CED19I039_DIP_assg1

January 14, 2022

1 CS5102 - DIP assignment 1

Author: Yash Kumar Sahu, CED19I039

1.1 Part1 - RMSE plotting using Color Images

[1]:
```
#importing necessary libraries
import glob
import math
import cv2
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"]=10,10
import matplotlib.image as mpimg
```

[2]:
```
#Input list of images from path
images = [cv2.imread(file) for file in glob.glob("dip1_dataset/y*.jpg")]
images=np.array(images)
## 3 channel images, colored
print(images.shape)
```

```
(100, 480, 640, 3)
```

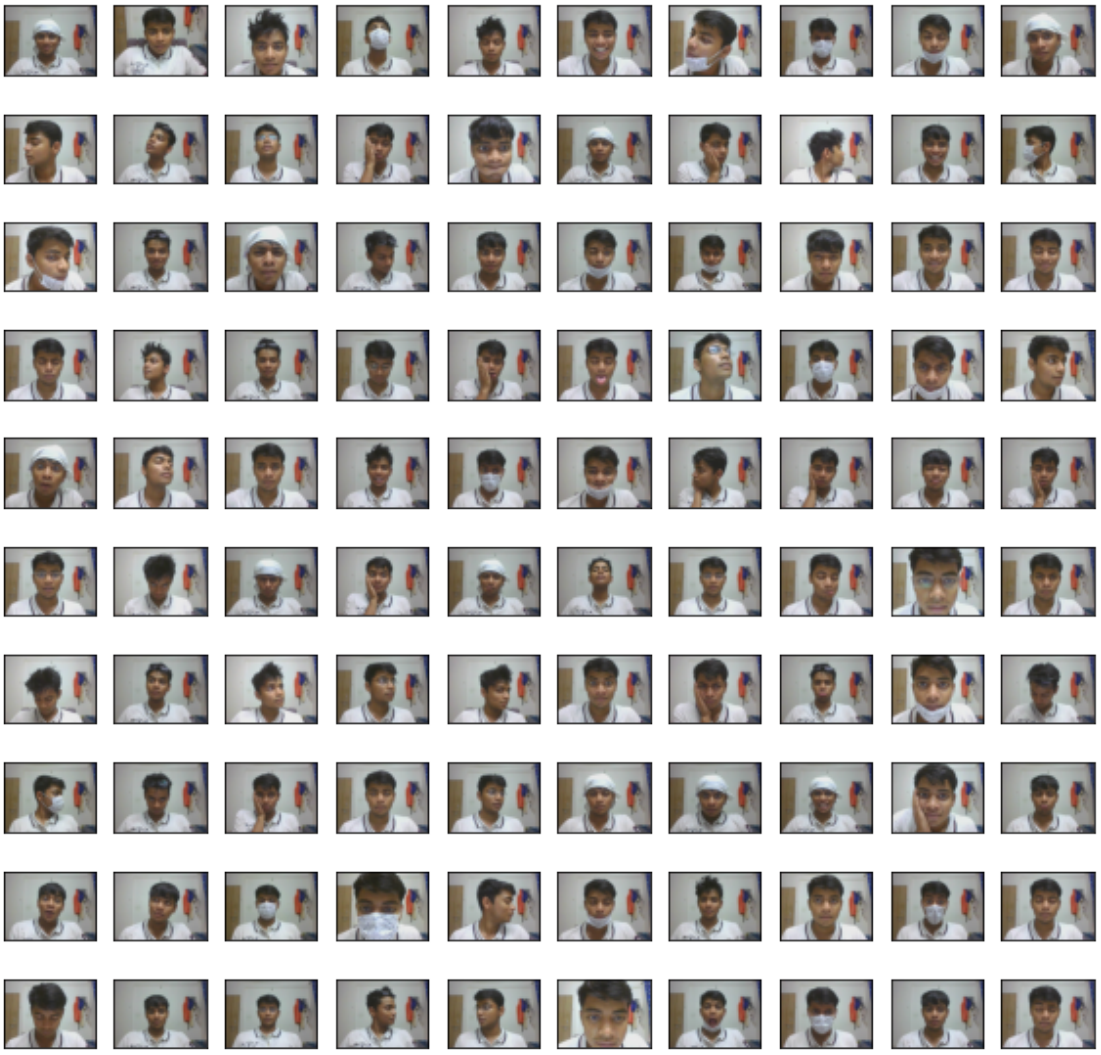
[3]:
```
#List containg image numbers
img_lst = list(range(1,len(images)))
img_lst=np.array(img_lst)
img_lst
```

[3]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
       52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68,
       69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85,
       86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99])
```

[4]:
```
#Verifying no. of input images
l=len(images)
l
```

[4]: 100

[5]:
```
#Ploting all input images
for i in range(l):
    plt.subplot(10,10,i+1),plt.imshow(cv2.cvtColor(images[i], cv2.
 ↪COLOR_BGR2RGB))
    #plt.title(i)
    plt.xticks([]),plt.yticks([])
plt.show()
print(l,"- Images")
```



100 - Images

[6]:
```
#Considering 1st input image as reference
ref_image=images[0];
```

[7]:
```
#Calculating rmse score for each image w.r.t reference image
rmse_score=[]
for i in range(1,l):
    rmse=math.sqrt(np.square(np.subtract(ref_image,images[i])).mean())
    rmse_score.append(rmse)
rmse_score=np.array(rmse_score)
print(rmse_score)
```

```
[ 9.73822985 10.35534293  7.05708087 10.43357149 10.22915171 10.41106584
  7.93780409 10.33394277  8.97321922 10.4617712   7.60411465  9.26335829
  8.53283022 10.56354598  4.93392605 10.41268221 10.74443155  8.84111305
  7.21005173 10.7008016   7.41627208 10.18785813  9.41603341  8.82139862
 10.33054277  6.37946331 10.45247984 10.1781089  10.2814768  10.28340651
 10.58158624  7.54002478  9.00751453  8.54768773 10.29213483 10.31643301
  8.55170923 10.42192888 10.25571878  9.02309398 10.32209502 10.2204048
  7.76149203  7.95337064 10.41341637 10.40496562  8.49991492 10.44819255
  8.49482577 10.31917718 10.3400765   5.00855226  9.22500018  4.93073134
  7.33448011 10.27275385 10.2963052  10.42650561 10.24486607 10.47692529
  7.13464807 10.73247577  9.07304529 10.34741879 10.20857419 10.22154497
  7.5764613  10.5033935  10.32456418  7.09777977  7.46296637  9.12541333
 10.25911281  9.17829045  8.75074922  7.55838667  4.90841325 10.29902155
  8.76641287  8.98221441 10.40353673  6.39112417 10.22988456 10.34063009
 10.44657862  8.28993864 10.24111711  7.91064704 10.29704872 10.42201613
  7.26229187  7.6780164   7.6932599   9.11874744 10.43582    7.71047176
 10.52822982  8.90333194 10.26870647]
```

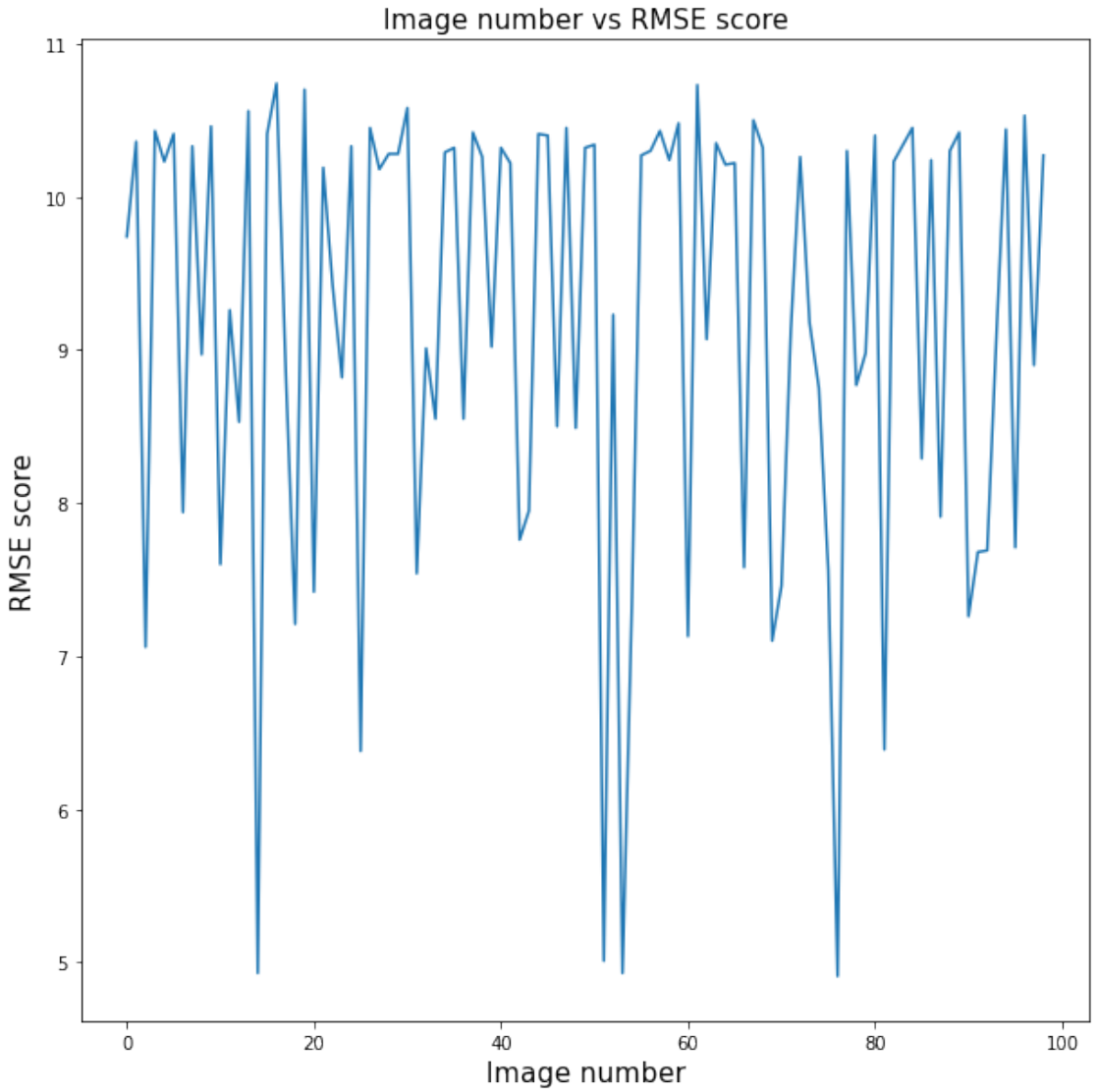
[8]:
```
#Rounding off values to two decimal places
rmse_score=np.round_(rmse_score, decimals=2)
print(rmse_score)
```

```
[ 9.74 10.36  7.06 10.43 10.23 10.41  7.94 10.33  8.97 10.46  7.6   9.26
  8.53 10.56  4.93 10.41 10.74  8.84  7.21 10.7   7.42 10.19  9.42  8.82
 10.33  6.38 10.45 10.18 10.28 10.28 10.58  7.54  9.01  8.55 10.29 10.32
  8.55 10.42 10.26  9.02 10.32 10.22  7.76  7.95 10.41 10.4   8.5  10.45
  8.49 10.32 10.34  5.01  9.23  4.93  7.33 10.27 10.3  10.43 10.24 10.48
  7.13 10.73  9.07 10.35 10.21 10.22  7.58 10.5  10.32  7.1   7.46  9.13
 10.26  9.18  8.75  7.56  4.91 10.3   8.77  8.98 10.4   6.39 10.23 10.34
 10.45  8.29 10.24  7.91 10.3  10.42  7.26  7.68  7.69  9.12 10.44  7.71
 10.53  8.9  10.27]
```

[9]:
```
# Plot the values in the graph with the indexed images in the X-axis and RMSE␣
 ↪score in the Y-axis.
plt.title("Image number vs RMSE score", fontsize=15)
plt.xlabel("Image number",fontsize=15)
```

```
plt.ylabel("RMSE score",fontsize=15)
plt.plot(rmse_score)
"""
plt.plot(img_lst,rmse_score)

##forcing integer x-axis
x=img_lst
new_list = range(math.floor(min(x)), math.ceil(max(x))+1)
plt.xticks(new_list)
"""
plt.show()
```



1.2 Part2 - RMSE plotting using Grayscale Images

[10]:
```
#Converting all images to grayscale
gray_images=[]
for i in range(l):
    gray=cv2.cvtColor(images[i], cv2.COLOR_BGR2GRAY)
    gray_images.append(gray)
gray_images=np.array(gray_images)
##Single channnel images, grayscale
print(gray_images.shape)
```

```
(100, 480, 640)
```

[11]:
```
#Ploting all input images
for i in range(len(images)):
    plt.subplot(10,10,i+1),plt.imshow(gray_images[i],cmap='gray', vmin=0,␣
 ↪vmax=255)
    #plt.title(i)
    plt.xticks([]),plt.yticks([])
plt.show()
print(l,"- Images")
```

100 - Images

[12]:
```
#Considering 1st input image as reference
gray_ref_image=gray_images[0];
```

[13]:
```
#Calculating rmse score for each image w.r.t reference image
gray_rmse_score=[]
for i in range(1,l):
    gray_rmse=math.sqrt(np.square(np.subtract(gray_ref_image,gray_images[i])).
 ↪mean())
    gray_rmse_score.append(gray_rmse)
gray_rmse_score=np.array(gray_rmse_score)
print(gray_rmse_score)
```

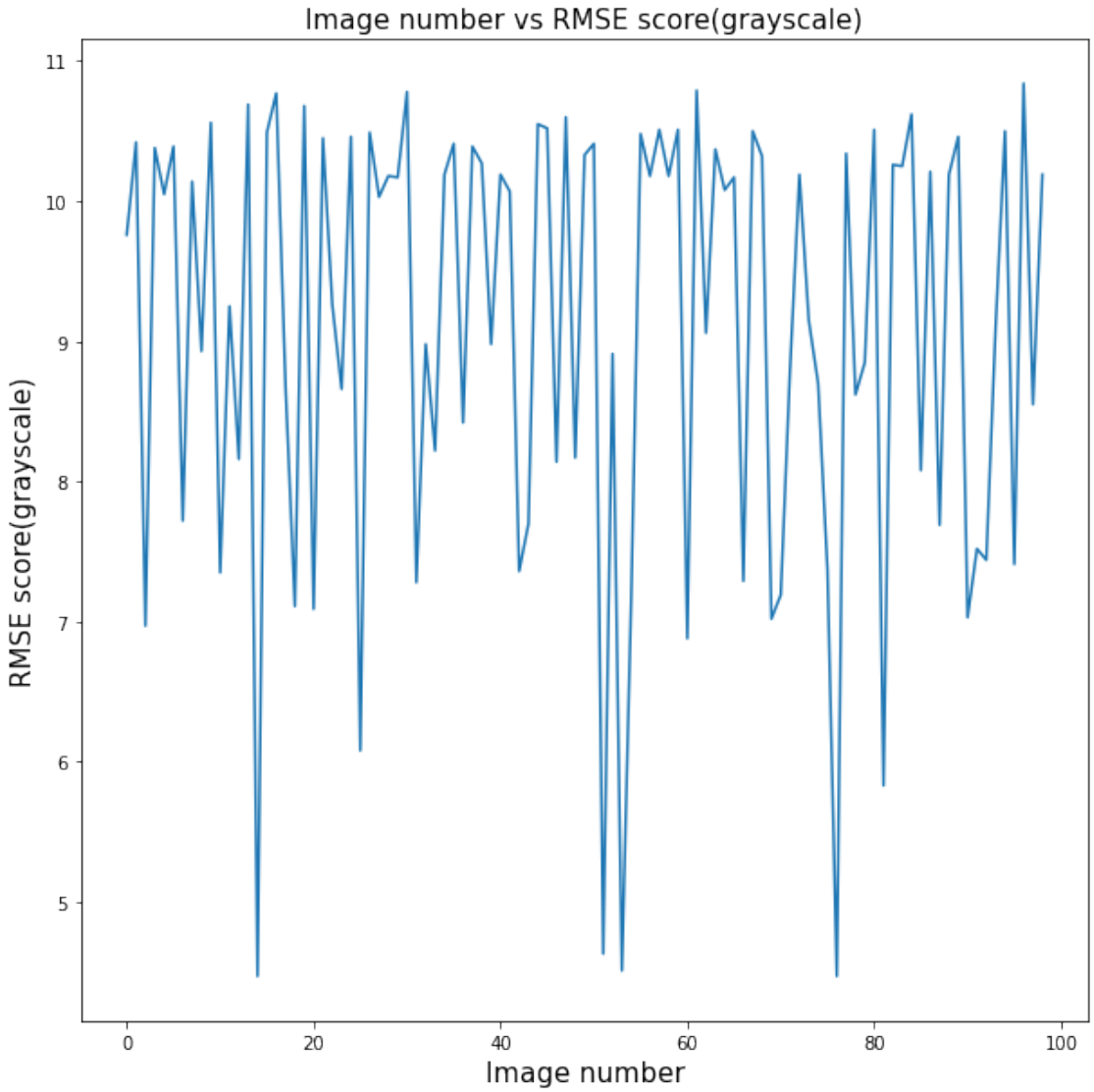
```
[ 9.75915719 10.42214642  6.97040689 10.38076522 10.05269077 10.3912445
```

```
 7.72140263 10.13758895  8.93387601 10.55983771  7.34592144  9.25330477
 8.16185452 10.68619327  4.47119979 10.48981277 10.77037057  8.66575535
 7.10790324 10.67544093  7.09293819 10.45154619  9.26301871  8.65777138
10.45527351  6.08496482 10.48577968 10.03320967 10.18384073 10.17421024
10.7785596   7.27604204  8.97950393  8.22450626 10.18860024 10.40701511
 8.41875039 10.38907805 10.27243717  8.97874244 10.1923295  10.06950213
 7.36370118  7.70499428 10.54995063 10.51618854  8.13695695 10.5969372
 8.16842303 10.32953517 10.41125161  4.62843728  8.91402895  4.50648982
 7.17944    10.47532303 10.18387477 10.5146274  10.18045689 10.50759197
 6.88351296 10.79425619  9.06405536 10.36626703 10.08441922 10.17058956
 7.29332176 10.50095435 10.32222758  7.02139216  7.19171412  8.81121112
10.18945597  9.14918082  8.6995321   7.36705456  4.47462027 10.33771302
 8.61913807  8.84910082 10.5106187   5.8296362  10.25772971 10.24892048
10.61810039  8.08250627 10.21182344  7.69434422 10.18707853 10.46497615
 7.03356281  7.51952104  7.44485084  9.11106313 10.50338797  7.41419032
10.84453783  8.54695193 10.18532473]
```

[14]:
```
#Rounding off values to two decimal places
gray_rmse_score=np.round_(gray_rmse_score, decimals=2)
print(gray_rmse_score)
```

```
[ 9.76 10.42  6.97 10.38 10.05 10.39  7.72 10.14  8.93 10.56  7.35  9.25
  8.16 10.69  4.47 10.49 10.77  8.67  7.11 10.68  7.09 10.45  9.26  8.66
 10.46  6.08 10.49 10.03 10.18 10.17 10.78  7.28  8.98  8.22 10.19 10.41
  8.42 10.39 10.27  8.98 10.19 10.07  7.36  7.7  10.55 10.52  8.14 10.6
  8.17 10.33 10.41  4.63  8.91  4.51  7.18 10.48 10.18 10.51 10.18 10.51
  6.88 10.79  9.06 10.37 10.08 10.17  7.29 10.5  10.32  7.02  7.19  8.81
 10.19  9.15  8.7   7.37  4.47 10.34  8.62  8.85 10.51  5.83 10.26 10.25
 10.62  8.08 10.21  7.69 10.19 10.46  7.03  7.52  7.44  9.11 10.5   7.41
 10.84  8.55 10.19]
```

[15]:
```
# Plot the values in the graph with the indexed images in the X-axis and RMSE␣
 ↪score in the Y-axis.
plt.title("Image number vs RMSE score(grayscale)", fontsize=15)
plt.xlabel("Image number",fontsize=15)
plt.ylabel("RMSE score(grayscale)",fontsize=15)
#plt.plot(img_lst,gray_rmse_score)
plt.plot(gray_rmse_score)
##forcing integer x-axis
"""
x=img_lst
new_list = range(math.floor(min(x)), math.ceil(max(x))+1)
plt.xticks(new_list)
"""
plt.show()
```



1.2.1 ————-You have reached the END of this Jupyter Notebook————-