Digital Image Processing Techniques: Interpolation, Classification, and Change Detection

Abdullah Durum
2448330

Cas Carree
2704930

***Abstract*—This paper explores three fundamental aspects of digital image processing: image interpolation techniques, hue histogram-based image classification, and image change detection using image subtraction. We compare bilinear and bicubic interpolation methods, analyze RGB to HSI color space conversion for desert/forest classification based on the hue histograms, and implement image subtraction techniques for change detection. Our findings demonstrate the trade-offs between different interpolation methods, the effectiveness and limitations of hue-based classification, and the importance of parameter tuning in change detection algorithms.**

***Index Terms*—Interpolation, image subtraction, histogram-based classification**

I. Image Interpolation Techniques

A. Background

Interpolation is the process of estimating intermediate values between discrete samples. This fundamental technique plays a critical role in digital image processing, enabling tasks such as image resizing and spatial distortion correction [1]. Common interpolation techniques include bilinear and bicubic interpolation, both of which utilize pixel information from neighboring pixels to predict the value of unknown pixels. Bilinear interpolation employs a 2x2 neighborhood surrounding the target pixel to compute its value, using a weighted average of the four surrounding pixels [2]. This technique is relatively simple and efficient, making it suitable for real-time applications. On the other hand, bicubic interpolation extends this concept by incorporating a 4x4 neighborhood, thereby utilizing sixteen surrounding pixels. This approach assigns higher weights to closer pixels, leading to smoother and more visually appealing results compared to bilinear interpolation, albeit at the cost of increased computational complexity [2].

B. Methodology

Our implementation involved several critical steps:

- Conversion from BGR to RGB color space using OpenCV.
- Specification of the preferred interpolation method.
- Calculation of the original image dimensions to facilitate resizing.
- Resizing the image with cv2's resize function while maintaining aspect ratio.
- Computation of the rotation matrix using cv2's getRotationMatrix2D.
- Application of the warpAffine function to obtain the final transformed image.

C. Results and Discussion

To evaluate the performance of the interpolation methods, we utilized Mean Squared Error (MSE) as our quality metric. Due to the discrepancies in dimensions between the output and the original images, we tested two methods for size adjustment:

1) Cropping Method:
   - First image: MSE of 1439 (bilinear) and 1349 (bicubic)
   - Second image: MSE of 550 (bilinear) and 580 (bicubic)
2) Padding Method:
   - First image: MSE of 862 (bilinear) and 806 (bicubic)
   - Second image: MSE of 269 (bilinear) and 284 (bicubic)

The padding method was found to be superior as it mitigated issues associated with black edges in the cropped output. The results indicate no clear distinction in performance; however, bicubic interpolation yielded better results for the first image, while bilinear performed better on the second. This discrepancy may arise from the different characteristics of the images, particularly the presence of well-defined edges in the second image. However, this hypothesis should be tested, which there wasn't sufficient time for anymore.

D. Limitations

While MSE is computationally efficient, providing a good baseline, it has notable limitations, including:

- The assumption of pixel independence, which does not reflect true image characteristics [3].
- Ignoring spatial relationships between pixels, which is crucial for image perception [3].
- The focus on absolute differences, neglecting structural variations that the human eye perceives [3].

Therefore, if an image is shifted by one pixel the MSE would be high while humans probably wouldn't even notice.

Given these limitations, alternative metrics such as the Structural Similarity Index (SSIM) could provide more comprehensive insights by incorporating structural information and demonstrating robustness against noise [4].

II. Object Recognition

A. Background

The HSI color model represents colors through three components: hue (H), saturation (S), and intensity (I), making it highly useful for image processing. Hue defines the type of color on a 0-360 degree scale, saturation indicates the color's purity, and intensity reflects brightness [5]. This model separates brightness from color information, aligning it more closely with human vision and making it effective for applications in varying lighting conditions and detailed image analysis [5].

The advantages of HSI are substantial:

- Enhanced image analysis through the separation of brightness from color, simplifying tasks such as edge detection [5].
- Improved image editing capabilities, allowing for adjustments in brightness and contrast without affecting color integrity [5].
- Robustness against varying lighting conditions, making it effective for applications such as facial recognition.
- Precision in color adjustments, enabling detailed modifications without altering brightness [5].

However, HSI also presents challenges:

- The computational complexity of converting from RGB to HSI can hinder real-time processing [5].
- The model's sensitivity to low saturation levels may result in inaccuracies for dull colors [5].
- Limited software support, as many tools are optimized for RGB, necessitating additional effort to utilize HSI [5].

B. Methodology

Our approach for converting RGB to HSI consisted of the following steps:

- Normalization of RGB values by dividing by 255, converting each component to a range between 0 and 1.
- Calculation of HSI values using established formulas [9], ensuring a proper representation of color information.

We generated a histogram for the hue channel using the numpy library's histogram function. The selection of bin sizes was critical, as too many bins could amplify noise effects, while too few would obscure important data distributions. Testing different bin values (30, 256) revealed minimal variation in results, as shown in the provided code, suggesting that the chosen bin size did not significantly impact the classification outcomes.

C. Classification Approach

The classification process involved several key steps:

- Calculation of Kullback-Leibler (KL) divergence from the hue histograms.
- Implementation of a nearest neighbor classification strategy based on the minimum KL divergence.

D. Results and Limitations

The hue channel proved sufficient for basic classification tasks due to its resilience against illumination changes [6]. However, several limitations were identified:

- Inability to differentiate between objects of similar colors, such as red apples and strawberries [7].
- Challenges encountered when classifying multi-colored images [7].
- Lack of spatial information in the classification process, which could limit accuracy [7].

Furthermore, in this approach, only the nearest neighbor was utilized due to the limited size of the dataset. However, for a more robust algorithm, k-nearest neighbors can be used, even though this requires hyperparameter tuning, which can be time-consuming. Due to the abovementioned limitation for more complex datasets, employing a combination of features, including color, texture, and shape, similar to the QBIC system, may yield improved classification results [7].

III. Image Change Detection

A. Theoretical Background

Image subtraction is a technique that computes the pixel-wise difference between two images. It is particularly useful for highlighting changes within the images [8]. When there is no change, the resulting difference should be close to zero, but variations may arise from noise or lighting conditions [8]. To obtain only the meaningful change in the image, thresholding can be utilized. Thresholding converts an image into a binary image based on its intensity values. For this, you need to select a threshold value that acts as a boundary separating pixels with significant changes from those due to noise and lighting [9].

B. Implementation

Separate approaches were developed for both grayscale and RGB images, allowing for adjusting the weights of the colors in the RGB case. This was done to give the red channel more weight as the distinguishing object was a red book.

1) Grayscale Implementation:
   - Conversion of images to grayscale using cv2's cvtColor function to prevent working with color images, which can lead to errors.
   - Calculation of absolute differences using cv2's absdiff function.
   - Application of thresholding to highlight significant changes, with the optimal threshold value determined to be 75 (within a tested range of 30-100).
   - Generation of the final output using cv2's bitwise_and function to enhance the visibility of changes.
2) RGB Implementation:
   - Processing of each RGB channel separately to detect changes.
   - Implementation of enhanced weighting for the red channel.

- Application of a threshold value of 50 to effectively highlight changes.
- Utilization of morphological operations open and close with an 11x11 kernel to refine the mask and eliminate noise.
- Generation of the final output using cv2's bit_wise_and function to enhance the visibility of changes.

C. Results and Discussion

The implemented image change detection approach demonstrated effectiveness in identifying changes across both grayscale and RGB images. The optimal configurations included:

- A threshold value of 75 for grayscale images, successfully highlighting significant changes.
- A threshold value of 50 for RGB images, effectively revealing differences.
- Utilization of an 11x11 kernel for morphological operations, enhancing overall change detection.

While the method proved effective for the tested images, further hyperparameter tuning of the threshold might improve it even further. Moreover, the RGB approach's generalization capacity will be limited due to the focus on the red channel. However, this can be mitigated by changing the relative weight of the different channels according to the new target object or set to 1/3 when the target isn't known.

IV. Conclusion

This study highlights the complexities and trade-offs inherent in various image processing techniques. Each method demonstrated its effectiveness in specific contexts, yet limitations were identified that could inform future developments. Further research could focus on:

- A comprehensive comparison of interpolation methods across diverse image types to establish more conclusive performance benchmarks.
- Integration of multiple feature sets for more robust image classification approaches, potentially enhancing accuracy and reliability.
- Development of more generalized processing techniques for RGB images to enhance application across diverse scenarios.

References

[1] A. A. K. J. M. H. M. A. Z. K. N. M. Donya Khaledyan, "Low-Cost Implementation of Bilinear and Bicubic Image Interpolation for Real-Time Image Super Resolution".

[2] Cambridge in Colour, "cambridgeincolour.com," [Online]. Available: https://www.cambridgeincolour.com/tutorials/image-interpolation.htm. [Accessed: 04-Nov-2024].

[3] Y. Lu, "The Level Weighted Structural Similarity Loss: A Step Away from MSE," 2019.

[4] Restack, "restack.io," 26-Oct-2024. [Online]. Available: https://www.restack.io/p/similarity-search-answer-structural-similarity-ssim-cat-ai.

[5] GeeksforGeeks, "geeksforgeeks.org," 10-Jun-2024. [Online]. Available: https://www.geeksforgeeks.org/what-is-hsi-color-space/.

[6] G. D. Finleyson, "researchgate.net," Apr-2002. [Online]. Available: https://www.researchgate.net/publication/2871537$_{H}ue_{T}hat_{i}s_{I}nvariant_{t}o_{B}rightne$ 41.

[7] S.-Y. C. Ya-Chun Cheng, "sciencedirect.com," 01-Aug-2003. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0262885603000696.

[8] R. E. W. Rafael C. Gonzalez, "Digital Image Processing Second Edition," in Digital Image Processing Second Edition, 2002, pp. 110-112.

[9] A. R, "scaler.com," 18-Aug-2023. [Online]. Available: https://www.scaler.com/topics/thresholding-in-image-processing/.