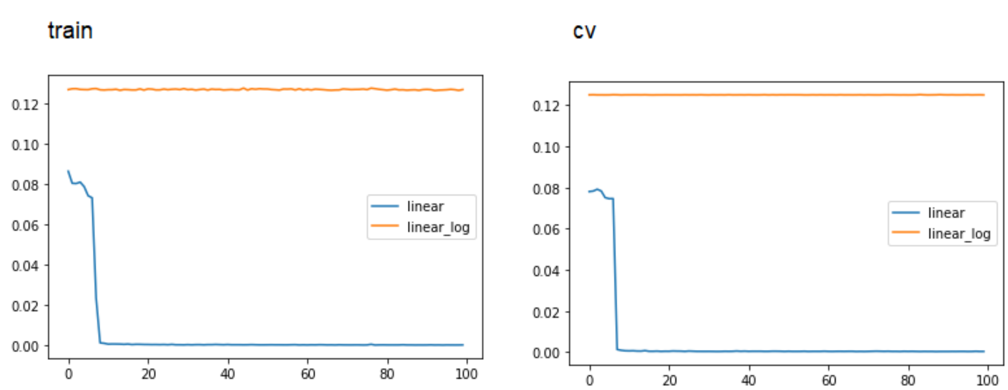
Second run



Linear: Mean: 1.672695, Median: 0.986252, TriMean: 1.456510, Best25: 0.329702, Worst25: 4.238528

Log: Mean: 1.921946, Median: 1.290510, TriMean: 1.671188, Best25: 0.627581, Worst25: 4.369729

Third Run



This time, the network failed to train properly and resulted in NaN values for the test metrics.

Log: Mean: nan, Median: nan, TriMean: nan, Best25: nan, Worst25: nan

Additional runs (including a 200 ep run with learning rate 1e-5) exhibited the same sensitive behavior.

Raw 3D Histograms - Ongoing Research

In the hopes that the 3D BIDR structures would be captured most directly by a 3D histogram, I will try this next as part of ongoing research.

No labels