BIRZEIT UNIVERSITY

Faculty of Engineering & Technology

Electrical & Computer Engineering Department

COMPUTER VISION

ENCS5343

Assignment #2

CBIR

Computer Vision

Prepared by : Qutaiba Olayyan 1190760

Instructor : Dr. Aziz Qaroush

Section : 1

Date : 23/12/2023

Abstract

This report explores the implementation and evaluation of a Content-Based Image Retrieval (CBIR) system, a technology that finds relevant images based on their content rather than textual descriptions. The main goal is to create a functional CBIR system using color histogram and color moment features and assess its performance through experimentation. The tasks involve building the system, experimenting with color histogram, and evaluating color moments with different weights. The report will utilize Wang database, detailing the system architecture, implementation specifics, and experimental setups. It aims to contribute insights into the effectiveness of color features for image retrieval, offering recommendations for future work.

Table of Contents

Table of Figures

I. Introduction

Let's explore a clever way of discovering pictures! It's called Content-Based Image Retrieval, or CBIR for short. Instead of using words, CBIR looks at what's actually in the pictures. Imagine having a bunch of photos and wanting to find ones that are similar. CBIR helps with that.

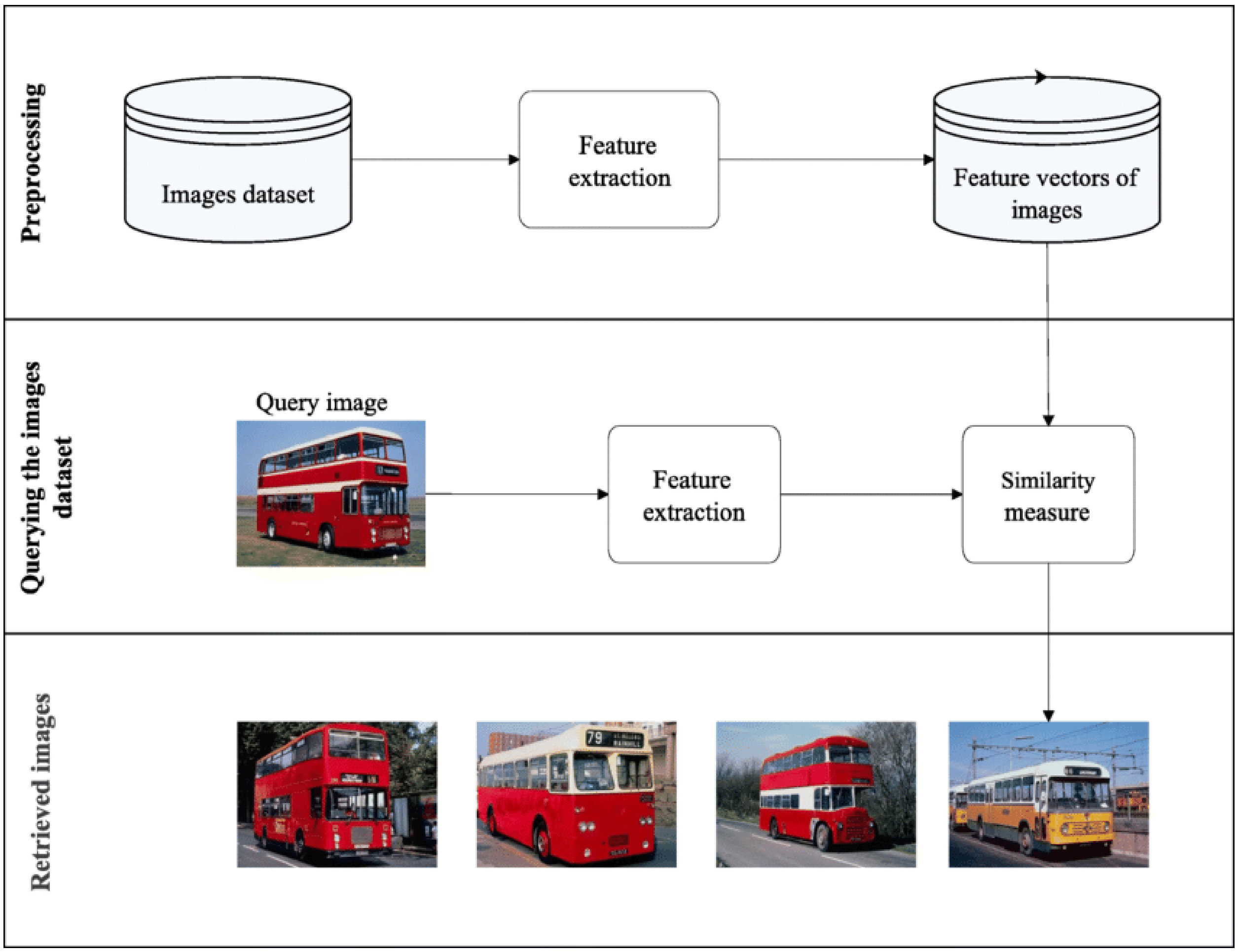

Figure 1: CBIR System

Now, onto the technical side of things! CBIR uses color features to do its magic. Two important things it uses are color histograms and color moments. Think of a color histogram as a summary of all the colors in a picture. Color moments help us understand things like the average color and how spread out the colors are.

A color histogram works by counting how many pixels have each color, organizing them into buckets for red, green, and blue. This creates a kind of color map for the picture. When comparing pictures, we can look at their color histograms. If the histograms have similar amounts of red, green, and blue, it suggests the pictures are alike in terms of color. It's like a visual fingerprint that our picture-finding system uses to understand and match pictures based on their unique color patterns.

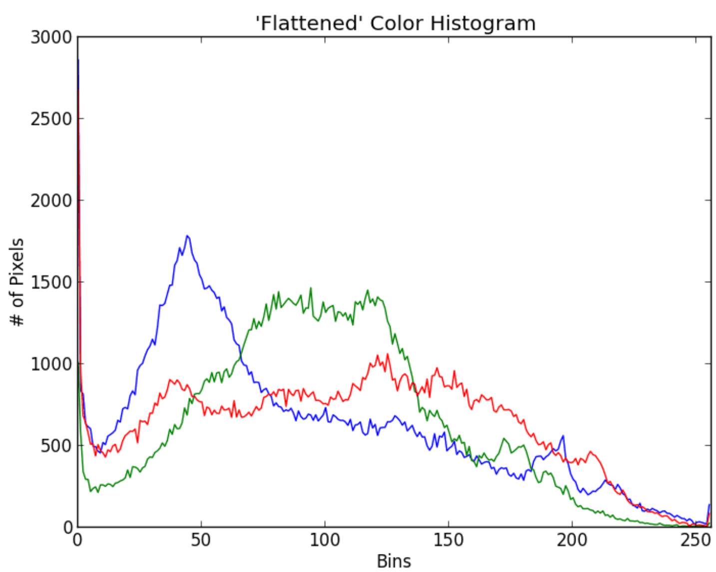

Figure 2: Color Histograms

Color moments are like the storytellers of a picture's color tale, revealing various aspects that go beyond just counting. Imagine each color as a character, and color moments help us understand their roles in the picture story.

- Mean: This is like the average color, telling us the central color theme of the picture. If we blend all the colors together, we get the mean.
- Standard Deviation: Think of this as a measure of how spread out the colors are. If the colors are all close to the average, the standard deviation is small; if they're spread out, it's larger.
- Skewness: Skewness gives a hint if one side of the color distribution is "heavier" than the other. It's like noticing if more colors are on the bright or dark side.
- Median: The median is the middle color value. If colors were lined up from lightest to darkest, the median would be the one in the middle.
- Mode: Mode points to the most common color. It's like finding the color that appears the most in the picture.
- Kurtosis: This tells us about the shape of the color distribution. If it's more peaked, like a mountain, it has high kurtosis; if it's flatter, it has low kurtosis.

Together, these color moments help our system grasp the nuances in a picture's color story, providing a richer understanding beyond just counting colors. They are the details that make each picture's color narrative unique.

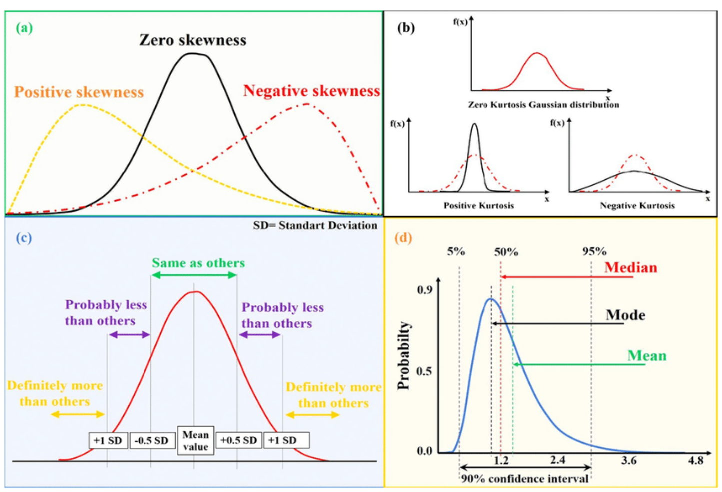

Figure 3: Color Moments

So, in simpler terms, CBIR is about finding pictures based on their content, and it uses color histograms and color moments to make it happen. Let's dive into how it all works!

II. System Implementation

Each module has a specific job. First, there's the "loading images" brings in all the pictures and the query image. Then, the "feature extraction" looks at each picture and figures out its color features. Next, the "distance computation" decides how similar pictures are by measuring the color differences. Lastly, the "ranking" arranges the pictures from most similar to least similar.

Now, let's chat about the technical stuff! We used Kaggle platform with Python language to build our system. I used libraries like OpenCV to handle pictures, Matplotlib to show us results, and NumPy for math stuff. We also borrowed some special skills from the Scipy library for statistics and the scikit-learn library for calculating area under the curve.

System Pipeline:

1. Read the dataset of images.
2. Get 10 random queries from the dataset, after removing it from the collection.
3. Create different features extraction methods for the images.
4. Choose 6 different retrieval thresholds.
5. For each extraction method I applied 10 queries, and for each query I applied 6 different retrieval thresholds.
6. I applied the Euclidean distance then ranked the results.
7. For each iteration I found the recall, precision and f1-score, then take the average of it
8. I used the results of recalls and precisions for plot the ROC curve.
9. At the end I calculated the area under the curve.

III. Experimental Setup and Results

III.1 Wang Dataset

Wang dataset has 1000 images from 10 different categories as shown in below figure, where 100 images correspond to each category.

I picked 10 queries to test the system methods. Each category I chose random image from its collection then removed the image from the collection, so each query image has 99 relevant images because the matching should be based on the category.

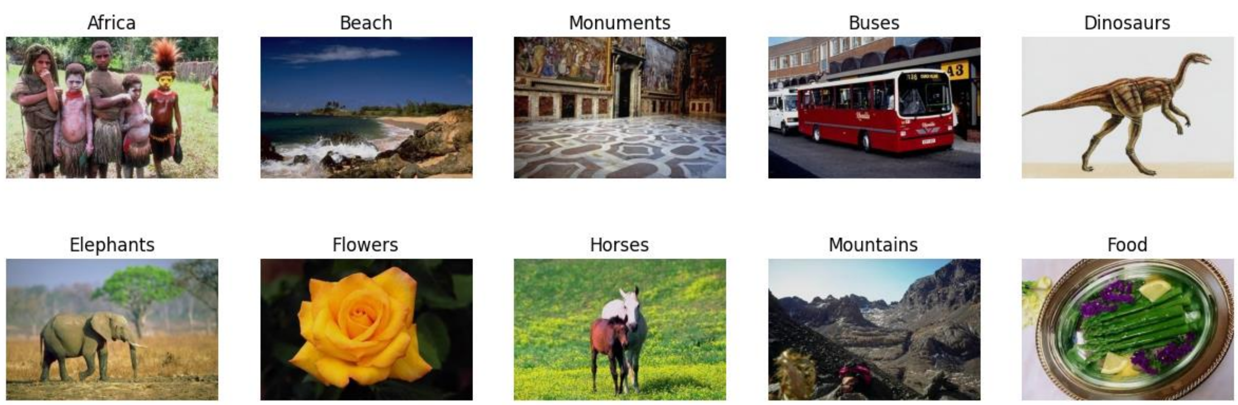

Figure 4: Wang Dataset

III.2 Euclidean Distance

```
def euclidean_distance(vector1, vector2):
    vector1, vector2 = np.array(vector1), np.array(vector2)
    distance = np.sqrt(np.sum((vector1 - vector2)**2))
    return distance
```

Figure 5: Euclidean Distance

III.3 Performance Metrics

```
def cal_performance_metrics(category, results):
    number_of_relevant_images = 99 # fixed for each category
    matching = len(list(filter(lambda res: res == category, results)))

    recall = matching / number_of_relevant_images
    precision = matching / len(results)

    f1_score = 0 if (precision + recall) == 0 else (2 * precision * recall) / (precision + recall)

    return recall, precision, f1_score
```

Figure 6: Performance Metrics

III.4 Results

III.4.1 Color Histogram 120 pins

Evaluation Metrics:

- Recall: 0.211
- Precision: 0.371
- F1_Score: 0.217
- AUC: 0.185
- Time: 1.13s

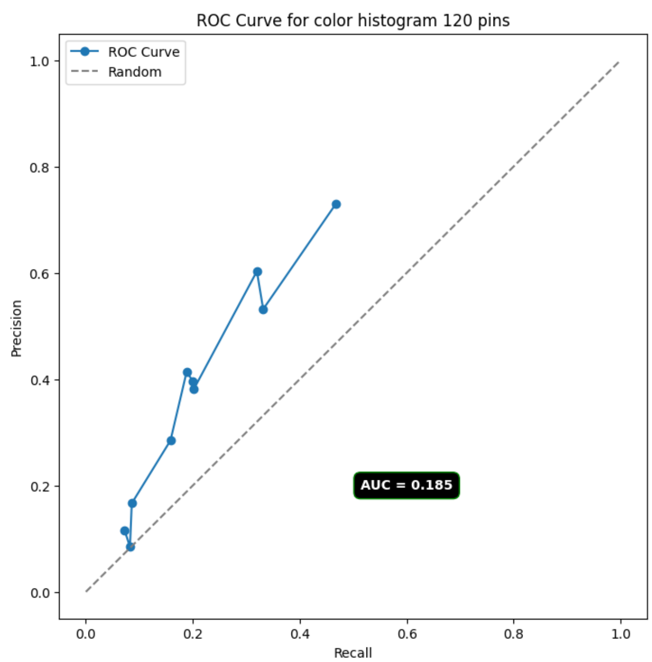

Figure 7: Color Histogram 120 pins

III.4.2 Color Histogram 180 pins

Evaluation Metrics:

- Recall: 0.212
- Precision: 0.369
- F1_Score: 0.218
- AUC: 0.183
- Time: 1.116s

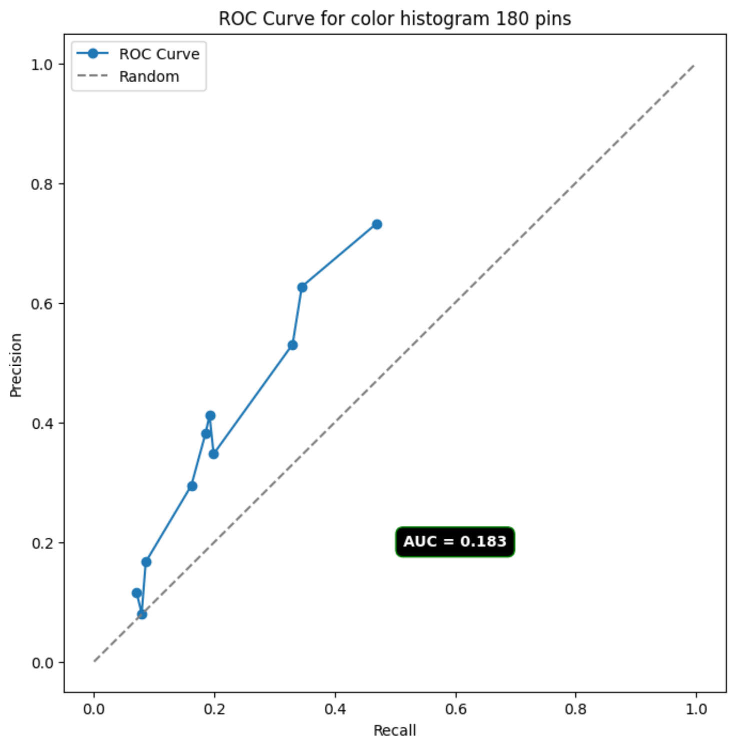

Figure 8: Color Histogram 180 pins

III.4.3 Color Histogram 360 pins

Evaluation Metrics:

- Recall: 0.209
- Precision: 0.368
- F1_Score: 0.216
- AUC: 0.177
- Time: 1.155s

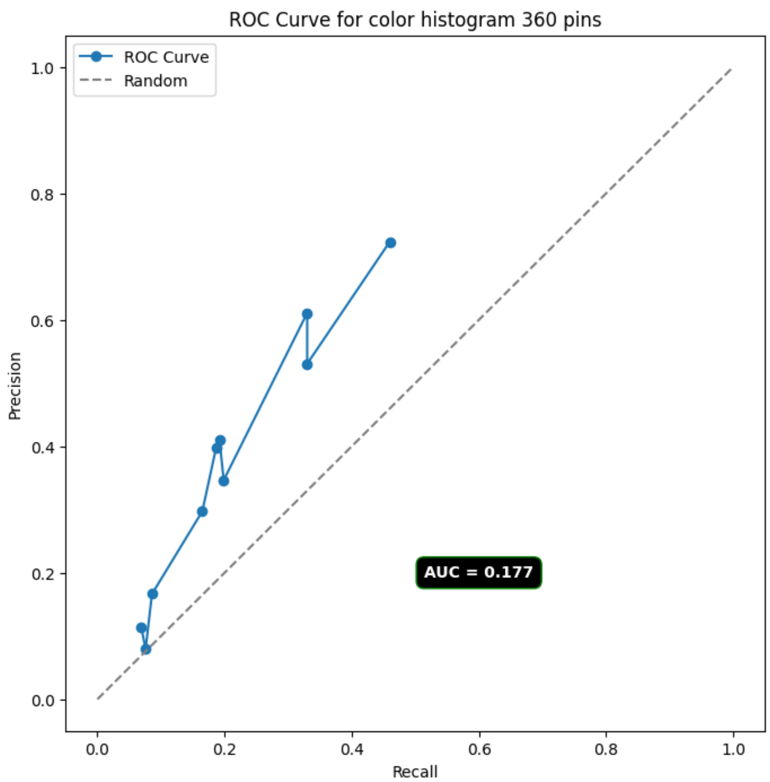

Figure 9: Color Histogram 360 pins

III.4.4 Color Moments 3 Metrics with equal weights

Three metrics for each RGP channel = [mean, std, skewness]

Evaluation Metrics:

- Recall: 0.297
- Precision: 0.416
- F1_Score: 0.296
- AUC: 0.291
- Time: 7.5s

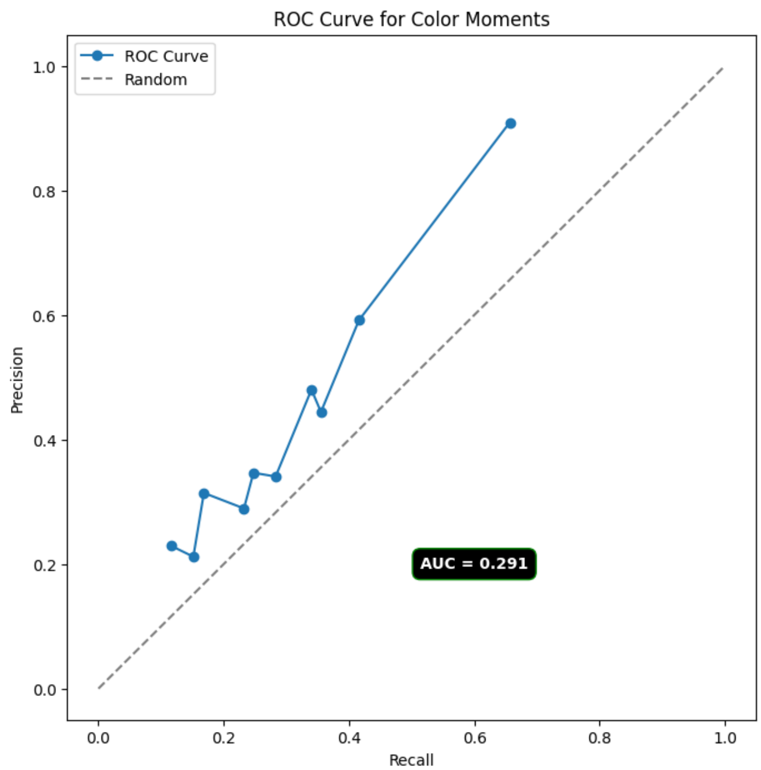

Figure 10: Color Moments 3 Metrics with equal weights

III.4.5 Color Moments 3 Metrics with different weights

Three metrics for each RGP channel = [mean, std, skewness]

Evaluation Metrics:

- Weights: [0.225, 0.166, 0.304, 0.265, 0.294, 0.833, 0.776, 0.047, 0.441]
- Recall: 0.258
- Precision: 0.373
- F1_Score: 0.257
- AUC: 0.317
- Time: 7.183s

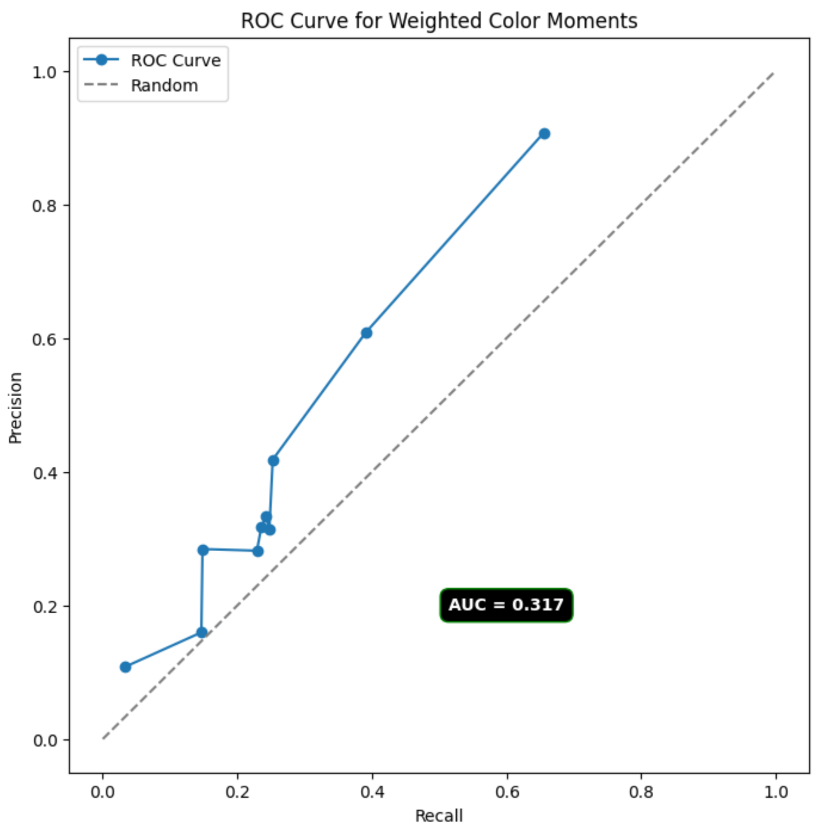

Figure 11: Color Moments 3 Metrics with different weights

III.4.6 Color Moments 6 Metrics with equal weights

Six metrics for each RGP channel = [mean, std, median, mode, skew, kurtosis]

Evaluation Metrics:

- Recall: 0.229
- Precision: 0.356
- F1_Score: 0.230
- AUC: 0.314
- Time: 27.041s

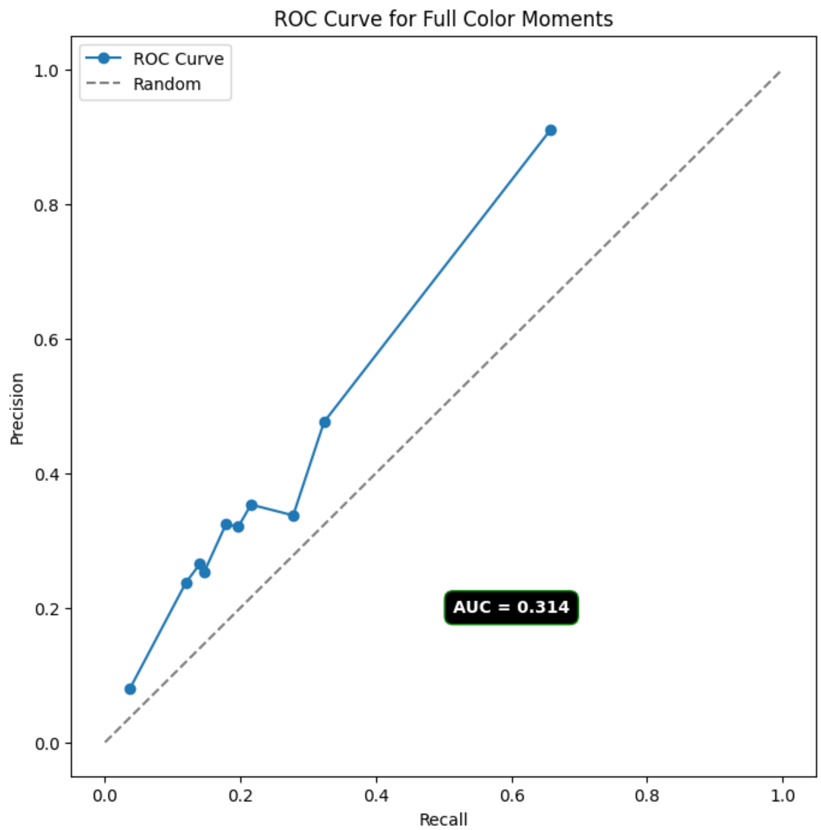

Figure 12: Color Moments 6 Metrics with equal weights

III.4.7 Color Moments 6 Metrics with different weights

Six metrics for each RGP channel = [mean, std, median, mode, skew, kurtosis]

Evaluation Metrics:

- Weights: [0.201, 0.063, 0.135, 0.029, 0.690, 0.675, 0.442, 0.091, 0.009, 0.830, 0.785, 0.753, 0.609, 0.233, 0.222, 0.662, 0.113, 0.351]
- Recall: 0.243
- Precision: 0.378
- F1_Score: 0.245
- AUC: 0.328
- Time: 26.726s

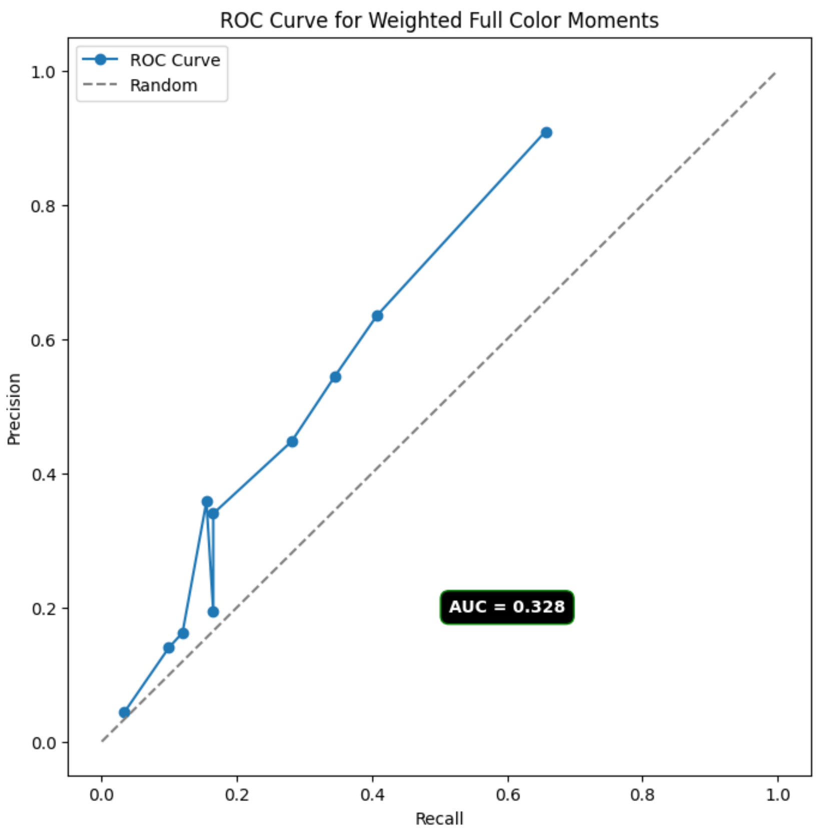

Figure 13: Color Moments 6 Metrics with different weights

III.4.8 Hu Moments

Evaluation Metrics:

- Recall: 0.194
- Precision: 0.283
- F1_Score: 0.193
- AUC: 0.310
- Time: 0.669s

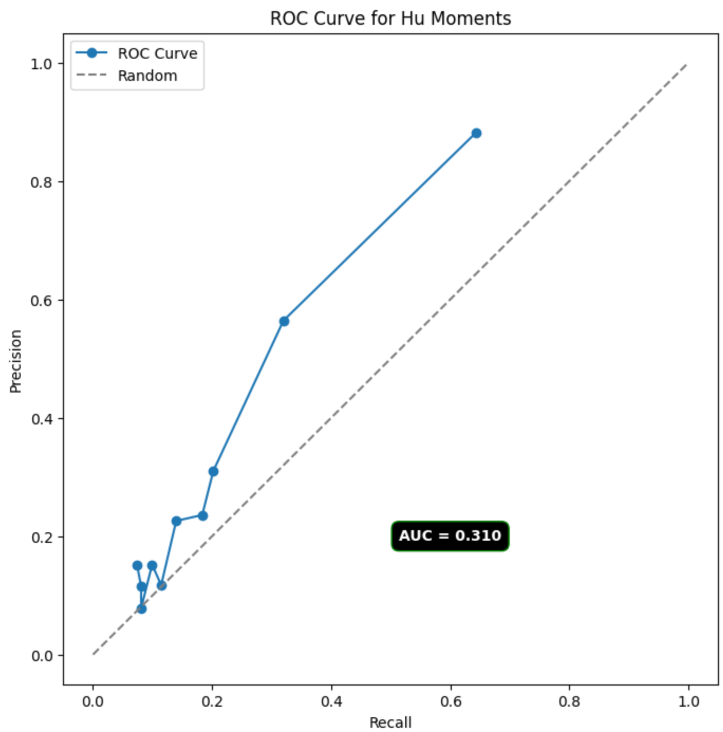

Figure 14: Hu Moments

III.4.9 HSV

Evaluation Metrics:

- Recall: 0.286
- Precision: 0.423
- F1_Score: 0.287
- AUC: 0.317
- Time: 12.977s

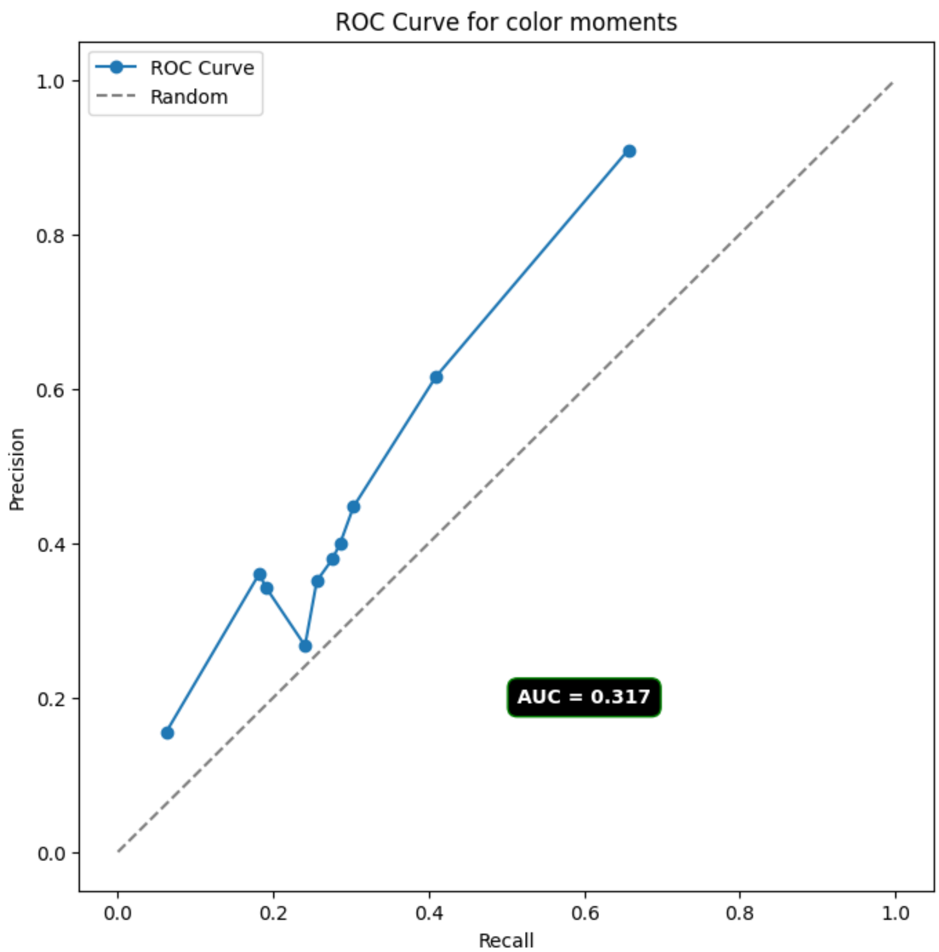

Figure 15: HSV

III.5 Retrieved Relevant Images

Figure 16: Queries

III.5.1 Color Histogram 120 pins

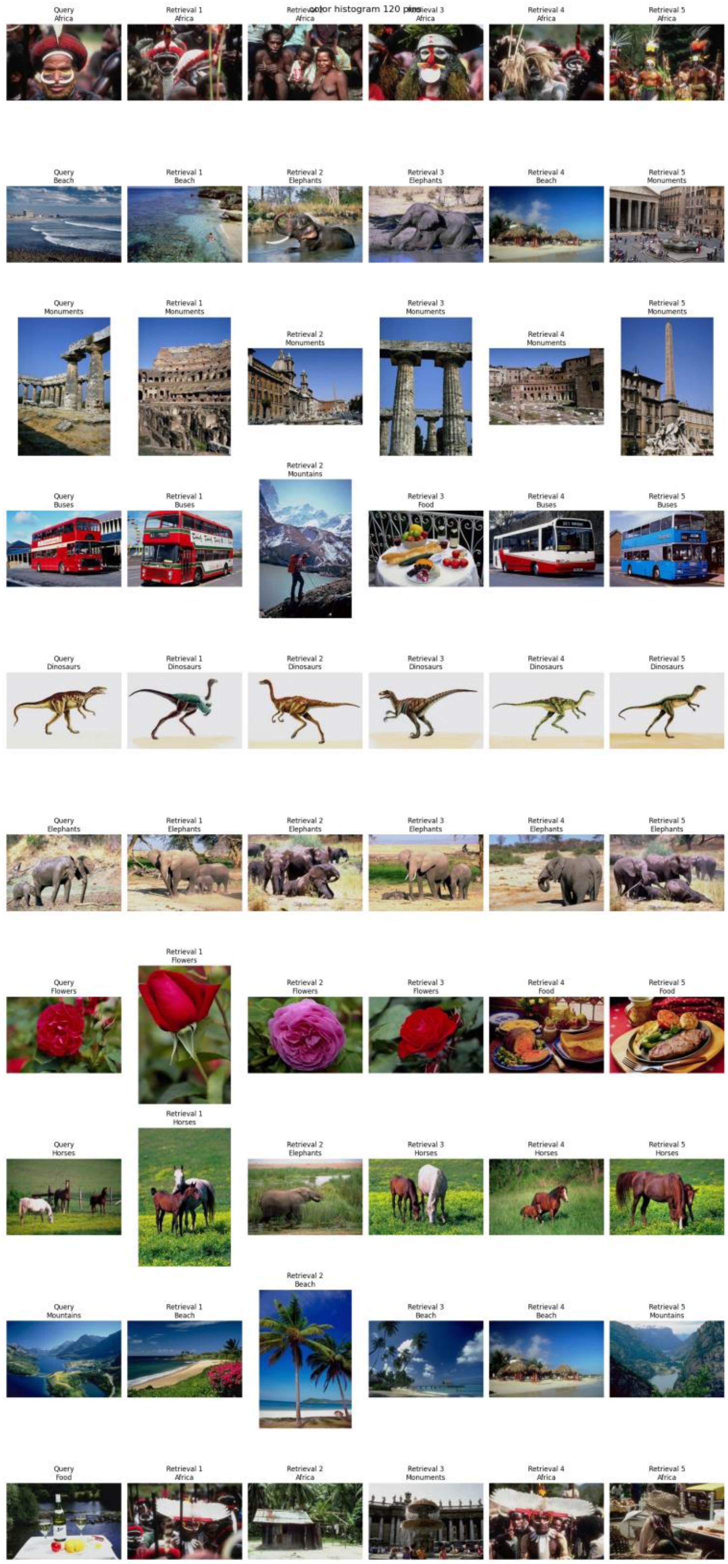

Figure 17: Color Histogram 120 pins Result

III.5.2 Color Histogram 180 pins

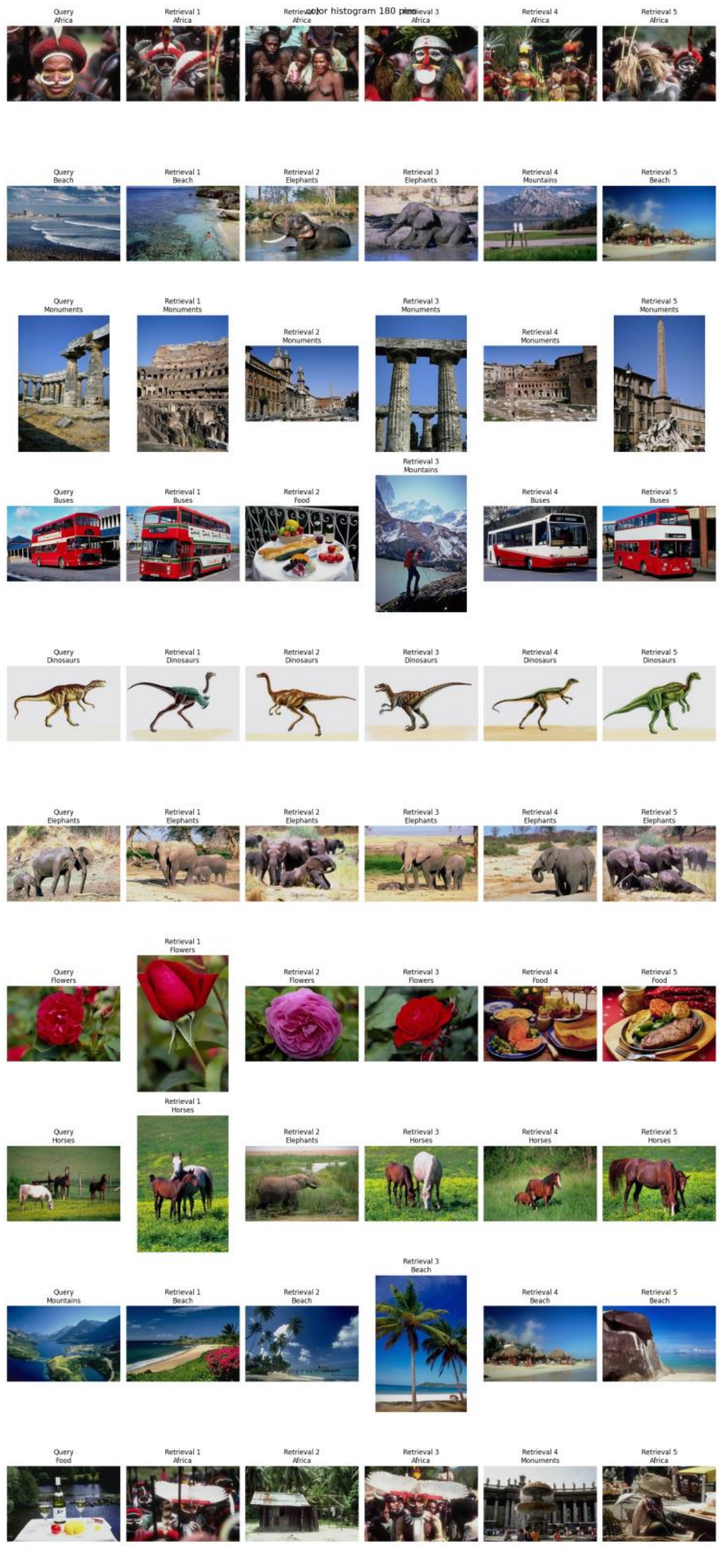

Figure 18: Color Histogram 180 pins Result

III.5.3 Color Histogram 360 pins

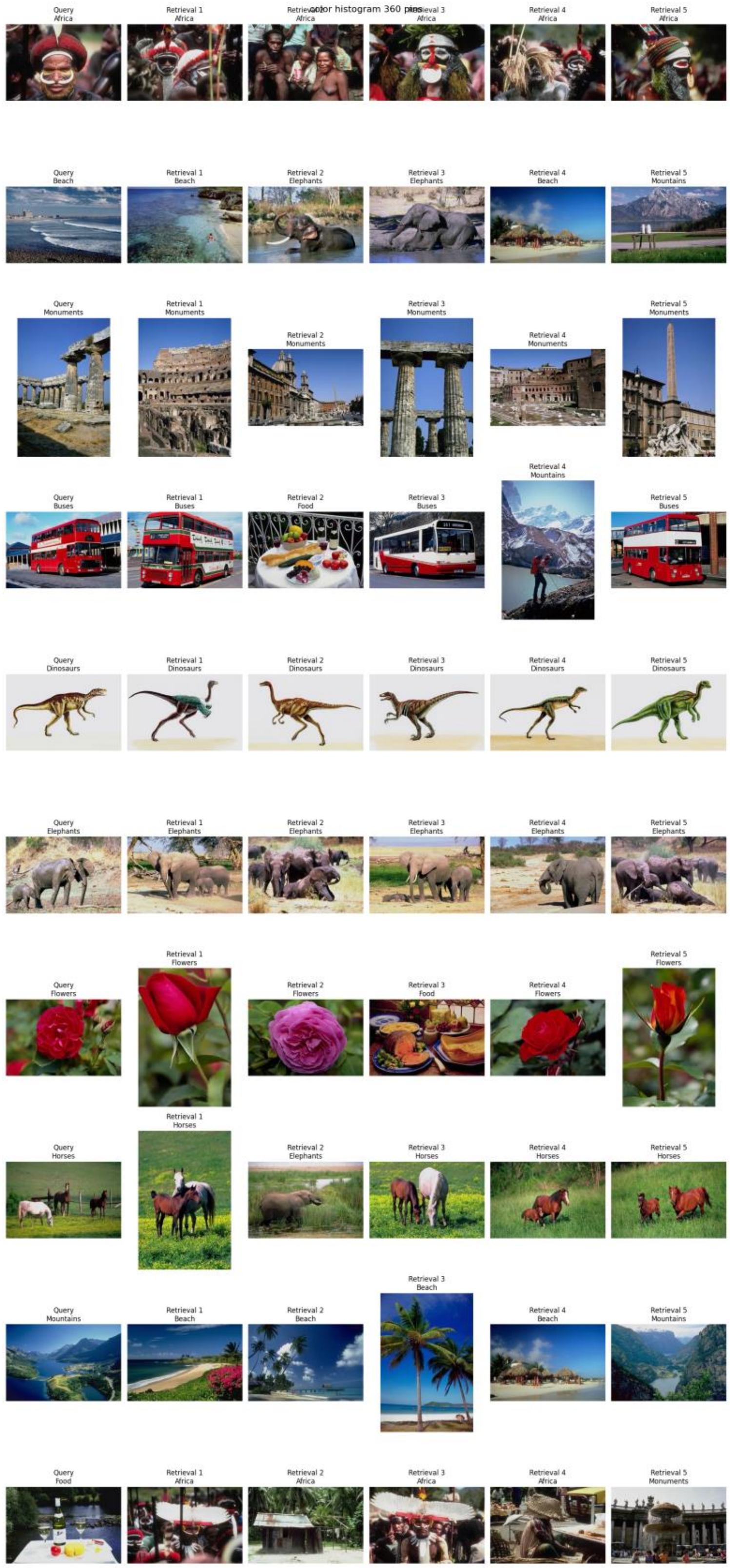

Figure 19: Color Histogram 360 pins Result

III.5.4 Color Moments 3 Metrics with equal weights

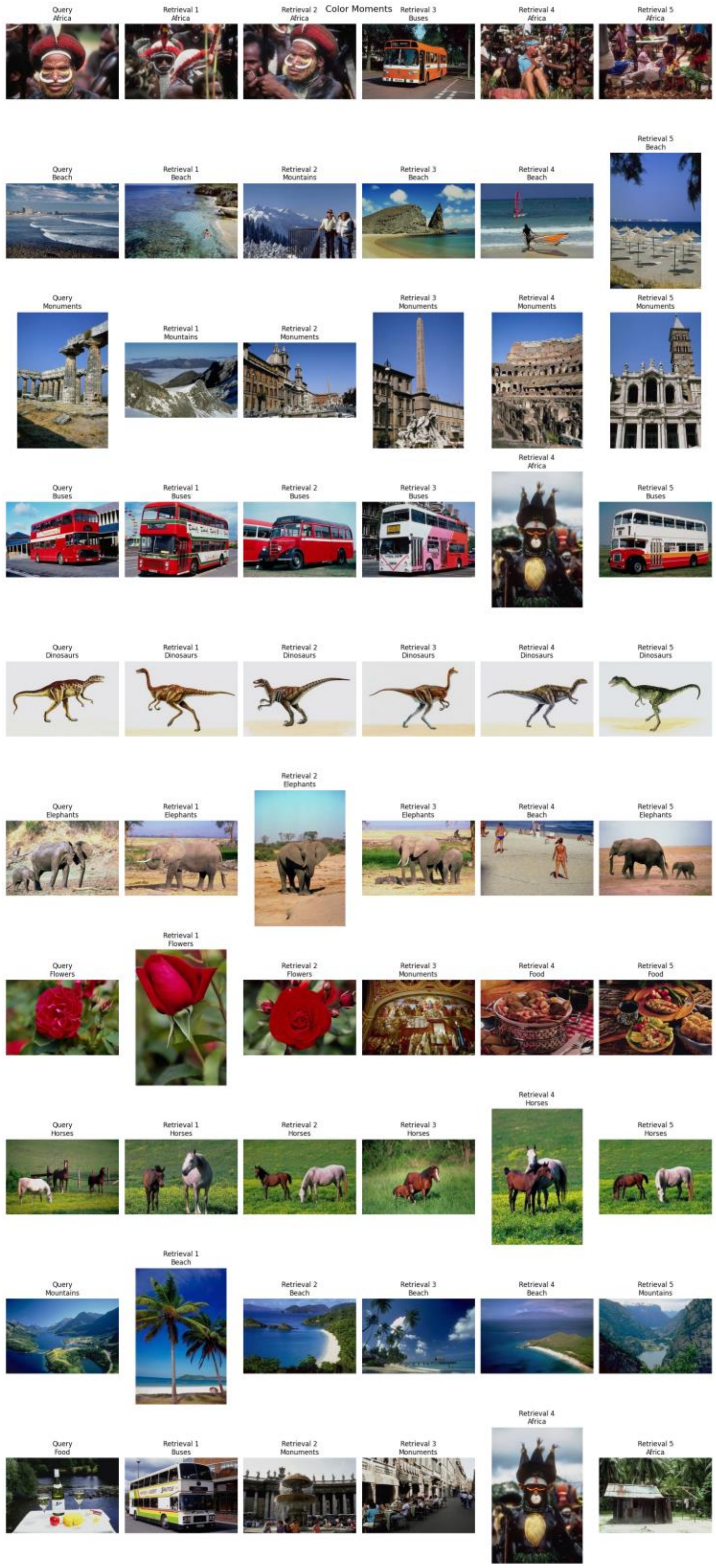

Figure 20: Color Moments 3 Metrics with equal weights Result

III.5.5 Color Moments 3 Metrics with different weights

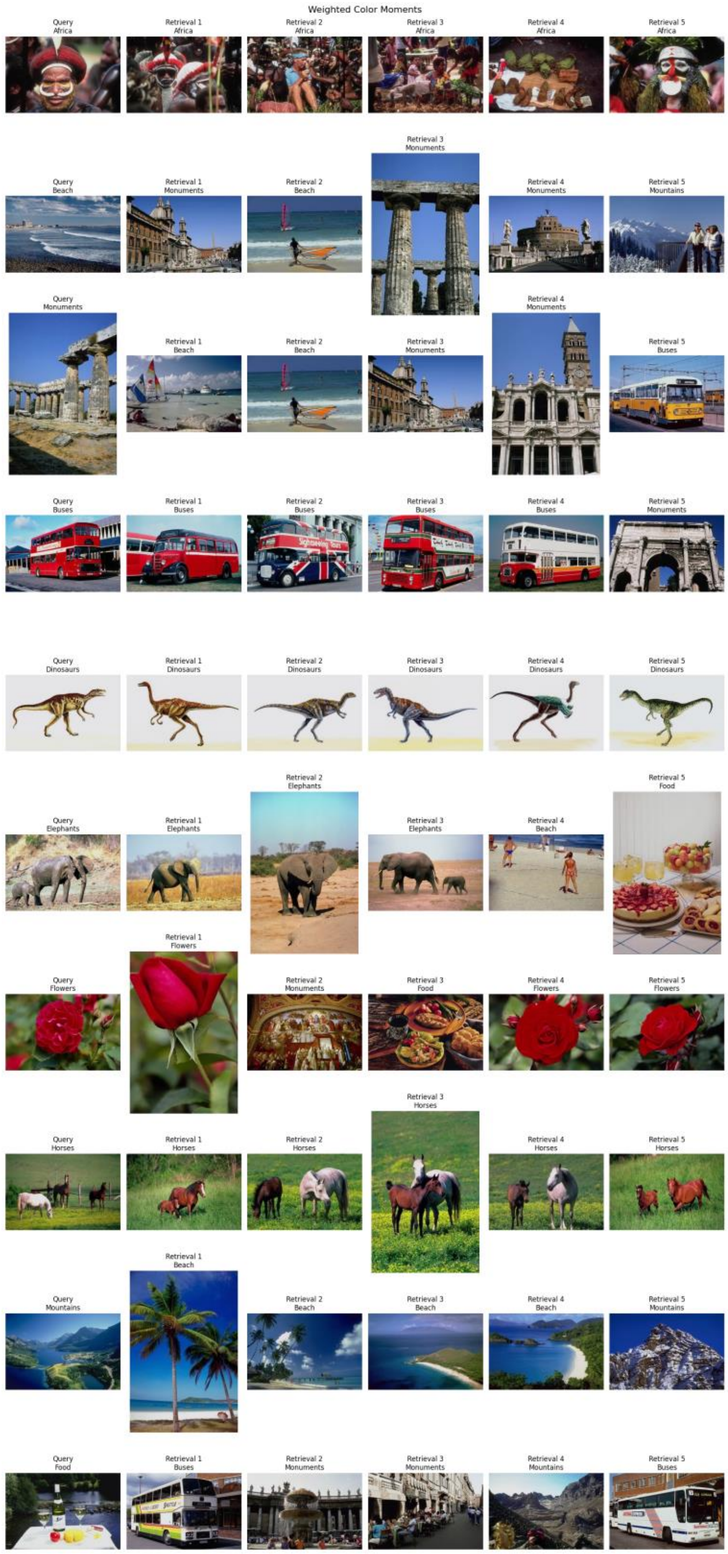

Figure 21: Color Moments 3 Metrics with different weights Result

III.5.6 Color Moments 6 Metrics with equal weights

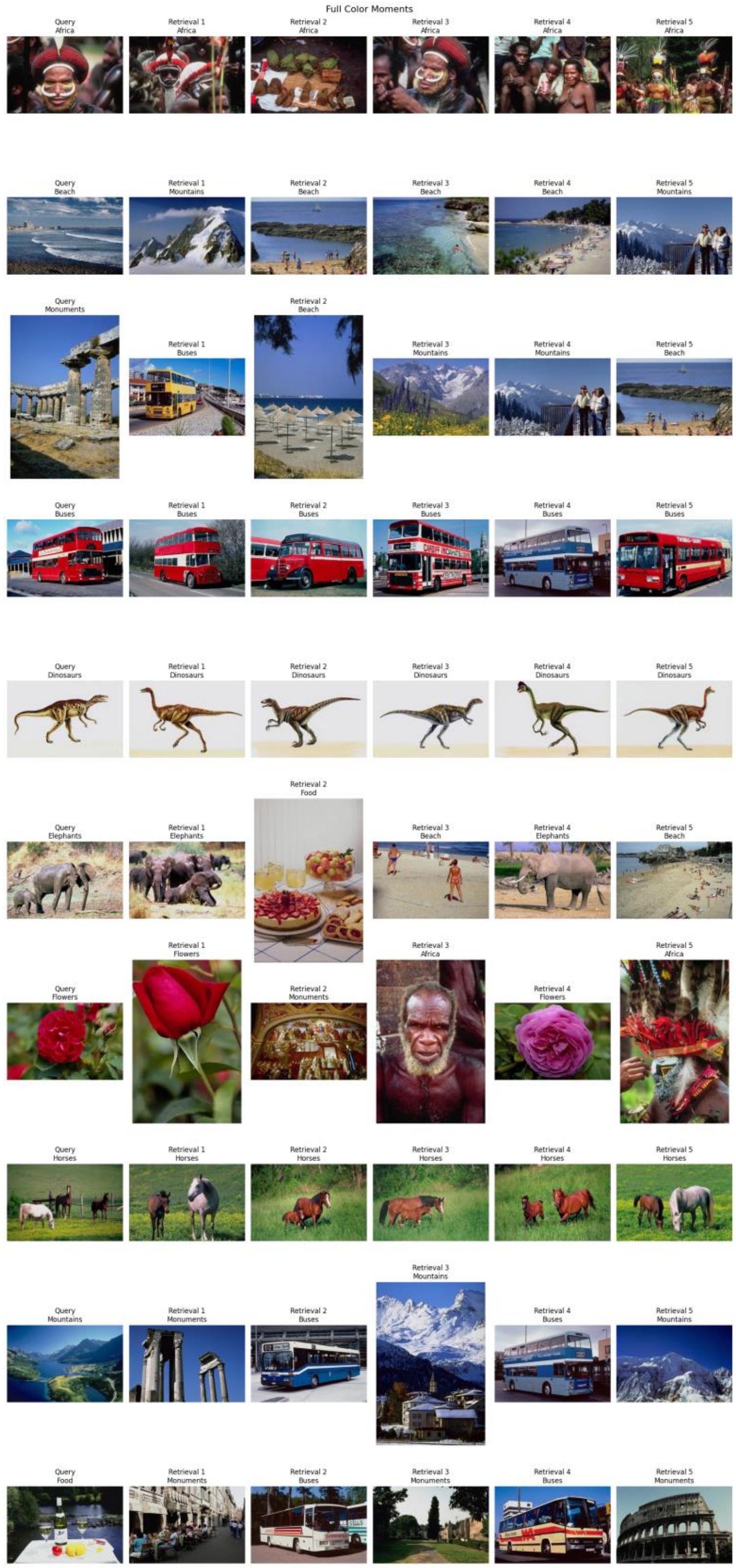

Figure 22: Color Moments 6 Metrics with equal weights

III.5.7 Color Moments 6 Metrics with different weights

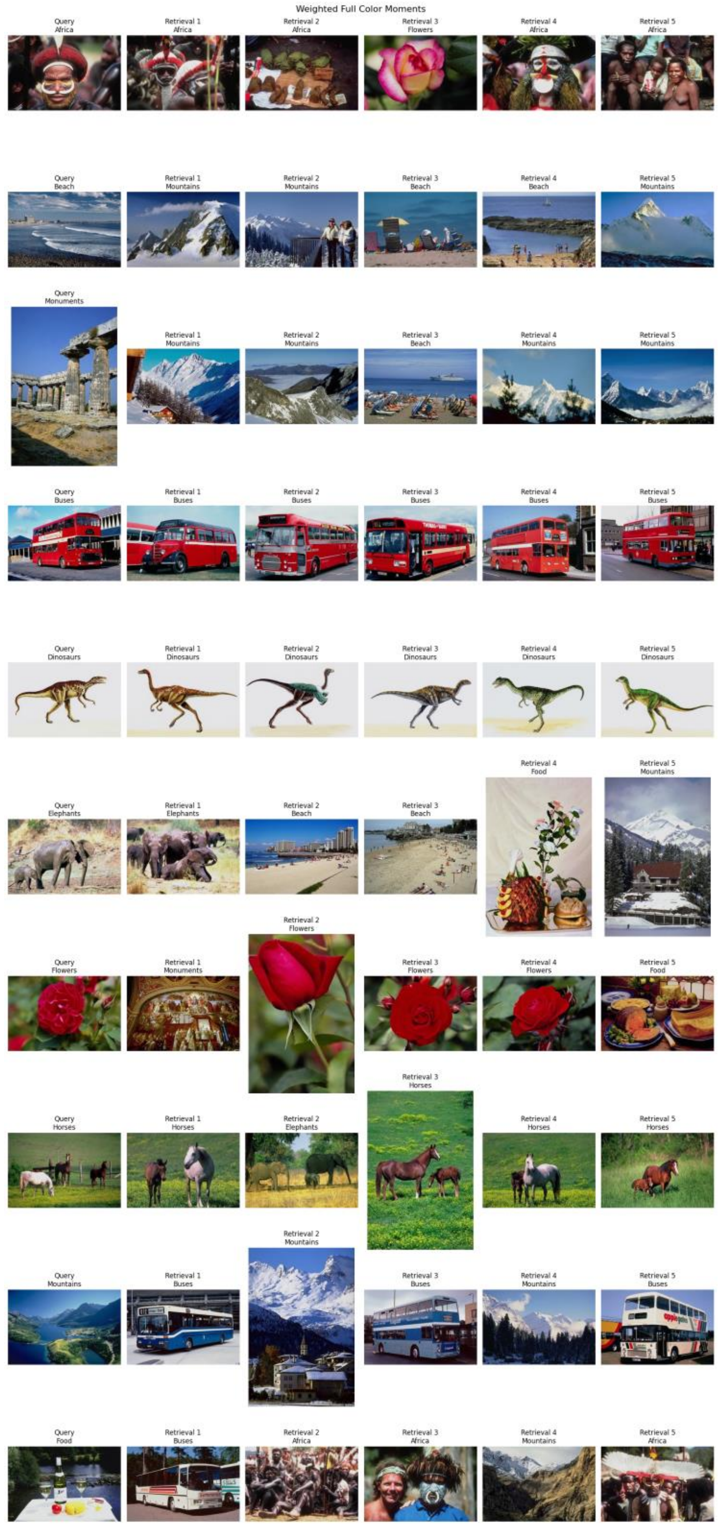

Figure 23: Color Moments 6 Metrics with different weights Result

III.5.8 Hu Moments

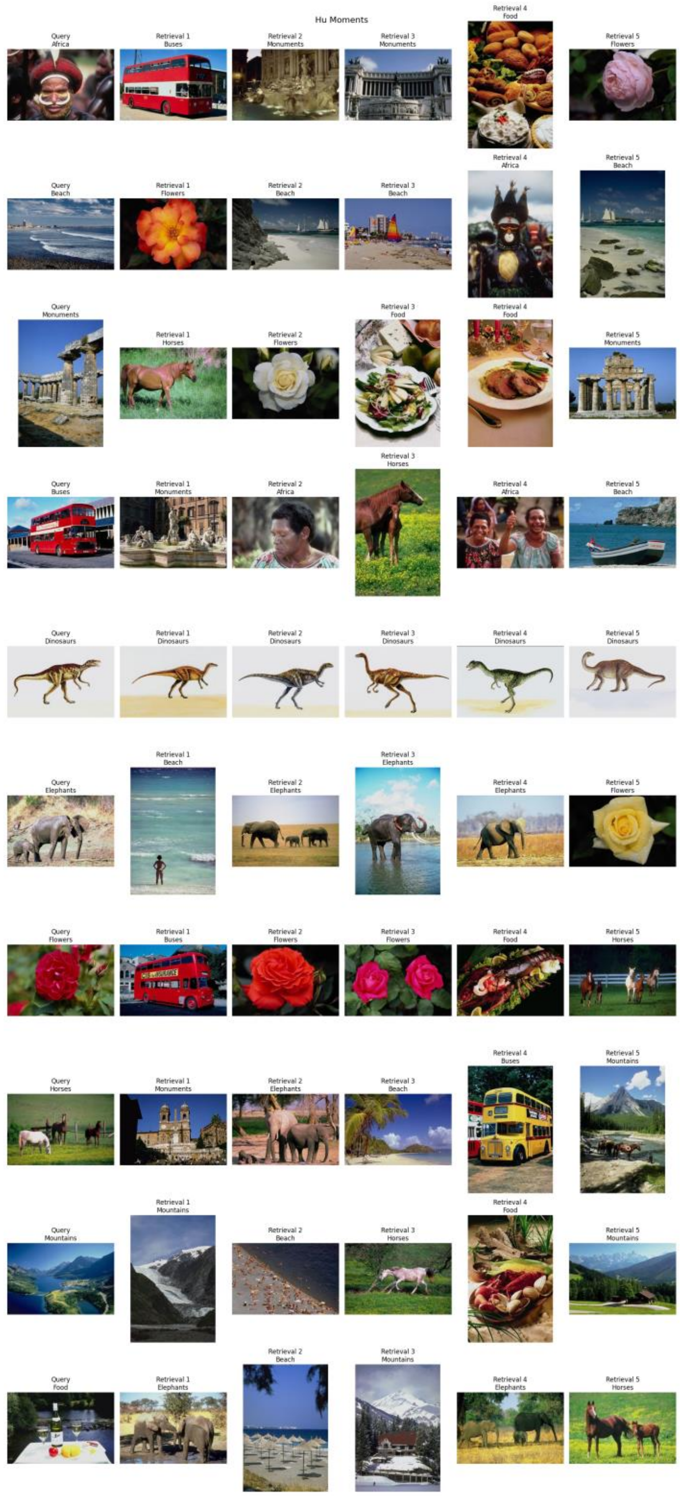

Figure 24: Hu Moments Result

III.5.9 HSV

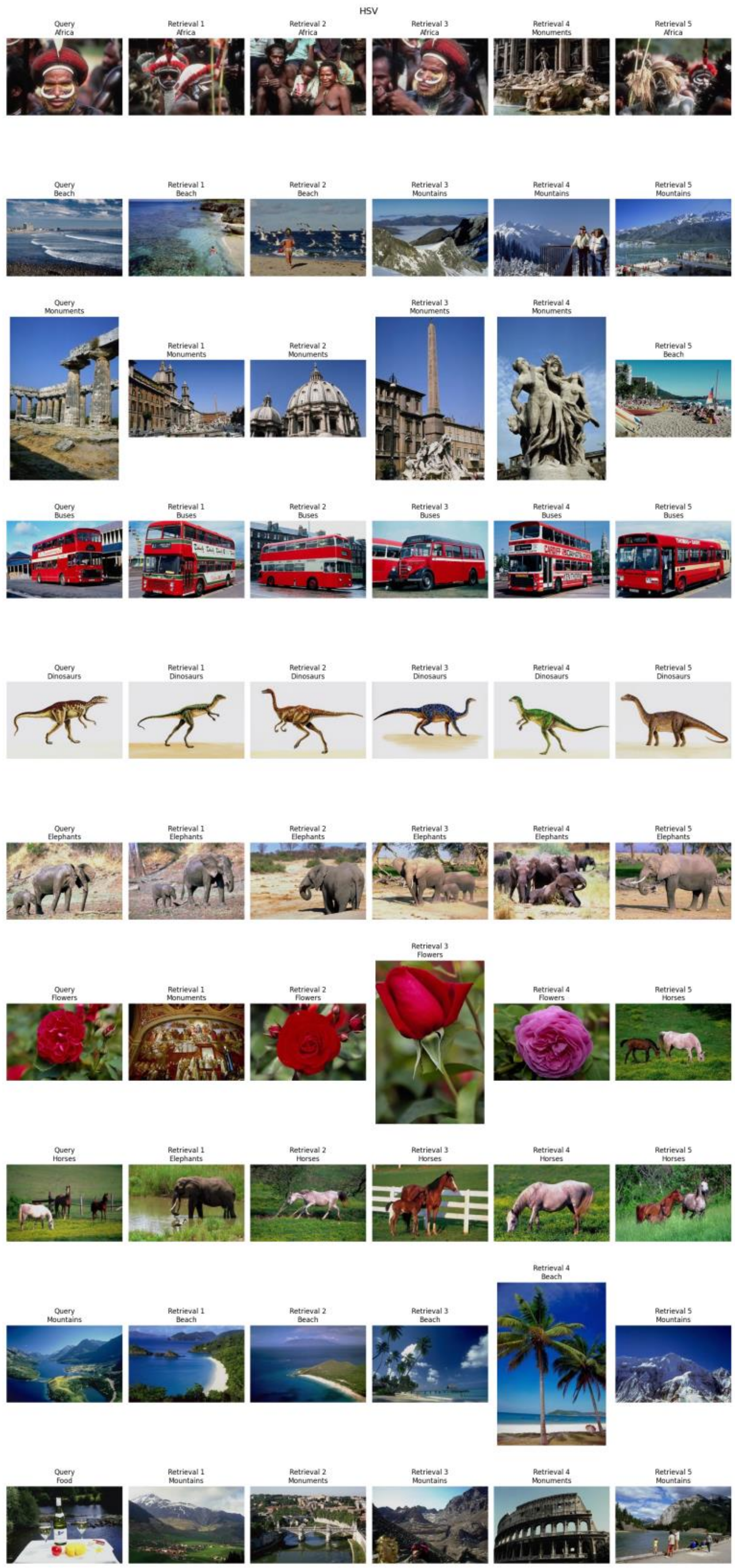

Figure 25: HSV Result

III.6 Discussion

Time Comparison: the time analysis revealed that color histograms exhibit significantly lower processing times compared to color moments, which require more computational resources. This indicates that color histograms are more efficient in terms of processing speed.

Optimal Pins for Color Histogram: when evaluating color histograms with different pin configurations, the results showed consistent performance across various pin numbers. This suggests that selecting the lowest number of pins can yield a less complex yet effective system, simplifying the image retrieval process.

Color Moment Metrics Importance: comparing color moments with three metrics to color histograms with different pin settings demonstrated superior performance. Notably, the skewness metric proved to be more crucial than standard deviation and mean, particularly when applying different random weights.

Effectiveness of Six-Metric Color Moments: the utilization of color moments with six metrics yielded the highest results. Through experimentation with various random weights, it became evident that skewness and kurtosis metrics hold greater significance than others in enhancing retrieval accuracy.

Features Extraction Methods – HSV: integration of different features extraction methods, such as HSV, revealed processing times falling between three-metric and six-metric color moments. The results closely aligned with the performance of three-metric color moments, emphasizing the efficiency of the latter.

Optimal Features Extraction Method - Hu Moments: among the features extraction methods, Hu Moments emerged as the most promising. It not only demonstrated results comparable to three-metric color moments but also exhibited processing times similar to color histograms. This method's efficiency and effectiveness in capturing shape information position it as a superior choice compared to other methods.

In summary, the comparative analysis of processing times and retrieval results indicates that the optimal choice depends on the specific requirements. While color histograms offer faster processing, three-metric color moments strike a balance between efficiency and accuracy. For those prioritizing shape information and seeking a well-rounded approach, Hu Moments prove to be the most favorable option, providing noteworthy results with reasonable processing times.

IV. Conclusion

To sum things up about finding pictures, we discovered that using color histograms is pretty fast and keeping things simple with fewer pins works well. Color moments, especially with three metrics, outshine histograms and get even better with six metrics.

During our exploration of color moments, skewness and kurtosis emerged as standout metrics, significantly influencing the success of our image retrieval system. Skewness, which gauges the asymmetry of color distribution, played a crucial role in discerning brightness and darkness tendencies, enhancing result accuracy. Similarly, kurtosis, measuring the shape of color distribution, proved pivotal in refining overall system performance. Our experiments, particularly with different random weights, highlighted the importance of these metrics in influencing retrieval accuracy. The careful consideration of skewness and kurtosis in color moments enhances the system's ability to capture and interpret nuanced aspects of visual content in images.

Exploring additional features extraction methods, HSV and Hu Moments emerged as noteworthy players in our quest for efficient picture retrieval. HSV, bridging the gap between three and six-metric color moments, showcases a balanced performance. On the other hand, the incorporation of Hu Moments, emphasizing shape information, proved to be particularly promising. Despite a processing time akin to color histograms, Hu Moments demonstrated results comparable to the more complex six-metric color moments. This positions Hu Moments as a compelling option, offering a harmonious blend of processing efficiency and retrieval accuracy.

Now, talking about the future, there's this cool thing called deep learning, like with Convolutional Neural Networks (CNN). They can understand pictures even better. Also, there's a model called YOLO that's great at spotting objects in images.

So, in the future, we might want to bring in these (CNN and YOLO) to make our picture-finding system even smarter. It's like adding extra brains to understand pictures in a more detailed way.

V. References

[1] Content Based Image Retrieval / Image Database Search Engine (SIMPLIcity, WIPE, Virtual Microscope) (psu.edu)