# Color Structure Descriptor-Based CBIR System

## PIV - Prog2

Gerard Agustina Ribera
gerard.agustina@estudiantat.upc.edu

Gil Jimenez Canellas
gil.jimenez@estudiantat.upc.edu

## Abstract

*This report extends the previous study on Content-Based Image Retrieval (CBIR) systems, concentrating on the utilization of color information. In particular, we investigate the application of the Color Structure Descriptor (CSD) from MPEG-7 as a tool for image characterization. We assess the performance of the CBIR system that employs CSD and compare it to the histogram-based method examined in the initial version of this system. Using quantitative metrics such as the F-measure and computational cost, we evaluate the outcomes of the CSD-based CBIR system. Our findings show that the CSD-based CBIR system surpasses the histogram-based approach in retrieval accuracy, albeit with a slightly higher computational cost. This study underscores the potential benefits of incorporating color information into CBIR systems and demonstrates the effectiveness of CSD as an image descriptor.*

## I. INTRODUCTION

This report outlines the advancements in the Content-Based Image Retrieval (CBIR) system developed in the PIV laboratory, building upon the previous version. To maintain consistency and allow for a direct comparison, the evaluation of the second version continues to use the same database of 2,000 images, featuring 500 distinct objects.

The second iteration incorporates significant enhancements, including the implementation of a color descriptor. By integrating the Color Structure Descriptor (CSD) from the MPEG-7 standard, we improve the system's ability to characterize images based on color information. This enhancement complements the previously employed histogram-based approach, providing a more comprehensive analysis of image content.

## II. OVERALL SYSTEM DESCRIPTION

The general system description for the second version remains consistent with that of the first, with the primary difference lying in the descriptors utilized. Rather than representing images as histograms, the second version uses the Color Structure Descriptor (CSD), which is also represented as a vector, denoted as $\mathbf{h}$.

The system takes an input file, *"input.txt"*, which contains the names of the images to be analyzed. For each image specified in the input file, the system performs the following steps:

1. Loads the matrix $\mathbf{H}$ and extracts the CSD vector, denoted as $\mathbf{h}$, from the current image.

2. Calculates the distance vector, $\mathbf{d}$, between the CSD vector $\mathbf{h}$ and all CSD vectors in $\mathbf{H}$.

3. Identifies the 10 smallest values in the distance vector $\mathbf{d}$ and retains their corresponding indices to determine the 10 image names to be written to the output file, *"output.txt"*.

In the next section, we will highlight the main differences between the first and current versions of the system, with a particular emphasis on the descriptor extraction process.

### Color Space: HMMD

In the second version of our CBIR System, we upgraded from black and white images to RGB images to better capture color details. To boost the effectiveness of the Color Structure Descriptor (CSD), we transform the images into the quantized HMMD color space. This transformation markedly improves the performance of the CSD descriptor.

HMMD color space consists of four color components named as H-hue, M-maximum, M-minimum and D-difference. HMMD color space components are derived

from the RGB values. This conversion is shown in (1) - (4) and Algorithm 1:

$$\textbf{Maximum} = max(R, G, B) \tag{1}$$

$$\textbf{Minimum} = min(R, G, B) \tag{2}$$

$$\textbf{Difference} = Maximum - Minimum \tag{3}$$

$$\textbf{Average} = \frac{(Maximum + Minimum)}{2} \tag{4}$$

**Algorithm 1:** Calculate Hue based on RGB values

**Input:** RGB values: R, G, B
**Output:** Hue value
Delta = Max -Min
**if** *Maximum = R & G* $\geq B$ **then**
  $\underline{H}ue = 60 \times \frac{G-B}{Delta}$
**else if** *Max = R & G* $\leq B$ **then**
  $\underline{H}ue = 360 + 60 \times \frac{G-B}{Delta}$
**else if** *G = Max* **then**
  $\underline{H}ue = 60 \times \left(2.0 + \frac{B-R}{Delta}\right)$
**else**
  $\underline{H}ue = 60 \times \left(4.0 + \frac{R-G}{Delta}\right)$
**return** *Hue*

The next step involves quantifying the four levels of the HMMD color space by consolidating them into a single level, using the provided table. This table allows us to assign a unique level to each HMMD color component, facilitating a quantitative representation of the HMMD color space for further calculations or analysis. Our system supports two quantification options: 128 levels and 256 levels.

| Subspace | 256 | | 128 | |
|---|---|---|---|---|
| | Hue | Sum | Hue | Sum |
| 0 [0-6) | 1 | 32 | 1 | 16 |
| 1 [6-20) | 4 | 8 | 4 | 4 |
| 2 [20-60) | 16 | 4 | 8 | 4 |
| 3 [60-110) | 16 | 4 | 8 | 4 |
| 4 [110-255] | 16 | 4 | 8 | 4 |

Table 1: HMMD COLOR SPACE QUANTIZATION

In Figure 1, you can see the 'ukbench00000' image in its original RGB format and its quantized version in the HMMD color space. The RGB image displays the true color representation, while the quantized HMMD image shows the same scene with color information condensed using predefined HMMD quantization levels. This process creates a more compact representation of the image's color data, which facilitates further analysis or processing within the HMMD color space.

(a) RGB (b) HMMD Quantification

Figure 1: Image 'ukbench00000.jpg' from the dataset

After converting the RGB image into its quantized HMMD representation, the subsequent step is to extract the Color Structure Descriptor (CSD).

## MPEG-7 Color Structure Descriptor

The Color Structure Descriptor (CSD) is a commonly employed method in still image retrieval. It aims to represent the local color structure within an image using a structuring element. While the CSD is similar in form to a color histogram, it differs in semantics. This approach enables the CSD to efficiently capture the distribution and relationships of colors within the image.

### CSD Algorithm

The system computes the Color Structure Descriptor (CSD) by sliding a structuring element of specific dimensions over the image pixels. Figure 2 illustrates an 8x8 structuring element. During each iteration of the algorithm, corresponding to each position of the structuring element on the image, the system examines the colors of the pixels within the element. For each unique color encountered, the corresponding position in the vector **h** is incremented by one.

It's important to note that even if a color appears multiple times within the structuring element, the increment for that color happens only once. Therefore, if the structuring element contains 64 different colors, 64 different positions in **h** will be incremented in that iteration. However, if all 64 pixels in the structuring element share the same color, only one increment will occur for that color in that iteration. This process repeats for each position of the structuring element, ultimately resulting in the full computation of the Color Structure Descriptor, represented by the vector **h**.

| Color | h(c) |
|---|---|
| $c_1$ | $h(c_1)+1$ |
| $c_2$ | $h(c_2)$ |
| $c_3$ | $h(c_3)+1$ |
| $c_4$ | $h(c_4)+1$ |

Before calculating the descriptor, the image must be subsampled according to its size. Research suggests that the subsampling factor and the size of the structuring element for CSD calculation generally adhere to a specific guideline.

For an image $\mathcal{I}_{h\times w}$ of dimensions, $h$ (height) and $w$ (width). Then, the constant $p$ is:

$$p = \max\{0,\ \text{round}\ (0.5 \cdot \log_2(h \cdot w) - 8)\} \qquad (5)$$

From p then we can compute $K$, the subsampling factor and $E$ the size of the structuring element $\mathcal{E}_{E\times E}$.

$$K = 2^p \qquad E = 8 \cdot K \qquad (6)$$

According to this rule (6), the size of the structuring element increases with the dimensions of the image. A larger structuring element E leads to longer execution times, necessitating the application of a downsampling factor. In our scenario, we use $p = 1, K = 2, E = 16$.

After obtaining the Color Structure Descriptor (CSD), represented by the vector **h**, a final processing step can be performed. Similar to the initial histogram analysis, different bin sizes can be used. The default size of **h** depends on the HMMD quantization level, either 128 or 256. From this point, we can merge the bins to work with a smaller vector **h**.

## III. RESULTS

Initially, our evaluation will focus on the input file provided by the lab professor. We will employ 256 levels of HMMD quantification and 256 bins. Our approach involves visualizing the Recall-Precision graph and computing the F-measure. This analysis will enable us to compare the system's performance across three distinct distance measures.

This time, unlike in the previous report, the chi-squared distance demonstrated the best performance among all the distance measures. Consequently, from now on, the analysis of other characteristics of the descriptor will be conducted exclusively using the chi-squared distance.

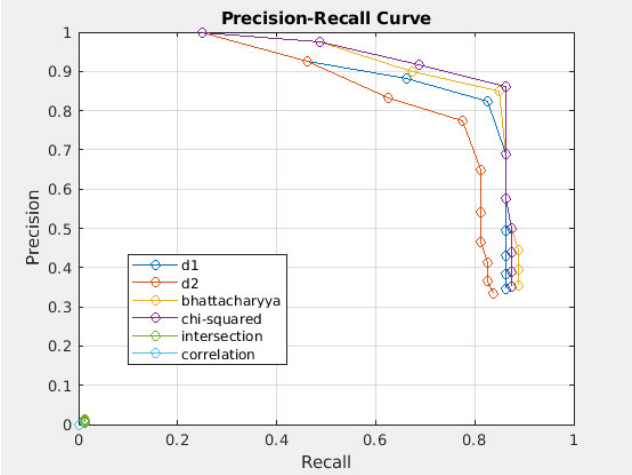


Figure 2: Precision-Recall Curve

This precision-recall graph shows the performance of our system using different distance metrics: Manhattan distance or L1 distance (blue curve), Euclidean distance (red curve), Bhattacharyya distance (yellow curve), Chi-squared distance (purple curve), Intersection (green curve) and Correlation (blue curve).

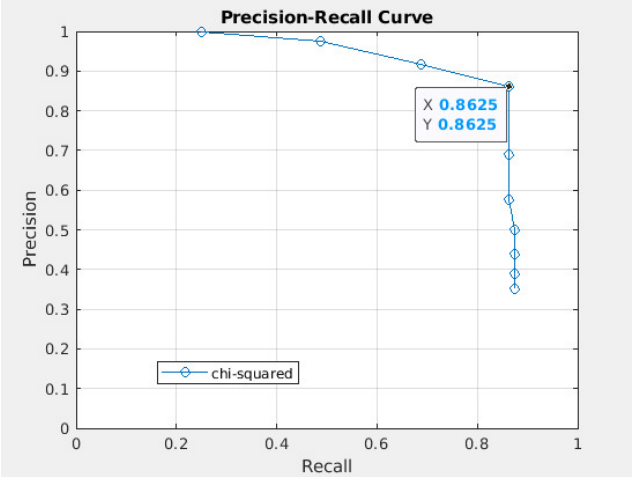


Figure 3: Chi-Squared curve

Here we show only the Chi-Squared graph, which has given us the best F-Score, to observe the points in more detail.

| | F-score | Execution time/image |
|---|---|---|
| 256 quantification | 0.8625 | 278 ms |

Table 2: Results for 20 images of the input.txt

Here we show only the Chi-Squared graph, which has given us the best F-Score, using the 256-level quantification. The table presents a comparison of the F-score and execution time per image for the 256-level quantification.

# IV. CONCLUSIONS

Here, we present only the Chi-Squared graph, which has provided the highest F-Score using 256-level quantification, to allow for a more detailed examination of the points. The table compares the F-score and execution time per image for the different distances. The 256-level quantification method achieves an F-score of 0.8625, indicating reasonably accurate classification.

It is important to note that these results are based on a specific experiment using the images from input.txt and may not be generalizable to other contexts. This, the results should be interpreted with caution, and further experimentation with different input.txt files may be necessary to confirm the conclusions.

Overall, the results align with our expectations. We found that, unlike in the first version, the study of bins showed appropriate behavior, indicating that a higher number of bins leads to better performance.

For this specific input file, the optimal configuration appears to be using the Chi-Squared distance with 256 levels of quantification. Table 3 below shows the best system configurations based on performance, listed from best to worst.

| Descriptor | Bins | Distance | F-score | Time/imag |
|---|---|---|---|---|
| CSD 256-quant | 256 | Chi Squared | 0.8625 | 278 ms |
| CSD 256-quant | 256 | Bhatacharyya | 0.85 | 294 ms |
| CSD 256-quant | 256 | L1 distance | 0.79 | 342 ms |
| CSD 256-quant | 256 | Euclidean distance | 0.75 | 326 ms |

Table 3: Better system settings

In summary, the Chi-Squared distance proves to be the most effective for achieving the highest F-Score. Additionally, a high number of bins is recommended for optimal performance, as evidenced by the 256-bin configuration, which achieved an impressive F-score of 0.8625. However, if memory conservation is a priority and fewer bins are required, a good F-score is still attainable. It is crucial to note that this optimal configuration is based on a specific experiment and may not be universally applicable. Therefore, further testing with different input files is necessary to validate these findings.

## Improvements

We can identify two primary areas for potential system enhancements: improving image description techniques and refining the decision-making process. In terms of image description, employing alternative methods like GoP/GoF (Group of Frames or Group of Pictures Descriptor) could lead to superior results.

To enhance the decision-making process, incorporating a machine learning system trained with the obtained descriptors is a viable strategy. This approach allows the system to learn from the descriptors and make more precise decisions.

Additionally, another approach worth considering is to replace the entire descriptor and decision-maker framework with a deep learning model. Specifically, using a convolutional neural network (CNN) to handle both tasks sequentially could be beneficial. By training the CNN on an extensive dataset, it can learn to extract meaningful features from the images and make informed decisions based on these features.

## References


[1] Authorless, Chi-Squared distance. https://en.wikipedia.org/wiki/Chi-square

[2] Authorless, bhattacharyya distance https://en.wikipedia.org/wiki/Bhattacharyya_distance

[3] NIST, Manhattan distance https://xlinux.nist.gov/dads/HTML/manhattanDistance.html

[4] Authorless, Euclidian distance https://en.wikipedia.org/wiki/Euclidean_distance

[5] Authorless, Content based image retrieval. https://en.wikipedia.org/wiki/Content-based_image_retrieval