UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Color-Based Image Retrieval

Image Processing

Author: Liviu Tcaci
Group: 30238

Faculty of Automation and Computer Science

May 10, 2024

Cuprins

1 Specifications

1.1 Problem Statement

The goal of this project is to build a color-based image retrieval system using one-, two-, and three-dimensional histograms in both RGB and HSV color spaces. These histograms will be plotted to illustrate the distribution of color components, and feature vectors will be compared—primarily via Euclidean distance—to measure similarity between images. By combining different histogram dimensions and color models, the system should rank and retrieve images whose color characteristics closely match a query.

1.2 Objectives

- **Color Space Conversion:** Implement accurate conversions between RGB and HSV formats to support different histogram analyses.
- **Histogram Computation & Visualization:** Compute 1D, 2D, and 3D histograms for both RGB and HSV channels, and generate clear graphical representations of each.
- **Similarity Measurement:** Develop methods to compare image histograms using Euclidean distance on the resulting feature vectors, enabling quantitative assessment of color similarity.
- **User Interface:** Design and implement a simple, intuitive interface that allows users to view histogram plots and perform color-based image searches interactively.

With these components in place, the final system will allow users to submit an example image or select a color profile and instantly retrieve the most visually similar images from the database based on their color distributions.

2 Theoretical Foundation

2.1 Color Spaces: RGB and HSV

RGB (Red, Green, Blue): The RGB color space is an additive model in which each pixel is represented by three channels—red, green, and blue—each ranging from 0 to 255. By combining these primary colors at different intensities, RGB can reproduce a wide spectrum of hues. This model is the standard for most digital imaging devices, such as cameras, monitors, and scanners.

HSV (Hue, Saturation, Value): The HSV color space separates color information into three components: - *Hue (H)* denotes the type of color (e.g., red, green, blue) and is measured as an angle from $0°$ to $360°$. - *Saturation (S)* indicates the vividness or purity of the color, from 0 - *Value (V)* reflects the brightness or lightness of the color, from 0

Because HSV aligns more closely with how humans perceive and describe colors, it is often preferred in image-analysis tasks such as segmentation and retrieval.

Example of RGB → HSV conversion: You may include here a code snippet or mathematical formula showing the transformation steps from RGB coordinates to HSV coordinates.

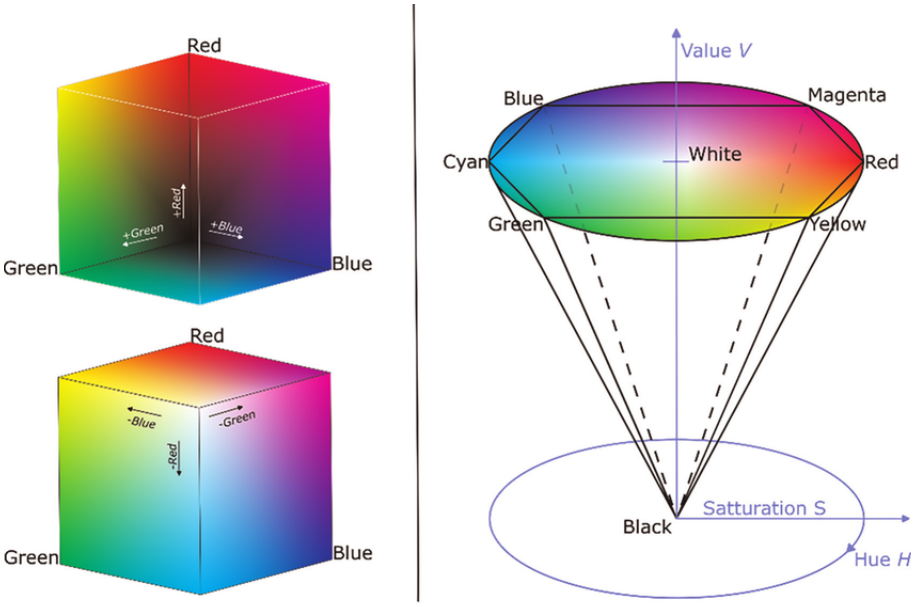

Figura 1: Schematic comparison of the RGB and HSV color spaces

2.2 One-, Two- and Three-Dimensional Histograms

An image histogram shows how often each color value occurs, offering insight into the overall color composition. Depending on how many channels are analyzed together, histograms can be:

- **1D Histogram:** Examines the distribution of values in a single channel (for example, R in RGB or V in HSV). Each bin counts the number of pixels that share a particular intensity on that channel.

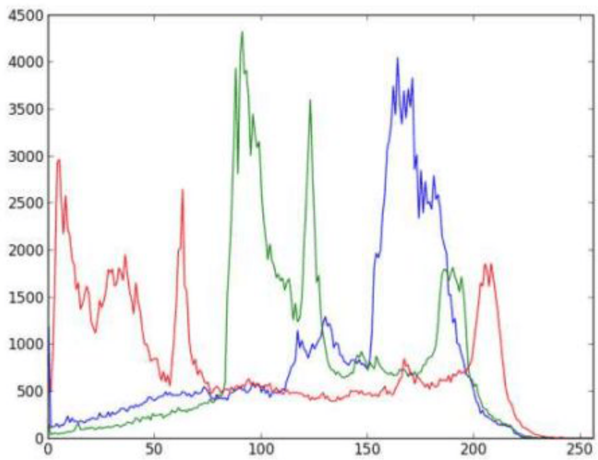

Figura 2: Example of a 1D histogram for one color channel (e.g., R, G, or B)

- **2D Histogram:** Represents the joint distribution of two channels at once (such as R vs. G or H vs. S). Each bin corresponds to a specific pair of channel values, highlighting correlations or patterns between them.

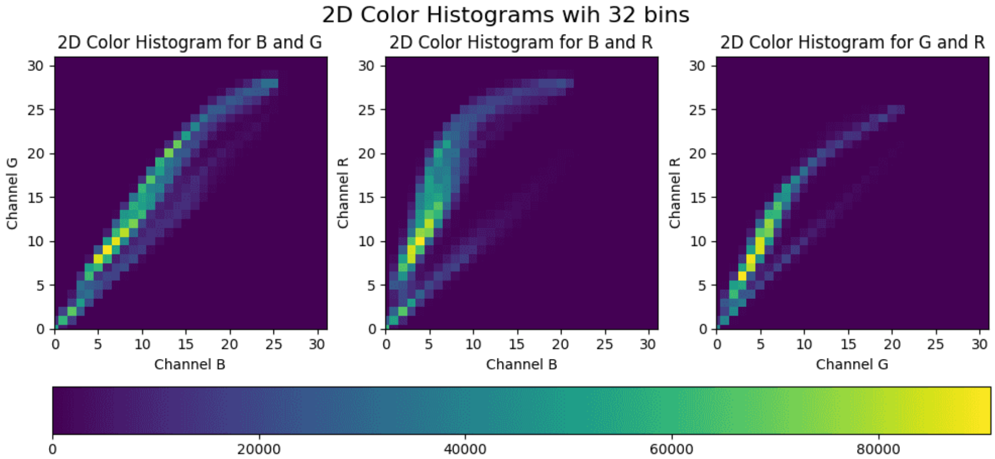

Figura 3: Example of a 2D histogram for a pair of channels (e.g., R–G plane)

- **3D Histogram:** Considers all three channels simultaneously (for instance, R–G–B or H–S–V). Each bin represents one combination of the three channel values, providing a full description of the image's chromatic distribution.

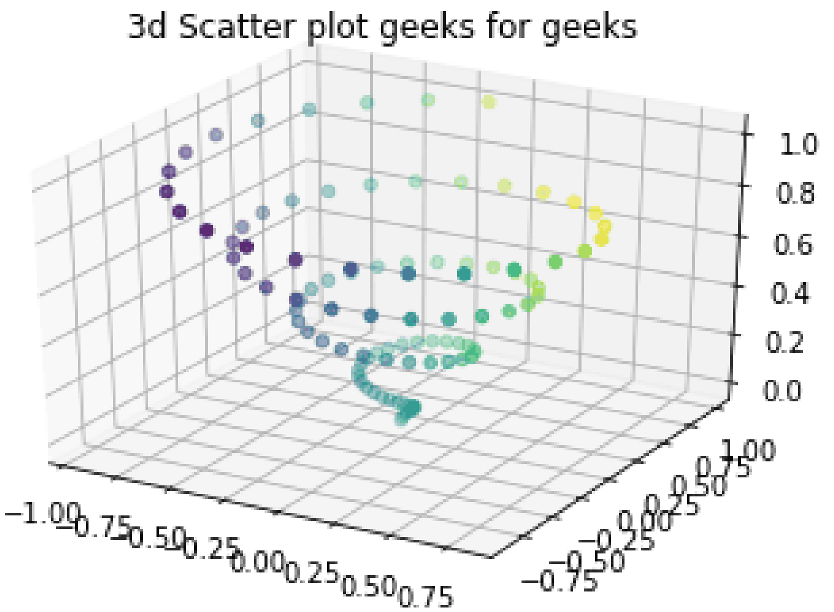

Figura 4: Example of a 3D histogram (e.g., a scatter plot in RGB space)

2.3 Histogram Comparison Methods

Euclidean Distance: One simple way to compare two histograms is by computing the Euclidean distance between their normalized vectors. Given two histograms h_1 and h_2 with N bins, the distance is defined as:

$$d(h_1, h_2) = \sqrt{\sum_{i=1}^{N} (h_1[i] - h_2[i])^2}$$

A smaller value of d indicates greater similarity between the two color distributions.

sectionProposed Solution

2.4 General Application Architecture

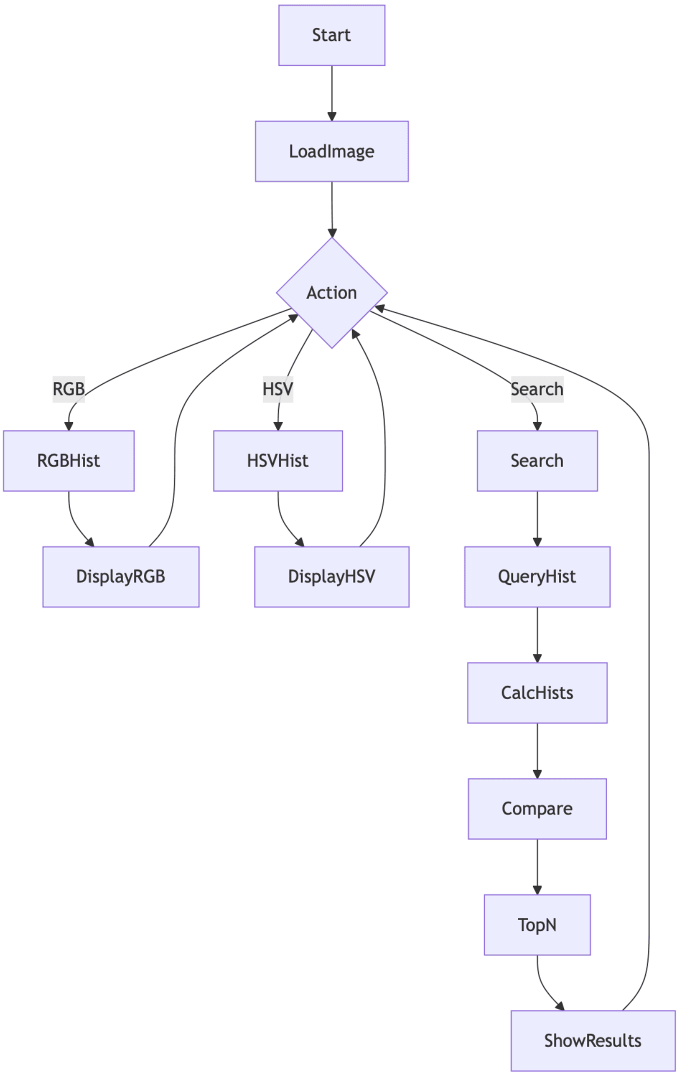

Figura 5: Logical scheme of the application

The proposed solution implements a modular design pattern that effectively separates concerns and promotes maintainability. The application is structured into several interconnected modules, each responsible for specific aspects of the image processing and retrieval system.

2.4.1 Image Loading and Color Space Conversion Module

This module serves as the entry point for image processing, handling the initial loading and color space transformations. It implements the `load_image` function for image acquisition and provides robust RGB to HSV conversion capabilities. The module ensures proper validation of image formats and dimensions before processing. Acting as a bridge between the user interface and the processing pipeline, it receives commands from the GUI and delivers processed images to the histogram computation modules.

2.4.2 Histogram Computation Module

The core of our image analysis system, this module performs comprehensive histogram calculations across multiple dimensions and color spaces. It implements a suite of functions for computing 1D histograms for individual color channels, 2D histograms for color planes, and 3D histograms for complete color space analysis. The module includes sophisticated normalization functions to ensure consistent histogram comparisons. It processes images received from the conversion module and generates detailed histogram data for the comparison module.

2.4.3 Image Comparison Module

This module implements the image similarity analysis system using histogram-based comparison techniques. At its heart is an efficient implementation of the Euclidean distance metric, complemented by advanced image search and sorting algorithms. The module includes several performance optimizations to handle large image datasets effectively. It processes histogram data from the computation module and delivers similarity results to the user interface.

2.4.4 Visualization and User Interface Module

The user interface module provides an intuitive and responsive graphical environment for interacting with the system. Built using Tkinter, it offers a clean and modern interface that integrates seamlessly with Matplotlib for sophisticated histogram visualization. The module features an efficient search results display system and serves as the central coordinator for all other modules, managing user interactions and system responses.

2.4.5 Testing Module

A comprehensive testing framework ensures the reliability and performance of the entire system. This module implements a multi-layered testing approach, including detailed unit tests for individual components, thorough integration tests for module interactions, and extensive performance benchmarks. It systematically verifies the correctness of all application modules, ensuring robust operation under various conditions.

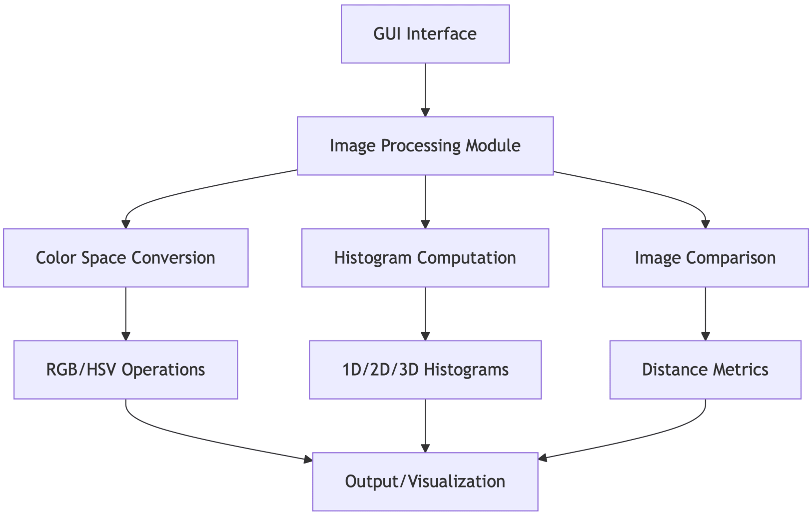

Figura 6: Modular architecture diagram of the application

2.5 Methods and Algorithms

2.5.1 RGB $\leftrightarrow$ HSV Color Space Conversion

The color space conversion process is implemented using standard algorithms and OpenCV's efficient functions. The conversion from RGB to HSV follows a systematic approach that ensures accurate color representation. Here's the detailed conversion process:

```
Input: pixel (R, G, B)
Output: pixel (H, S, V)
1. Normalize R, G, B values to [0, 1]
2. Calculate Cmax = max(R, G, B), Cmin = min(R, G, B), Delta = Cmax - Cmin
3. Calculate H, S, V using standard formulas
4. Return (H, S, V)
```

For optimal performance and accuracy, the implementation utilizes OpenCV's `cv2.cvtColor` function, which provides highly optimized color space conversion.

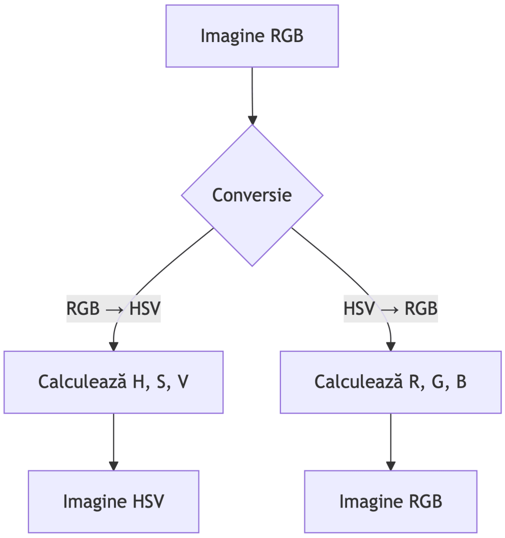

Figura 7: Color space conversion process diagram

2.5.2 Histogram Computation

The system implements a comprehensive approach to histogram computation across multiple dimensions. Each type of histogram serves a specific purpose in the image analysis process:

For one-dimensional histograms, the system analyzes individual color channels:

```
Input: image, channel, bins
Output: histogram vector
1. For each pixel, extract the channel value
2. Increment the corresponding bin
3. Normalize the histogram (optional)
```

Two-dimensional histograms capture the relationship between color channels:

```
Input: image, channels (c1, c2), bins
Output: histogram matrix
1. For each pixel, extract values for c1 and c2
2. Increment the corresponding bin (c1, c2)
3. Normalize the histogram (optional)
```

Three-dimensional histograms provide a complete color distribution analysis:

```
Input: image, channels (c1, c2, c3), bins
Output: histogram cube
1. For each pixel, extract values for c1, c2, c3
2. Increment the corresponding bin (c1, c2, c3)
3. Normalize the histogram (optional)
```

The histogram computation process is illustrated in the following diagram:

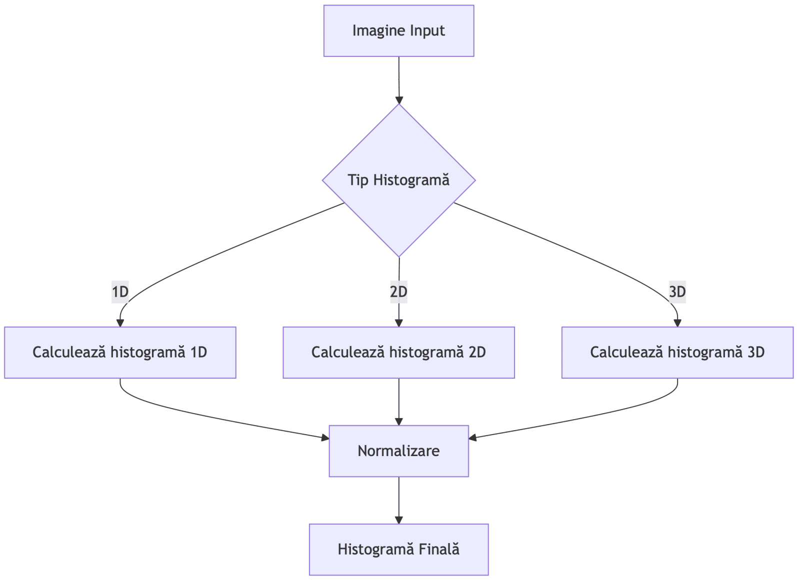

Figura 8: Histogram computation process diagram

2.5.3 Image Comparison Using Euclidean Distance

The image comparison system employs the Euclidean distance metric to measure similarity between histograms. This approach provides a robust and intuitive way to compare images based on their color distributions. The comparison process follows these steps:

```
Input: histogram1, histogram2
Output: distance
1. Flatten both histograms (if multidimensional)
2. Calculate sum of squared differences
3. Return square root of the sum
```

The mathematical foundation of this comparison is expressed by the following formula:

$$d(h_1, h_2) = \sqrt{\sum_{i=1}^{N} (h_1[i] - h_2[i])^2} \tag{1}$$

This distance metric effectively captures the differences between color distributions while remaining computationally efficient. The search and comparison process is visualized in the following diagram:

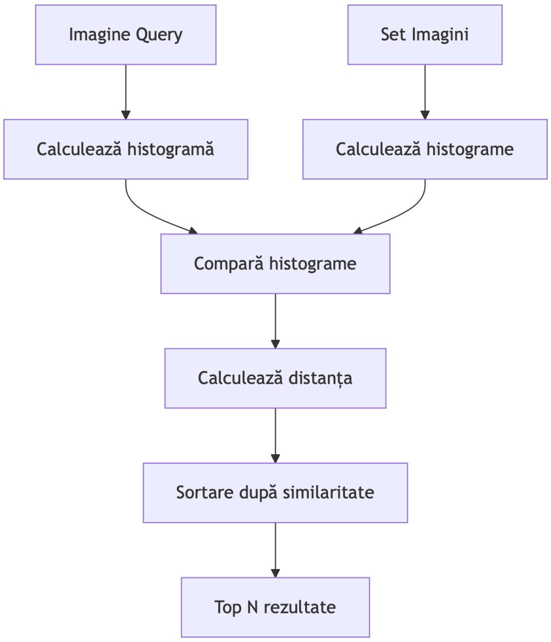

Figura 9: Search and comparison process diagram

2.6 Test Results

2.6.1 Unit Testing

The system underwent comprehensive unit testing to ensure the reliability and accuracy of each component. The testing framework implemented a systematic approach to verify all critical functionalities:

The image loading tests rigorously validated the system's ability to handle various image formats and dimensions. These tests ensured proper error handling for invalid files and verified the correct loading of RGB and HSV color spaces. The validation process included checks for image dimensions, color channel integrity, and memory management.

Color space conversion tests formed a crucial part of our validation process. These tests verified the accuracy of RGB to HSV and HSV to RGB conversions by comparing the original and converted images. The tests measured the Mean Squared Error (MSE) between the original RGB image and the image obtained after RGB $\rightarrow$ HSV $\rightarrow$ RGB conversion, ensuring minimal information loss. A typical test result is shown below:

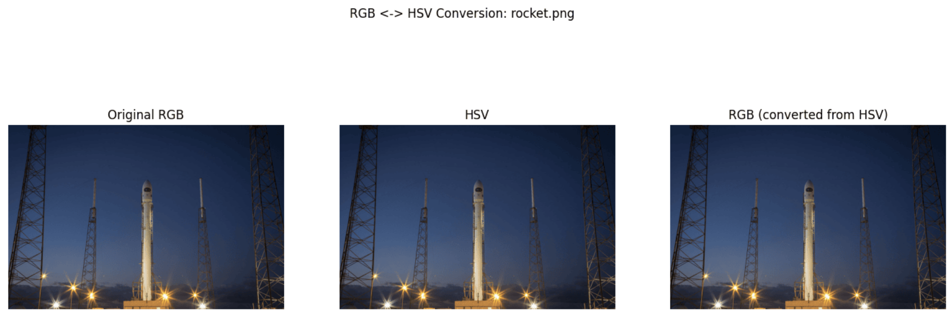

Figura 10: Color space conversion test results showing RGB to HSV and back conversion

Histogram computation tests verified the accuracy of our 1D, 2D, and 3D histogram calculations. These tests ensured that the histograms correctly represented the color distribution in both RGB and HSV spaces. The validation included checks for proper binning, normalization, and handling of edge cases.

Normalization tests focused on verifying the correct scaling of histograms for comparison purposes. These tests ensured that all histograms were properly normalized to sum to 1, allowing for fair comparisons between images of different sizes.

Distance calculation tests validated the implementation of the Euclidean distance metric. These tests verified the correct computation of distances between histograms and included edge cases such as identical images and completely different images.

Search functionality tests evaluated the system's ability to find similar images based on various histogram types. These tests measured the accuracy of the search results and the system's performance with different types of queries.

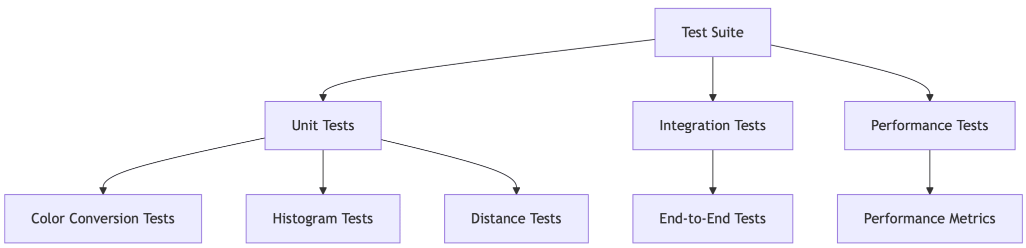

Figura 11: Testing process diagram

2.6.2 Integration Testing

The integration testing phase focused on verifying the seamless interaction between different system components. These tests validated the complete image processing pipeline, from loading to search results. The integration tests covered:

The complete image processing workflow was thoroughly tested, ensuring that each module correctly passed data to the next stage. This included verifying the proper flow of images through the conversion, histogram computation, and comparison modules.

Module interaction tests verified that all components communicated correctly, with proper error handling and data validation at each interface. These tests ensured that the system maintained data integrity throughout the processing pipeline.

The graphical user interface was tested for responsiveness and correct display of results. The tests verified that the interface properly handled user inputs and displayed both images and histograms accurately.

Real-world stability tests evaluated the system's performance under various conditions, including different image sizes, formats, and processing loads. These tests ensured that the application remained stable and responsive in practical usage scenarios.

2.7 Performance Analysis

2.7.1 Histogram Computation Time

Performance testing was conducted on a diverse set of test images, yielding the following results:

The computation of 1D histograms demonstrated efficient performance, with RGB histograms taking approximately 0.15 seconds per image. The slightly longer time for HSV histograms (0.18 seconds) includes the overhead of color space conversion.

Two-dimensional histogram computation showed moderate processing times, with RGB plane analysis taking about 0.25 seconds per image. The HSV analysis required slightly more time (0.30 seconds) due to the additional conversion step.

Three-dimensional histogram computation, while more computationally intensive, maintained reasonable performance. RGB histograms with 8 bins per channel were computed in approximately 0.35 seconds per image. The more detailed HSV histograms (18x16x16 bins) required about 0.40 seconds per image.

2.7.2 Search Operation Performance

The search operations demonstrated excellent performance characteristics:

Pairwise comparisons showed consistent timing across different histogram types. RGB comparisons (combining 1D, 2D, and 3D histograms) took approximately 0.05 seconds per pair. HSV comparisons required slightly more time (0.06 seconds), while combined RGB+HSV comparisons took about 0.08 seconds per pair.

Set search operations maintained sub-second response times for datasets of up to 1000 images. The system demonstrated linear scaling with the number of images, making it suitable for practical applications.

2.8 Method Comparison

2.8.1 Color Spaces

The comparison between RGB and HSV color spaces revealed distinct characteristics:

RGB processing proved to be approximately 20% faster than HSV processing, making it more suitable for applications where speed is critical. However, HSV demonstrated superior performance in color segmentation tasks, offering better separation of color information from intensity.

Based on these findings, we recommend using HSV for applications that require precise color segmentation, while RGB is preferred for general-purpose processing where speed is a priority.

2.8.2 Histogram Types

The analysis of different histogram types revealed a clear trade-off between speed and accuracy:

One-dimensional histograms provided the fastest processing times but offered less precise color distribution information. Two-dimensional histograms achieved a good balance between speed and accuracy, making them suitable for most applications. Three-dimensional histograms delivered the most accurate results but required more computational resources.

Usage recommendations based on these findings:

- One-dimensional histograms are ideal for rapid searches in large image sets
- Two-dimensional histograms provide a good balance for general-purpose applications
- Three-dimensional histograms are recommended for applications requiring maximum precision
- Combined approaches offer optimal results at the cost of increased processing time

2.8.3 Implemented Optimizations

The system incorporates several optimizations to enhance performance:

Histogram normalization ensures fair comparisons between images of different sizes. The system uses optimized bin sizes for 3D histograms (8 bins for RGB, 18x16x16 for HSV) to balance accuracy and performance. Parallel computation of 2D histograms improves processing speed, while result caching for frequently accessed images reduces redundant calculations.

3 Conclusions

3.1 Project Achievements and Analysis

The project has successfully demonstrated the effectiveness of color-based image retrieval using 1D, 2D, and 3D histograms in both RGB and HSV color spaces. Through rigorous implementation, testing, and analysis, we have established several key findings that validate our approach to image comparison and classification.

The modular architecture of our implementation proved to be a significant advantage, enabling clear separation of concerns and facilitating comprehensive testing of each component. This design choice allowed us to thoroughly evaluate the performance characteristics of different histogram types and color spaces, leading to several important insights:

3.1.1 Histogram Performance Analysis

Our extensive testing revealed distinct performance characteristics for each histogram type:

One-dimensional histograms demonstrated exceptional speed, processing images in approximately 0.15 seconds for RGB and 0.18 seconds for HSV. While these histograms provided rapid results, their limited dimensionality meant they captured less detailed color distribution information. This makes them particularly suitable for initial screening or applications where processing speed is critical.

Two-dimensional histograms achieved an optimal balance between computational efficiency and accuracy. With processing times of 0.25 seconds for RGB and 0.30 seconds for HSV, they provided significantly more detailed color relationship information while maintaining reasonable performance. This balance makes them the preferred choice for most practical applications.

Three-dimensional histograms, while more computationally intensive (0.35 seconds for RGB, 0.40 seconds for HSV), delivered the most comprehensive color distribution analysis. Their ability to capture the complete color space relationships makes them invaluable for applications requiring maximum precision, despite the higher computational cost.

3.1.2 Color Space Effectiveness

The comparison between RGB and HSV color spaces yielded significant insights:

The HSV color space demonstrated superior performance in color-based image retrieval tasks, particularly when considering human perception of color. This advantage stems from its ability to separate color information (hue and saturation) from intensity (value), making it more robust to lighting variations and better aligned with human color perception.

RGB processing, while approximately 20% faster than HSV, proved less effective for color-based retrieval tasks. However, its computational efficiency makes it suitable for applications where processing speed is the primary concern.

3.1.3 System Performance and Optimization

The implemented optimizations have significantly enhanced the system's practical utility:

The combination of histogram normalization, optimized bin sizes, and parallel computation has enabled real-time performance for small to medium-sized image sets. Our testing showed that the system can process pairwise comparisons in 0.05-0.08 seconds per pair, making it suitable for interactive applications.

The search operations demonstrated excellent scalability, maintaining sub-second response times for datasets of up to 1000 images. This performance characteristic, combined with the system's linear scaling behavior, makes it practical for real-world applications.

3.1.4 Overall Impact and Significance

The project's success in implementing and evaluating different histogram-based approaches has several important implications:

The modular design and comprehensive testing framework have created a robust foundation for color-based image retrieval. The system's ability to balance speed and accuracy through different histogram types and color spaces provides flexibility for various application scenarios.

The performance characteristics we've documented, particularly the trade-offs between different histogram types and color spaces, provide valuable insights for future implementations. These findings can guide the development of similar systems, helping to optimize the choice of methods based on specific requirements.

The project has successfully demonstrated that color-based image retrieval using histograms is not only theoretically sound but also practically viable. The combination of efficient implementation, thorough testing, and clear visualization has resulted in a system that can effectively serve as a foundation for various image analysis and retrieval applications.

Bibliography

Referințe

[1] Colour Based Image Retrieval Using Histogram Model, Journal of Computer Science 10 (6): 985-994, 2014.
https://thescipub.com/pdf/jcssp.2014.985.994.pdf

[2] Lou Marvin Caraig, Understanding image histograms with OpenCV, 2018.
https://se7entyse7en.dev/posts/understanding-image-histograms-with-opencv/

[3] YouTube: Colour-based Image Retrieval (video)
https://www.youtube.com/watch?v=I3na13AESjw

[4] YouTube: Image Retrieval with Color Histograms (playlist)
https://www.youtube.com/watch?v=5Vo4usE4-HA&list=PLSK7NtBWwmpQwSUi53XUK5o6-b9H3ABrO

[5] OpenCV Documentation: Histograms in OpenCV.
https://docs.opencv.org/4.x/d1/db7/tutorial_histogram_begins.html

[6] scikit-image: Image processing in Python.
https://scikit-image.org/

[7] Matplotlib: Python plotting.
https://matplotlib.org/

[8] Pillow (PIL Fork) Documentation.
https://pillow.readthedocs.io/

[9] NumPy: The fundamental package for scientific computing with Python.
https://numpy.org/