Unique Identification Through Colour Extraction (3D Model)

[1]Shivam Sharma
[1]School of Computer Science and Engineering
Galgotias University
Greater Noida, India
[1]shivam231198@gmail.com

[2]Vaishnavi Rai Sahu
[2]School of Computer Science and Engineering
Galgotias University
Greater Noida, India
[2]vaishnavirai08@gmail.com

[3]Saurav Jhanji
[3]School of Computer Science and Engineering
Galgotias University
Greater Noida, India
[3]sauravjhanji01@gmail.com

abstract— This proposal is about identifying baggage uniquely by its features and colors. We have seen a lot of identifications of a person by their facial features but we have never worked on identifying materialized objects which do not contain many unique features. So we are developing a product by which we will be able to take images of the target as input and then we will extract all the possible features and all the colors out of it. Then we will apply our model on some n number of videos and after the complete analysis, we will be able to tell that which video file that object (bag in our case) was detected. It will also provide the exact time at which the object was detected with a precision of a minute. The application can further be used and integrated with identification systems or security systems, like tracking criminal's bags or finding lost bags. This application will further be capable of integrating with any other technologies like IoT devices or mobile apps etc.

Index Terms—*Computer Vision, Deep Learning, Image Processing, Python, Storing features.*

I. INTRODUCTION

With the vast amount of multimedia data generated and transmitted in digital formats, content-based data management has emerged as an important research area. A significant amount of research effort has been elaborated upon addressing the problem of image management, such as content-based image retrieval (CBIR), image classification, and also indexing and navigating through an image database.

Researchers follow the paradigm of representing images by their visual content, such as color, texture, and shape, and define proper similarity measures for each feature. Among them, the color feature is deemed the most common feature extracted for various image applications [1].

Color features are extracted to represent image color distributions and are often presented as color histograms. The traditional extraction methods rigidly partition the underlying color spaces into a fixed number of bins, each of which corresponds to a bin in the histogram. The extraction process is to place pixels into their closest color bin. The weight of a bin in turn denotes the percentage of pixels in an image belonging to that bin. Therefore, a color histogram can be thought of as a quantized color distribution of an image.

The major drawback of these traditional methods is that they do not take the image color distribution into consideration. In other words, disregarding the image color distribution, these methods use the same set of representative colors for each image. Therefore, a traditional method provides little adaptability to the color content of an image, and color features may be heavily distorted if a small number of bins are used. For instance, when one extracts color features from a sea image whose pixels are mostly deep blue, a finely quantized histogram would be inefficient to represent this feature. On the other hand, a coarsely quantized histogram would be inadequate to represent an image containing numerous flowers of different colors. As a result, the traditional color extraction method is not able to achieve a balance between expressiveness and compactness.

Compared with prior work in extracting color moments [2], [3], where only moments of each color channel are extracted to present color features, FC and VC do represent color features as color histograms rather than as the values of the first three moments. Moreover, the BQMP thresholding technique used in FC and VC is able to calculate color moments by treating the value of a pixel as an entity rather than calculating color moments in three separate channels.

As will be shown in our experimental results, the method of extracting color channel moments has restricted performance due to its extremely compact representation. In contrast, the proposed extraction scheme not only possesses the property of preserving color moments but also represents features as a color histogram, which is a more precise and scalable representation.

The resulting histograms from FC or VC will consist of different sets of colors due to the adaptive extraction. In [4], Rubner et al. proposed the earth mover's distance (EMD) to measure the distance between two distributions, which differs from the prior distance measures, such as the Minkowski-form distance, the intersection distance [5], and the quadratic-form distance [6], by eliminating the limitations that two histograms must have the same set of colors. Moreover, the EMD is designed to measure distances between distributions, which is indeed directly applicable to our case where the histograms can precisely represent image color distributions. The EMD method models the problem of measuring distances as the transportation problem in linear programming [7] and the distance is then determined by finding the optimal flows to transform one distribution to the other.

II. PROBLEM FORMULATION

The new color extraction method, 3d K-mean Clustering, proposed in this paper aims to extract color features according to the image color distribution. K-mean clustering as shown in Figure 1 attempts to preserve the image color distribution during the extraction process. If we treat the value of a pixel in an image as a random vector, the color distribution of this image will be equivalent to the probability distribution of this random vector. In view of this, we apply the percentage comparison of the colors in a single image of the 3d model to match with each side. This data hence found could also be stored in the database so that it can also be used in the future for comparison.

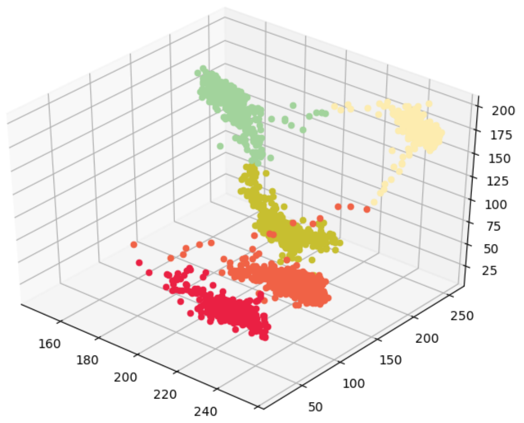


Figure 1. This figure shows the clustering of the colors of an image.

In many cases, it is desirable to reduce the image processing time, especially for supporting the upcoming applications where end-user devices vary significantly in terms of processing capability. In addition to improving the efficiency of measuring color feature distances by introducing CHIC, we also improve the efficiency of the color feature extraction process. The extraction process can be speeded up by reducing the number of memory reads, and in view of this, two orthogonal approaches are devised. The first is to add a swapping mechanism to the BQMP technique so as to place pixels belonging to the same cluster into a contiguous memory segment, which enables us to quickly locate a pixel cluster. The second is to modify the original BQMP technique so as to eliminate the need for a preliminary scan by some additional computation, which is preferable for hardware implementation because of the inherent concurrent property of hardware. The modification for hardware implementation justifies the feasibility of many possible applications on embedded devices where processing capability is often restricted.

It is noted that minimizing the distortion incurred in the extraction process can improve the performance of the subsequent various image applications, and we evaluate the meaningfulness of the new extraction methods by the application to CBIR. It was reported that the global color feature alone is adequate to classify landscape images into the sunset, forest, and mountain. Therefore, the database used in our experiments consists of images solely from landscape scenes. A query image is one of the images in the database and belongs to some image class. The meaningfulness of a color extraction method is measured by the average retrieval precision rate of those returning matches which fall into the correct image class of the query image. Our experimental results show that the proposed extraction methods can enhance the average retrieval precision rate by a factor of 25% over that of the binning method. Note that our intention is to compare the performance of the proposed color extraction scheme with that of the traditional method. Many existing techniques proposed towards a more general CBIR problem, such as fusing several low-level features together or using a user's feedback are in fact orthogonal to the technique that we want to advocate in this paper.

KMeans algorithm creates clusters based on the supplied count of clusters. In our case, it will form clusters of colors and these clusters will be our top colors. We then fit and predict on the same image to extract the prediction into the variable labels. Now moving this idea further into a 3D model where we want to identify an object in a 3d space. For example, a video that contains n numbers of image frames displaying after one another. In such a case, an object can be on any three-dimensional side. We offer to identify objects in such cases as well as make it possible to store such object features into JSON format form of Red, Green, and Blue colors. We are also taking the help of the HSV color measure in order to save color features in a set range of colors.

III. COLOR EXTRACTION

In this section, we will learn about extracting the colors from a single frame and then storing it into a database-compatible format.

Any image is formed from an array of RGB colors, we call it a pixel. Depending upon the size of that image the number of pixels is calculated. Now what's the problem in saving every single pixel value of an image is that it will result in giving too many pixels for a single image and it will get difficult to store each and every pixel value. Therefore, in solution to that, there is a need of creating a group of colors. That group can contain a max of seven colors which then becomes easy to store and use.

Creating a group is a challenge in itself as we know a single color can contain so many shades of itself so we cannot name shades of red color as red. As we can see in Figure 2.1 there are completely two different shades of red color and we cannot group both of these colors into the same group category as red. Therefore we need a different strategy to group colors and keep their key importance in order to identify the objects with accuracy.

Figure 2.1 A comparison between two shades of red color

So another way is that we can specify a range of a particular color and if the different shades come under the same range we can group them together. This way we will be able to solve the problem of storing each and every pixel of an image. But it's nearly impossible to create a range in RGB format. For this purpose, we can use another representation of color i.e. HSV format. HSV stands for Hue Saturation Value. As it can be seen in Figure 2.2 we get a range of colors all dependant on a single value Hue which ranges from 0 to 360. This way it becomes possible for us to create a range of a single color and store it.

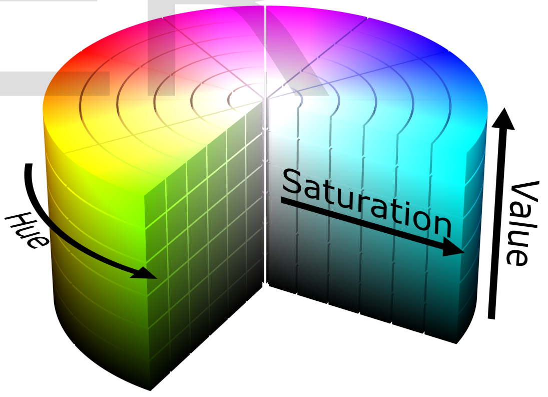
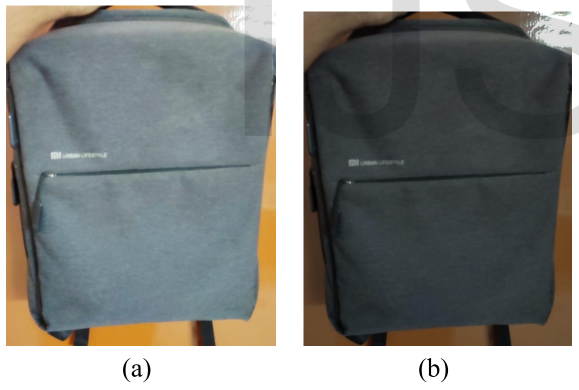


Figure 2.2 HSV color representation

The range we are creating of a single color is on the basis of the HSV value of a color. Let see the range of each and every value in HSV. Hue "H" has a range from 0 to 360, Saturation "S" has a range from 0 to 100, and Value "V" has a range from 0 to 100. Now after getting a pixel value and converting it into an HSV value we are creating an upper and lower limit of the HSV value which forms up in the form of a range of a pixel.

In order to set that limit, we are subtracting and adding a specific threshold to each value of the HSV value of that pixel. Now, this threshold value is decided on the basis of a number of experiments done in order to get such a value by which the color or that pixel can be identified in different conditions of lights, resolution, brightness, and dullness.

The threshold values for creating a range are “10”, “20” and “15” respectively for “H”, “S” and “V”. For example let’s say we have an HSV value (30, 40, 50) of a pixel from a single frame then our upper and lower limits will be (40, 60, 65) and (20, 20, 35) respectively. Someone can ask for proof and reason for such threshold values but these values have been decided on the basis of experiments alone. Due to this range of a pixel, we are able to identify that same color in dark and light mode. See Figure 3.1 where (a) represents a bag in bright light, (b) represents the bag in dark light, and (c) represents the bag in normal light. The range we have created will be able to detect all three bags representing different lighting conditions. This is possible because of the HSV ‘Value’ attribute which controls the brightness and darkness in an image.


(a) (b)

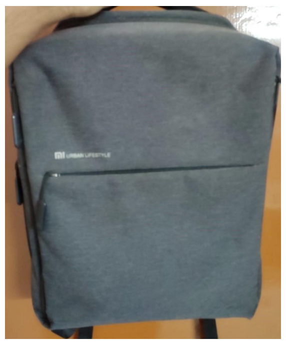

(c)

Figure 3.1 Bag image in different lighting condition: (a) bright, (b) dark, and (c) normal.

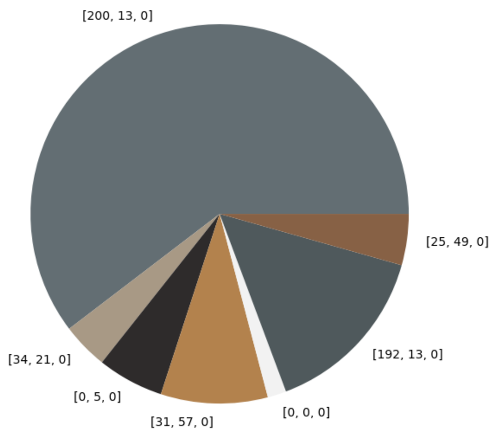


Figure 3.2 Bag image color distribution with respective HSV values.

After creating the range of all the pixels of an image we apply the KMeans Clustering algorithm to get the overall 7 colors from the image. Now with respect to this, we also store the percentage of each color. As it can be seen in Figure 3,2 we have a color distribution of image 3.1 (c) where the colors [200, 13, 0] and [192, 13, 0] are showing the majority of the image. Now if we will calculate the range of these 2 HSV values we will get [190, 0, 0] - [210, 33, 15] and [182, 0, 0] - [202, 33, 15] respectively. If we will merge these 2 ranges we will get the union [182, 0, 0] - [210, 33, 15]. So any color of the image that comes under this range with the same color concentration will be considered a match.

Our proposal also promises to store these range in a format that it can be operated on a server with API calls. So these values can be stored in the form of a JSON data format. It is very commonly used in API development and can be accessed easily. JSON format contains an object-like structure covered with curly braces ‘{‘ ‘}’ where we have key-value pare. That means we can access a value with the help of the key and when pass returns the value. Now think about an array of length 3 where each position storing the HSV value respectively representing a pixel. Now the array of such pixel array can be stored and access using key-value pair in JSON format. Similarly, the range of each pixel HSV value can be stored. For this we will make two objects by the name “upper limit” and “lower limit” storing the upper range and lower range respectively. This way it can be done by each pixel of the image and the main object is created with further can be used in server-side coding to operate in API calls.

IV. CONCLUSION

An adaptive color feature extraction scheme by preserving color distributions up to the third moment is proposed. It is noted that minimizing the distortion incurred in the extraction process can enhance the accuracy of the subsequent various image applications, and we evaluate the meaningfulness of the new extraction methods by the application to content-based image retrieval. The experimental results have shown that the new extraction methods can achieve a substantial improvement over the traditional color feature extraction methods. Further, the efficiency issues are considered in this paper. In addition to devising an efficient and effective distance measure, two orthogonal approaches to speeding up the feature extraction process are given, and one of them aims to take advantage of the concurrent properties of hardware.

In this paper, we focus on applying the new extraction methods to the global color feature extraction. However, as commonly used in practice, an image is usually divided into several subblocks and the regional color features are extracted. In this case, our extraction methods and distance measures will be even more preferable to the traditional binning methods because the color content in a subblock tends to be dominated by only a few colors. Our extraction methods are able to adaptively extract those dominated colors while the binning methods do not provide such adaptability. Moreover, the VC extraction method, which terminates the extraction process based on the heterogeneity of the pixel values, is applicable to this case to achieve a balance between expressiveness and compactness.

REFERENCES

[1] F. Long, H. Zhang, and D. D. Feng, “Fundamentals of content-based image retrieval,” in Multimedia Information Retrieval and Management Technological Fundamentals and Applications. New York: Springer-Verlag, 2003.

[2] M. K. Mandal, T. Aboulnasr, and S. Panchanathan, “Image indexing using moments and wavelets,” IEEE Trans. Consum. Electron., vol. 42, no. 3, pp. 557–565, Aug. 1996.

[3] M. Stricker and M. Orengo, “Similarity of color images,” in Proc. Storage and Retrieval for Image and Video Databases (SPIE), 1995, pp. 381–392.

[4] Y. Rubner, C. Tomasi, and L. J. Guibas, “A metric for distributions with applications to the image.

[5] M. J. Swain and D. H. Ballard, “Color indexing,” Int. J. Comput. Vis., vol. 7, pp. 11–32, 1991.

[6] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, “Query by image and video content: The QBIC system,” IEEE Computer, vol. 28, no. 9, pp. 23–32, Sep. 1995.

[7] F. S. Hiller and G. J. Liberman, Introduction to Mathematical programming. New York: McGraw-Hill, 1990.