**Dense Color Moment: A New Discriminative Color Descriptor**

*Kylie Gorman Department of Computer Science The College of New Jersey Ewing Township, New Jersey 08618 Email: gormank2@tcnj.edu*

*Mentor: Andrea Salgian Department of Computer Science The College of New Jersey Ewing Township, New Jersey 08618 Email: salgian@tcnj.edu*

*Mentor: Yang Zhang School of Electrical Engineering and Computer Science University of Central Florida Orlando, Florida 32816 Email: yangzhang4065@gmail.com*

**Abstract**

*In computer vision, key points are positions of high interest in an image and descriptors are numerical values that are computed around those key points. Therefore, two different descriptors can be compared to determine if two images have the same contents. Most of these descriptors focus on the shape features of an image and disregard any information about color. Color description is a challenging problem to overcome due to events such as shadows, shading, or differing perspectives [11]. This paper will discuss a new technique for color description referred to as blockwise color moment feature, which is robust towards these scene accidental events. Color descriptors are most successful when combined with another descriptor, so we will combine our color descriptor with Dense SIFT to improve the current standard.*

**Key Words**

*Color Descriptor, Color Moments, Dense SIFT*

1. **Introduction: Problem Motivation**

Our project pursued a more accurate means of solving the problem of image classification. More specifically, when given an image, our system will look for pictures that are similar to it in a given database. Previous methods for image classification have included using shape descriptors. We seek to prove that color information included with these shape features will produce more precise results. Therefore, we are introducing the blockwise color descriptor. Our descriptor improves previous attempts because it first breaks up the images into small blocks before it calculates the color information. We will then combine these features with the Dense SIFT shape descriptor to get the best results.

Color description is challenging due to significant variations in RGB values for similar colors as a result of events such as shadows, shading, specularities, illuminant color changes, and differing perspectives [11]. In order to enhance current results, it is essential that our new descriptor has the capability to differentiate colors that are encountered in everyday life [7].

1. **Related Work and Previous Approaches**

Many approaches to color description have been employed in the past. One former method utilized color histograms to establish the distribution of color throughout the image. However, problems can arise due to the fact that different images may produce the same histogram. For example, Figure 1 displays the distribution of color in the image containing the parrot. However, an image of a beach ball may have the same color distribution and therefore be recognized as the same object. Additionally, in order for a color histogram to be classified using a Support Vector Machine (SVM), it requires a nonlinear kernel, which is time-consuming to compute.



Figure 1. Example of Color Histogram being utilized as a color descriptor

Another approach, referred to as color mapping, uses a function to map the colors of an image to a desired color. Color mapping is useful in reducing the amount of color in an image by delegating a wide range of RGB values to the same color. This method can be useful in the third case of Figure 2; however, when there is too much detail required in the images, the first and second image produces abnormal outcomes [6].

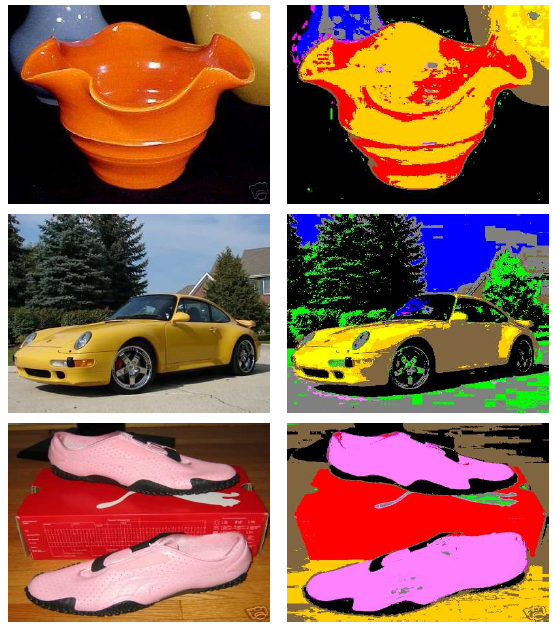


Figure 2. Visualization of Color Mapping

The most successful technique for color description uses calculations called color moments which are measurements that differentiate images based on their color content. These calculations are useful because they can numerically determine the color similarity between images [10]. Instead of only referring to a color by its pixel value, we can now use three different probability distributions based on the color distribution in an image.

The first color moment is the mean color value in the image, as shown in Equation (1), where *i* is the color channel (red, green or blue), *j* is the pixel number, P*ij* is the pixel value, and N is the number of pixels in the image.

(1)

The next calculation can be seen in Equation (2) and is the standard deviation (the square root of the variance) of the distribution [8]. The standard deviation requires the mean calculation, Ei.

(2)

The final calculation (3) measures the skew or the degree of asymmetry in the distribution [8].

(3)

Once all three color moments have been computed for two images, they can be compared to determine image similarity.

1. **Approach and Background Information**

Although using color moments is the most successful of the three methods mentioned previously, we seek to improve this method further. To glean more precise results when classifying images, we propose to use a blockwise color moment feature, which will be overall a more comprehensive representation of the color in an image than pixel color value, or traditional color moment. The objective is to integrate spatial context information to design a more efficient means to map the color name to a pixel value.

We will integrate all three color moment calculations discussed thus far. However, instead of computing these distributions over the entire image, we will first break up the image into blocks of 8 pixels by 8 pixels in size. We will then calculate the three color moments on each channel for each block. We plan to use three different 3-channel color spaces: CIELAB, HSV, and RGB. Since each color space is comprised of three channels, each block will have nine total measurements.

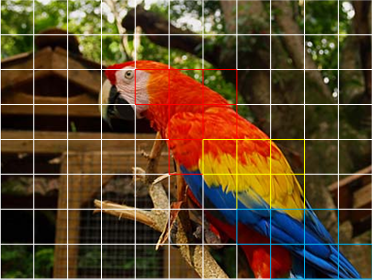


Figure 3. Visualization of the Blockwise Color Moment Feature from an image in the Birds 200 Dataset [4]

To increase the robustness of our descriptor, we will be using real world images from Google and EBay for training and testing. These images will be used in place of the chip-based color mapping technique, which uses colors created in the lab. Learning color from everyday life images will allow our descriptor to be utilized in real-world applications [7].

Our first datasets were comprised of images taken from Google and EBay. Each set contained the eleven basic colors: black, blue, brown, green, grey, orange, pink, purple, red, white, and yellow. The Google set contained 100 images per color with 1,100 images total, obtained by searching each color on Google. The EBay set contained four separate categories, dresses, cars, shoes, and pottery, with eleven colors per category and twelve images per color [6]. Some examples can be seen in figure 4.



Figure 4. Examples from the EBay Dataset [6]

Once we compute the color moment calculations for every image, we will produce a feature matrix that has a number of rows equal to the number of blocks created by nine columns, one for each corresponding color moment calculation per block. Due to the large number of blocks per image, to reduce processing time we must reduce the amount of data.

Principal Component Analysis (PCA) is a statistical procedure that reduces the number of dimensions of a data set. It condenses the data into its most basic components, removing any unnecessary elements. Given an mxn matrix, this matrix will represent a single point in an n dimensional space. PCA projects this point into a space with fewer dimensions effectively reducing the number of dimensions. This reduces processing time and eliminates unessential details [12].

Once the PCA calculations are completed, we will then use those results to compute the Gaussian Mixture Model (GMM), which is a collection of K Gaussian or normal distributions. Each distribution represents a cluster of data points. GMM is similar to K- means clustering because it groups similar features found throughout the dataset. Through this process we are creating a visual library of the possible features contained in each image [5].

Now that we have determined the different types of features encompassed within our dataset from our GMM calculations, we need to determine which category contains which specific features. Fisher encodings can be used to summarize many feature descriptors into a vector and further reduce the amount of data. Fisher vectors will record how often a feature occurs within an image, however they will also store the differences between the images [9].

All of the resulting Fisher vectors will have the same length because each vector will record the same number of features. We will then use a different Support Vector Machine (SVM) for each unique group of images. A linear SVM compares one category against all of the other categories. By considering the images as a single point in space and using our training data, we can define a line that gives us the most room for error between the category we are learning and all of the other images. We can then use this line on the testing data and by determining which side of the line the data falls on we can verify if it belongs to the category we are currently testing [3].

Color descriptors are even more valuable when they are combined with an existing shape descriptor. In our experiment we will be using the shape descriptor Dense Scale Invariant Feature Transform (Dense SIFT). Dense SIFT is a faster version of SIFT, which is an algorithm to detect and describe local features in images [1]. We will concatenate the feature matrices generated from Dense SIFT with the feature matrices created from our own descriptor and continue through the same steps already mentioned.

1. **Implementation**

The entire project was implemented in MATLAB, using the open source VLFeat library [2] for most of the algorithms our project required.

We began by writing code to separate the images into small blocks, each containing 64 pixels. A separate function was created to compute the color moment calculations. Therefore, when we integrated Dense SIFT, we could reuse the same code to break up the image, but replace the single function to compute the information from each block.

Once our color descriptor was created, the majority of the project concentrated on data management. After we obtained a feature matrix for each image, we needed to concatenate our results and pass them into the premade algorithms. We used included functions for PCA, GMM, fisher vectors, SVM, and Dense SIFT.

1. **Experiments**

We first tested our color descriptor against the Google and EBay images. This required creating a pipeline in MATLAB that was adaptable to different datasets.

As seen in figure 5, we first separated the training and testing data into separate folders. Using the color moment calculations, we obtained a feature matrix for every image. We then concatenated all training data feature matrices to calculate the PCA and concatenated the result to compute the GMM. We were then able to use those results to calculate the Fisher vectors for both the training and testing data.

The training Fisher vectors could then be used to train the eleven SVM’s, one for each basic color. We then proceeded to classify the testing Fisher vectors and the results produced a score matrix containing a row for each vector and a column for each color. Each entry contained the probability that a particular image was the color represented in a given column. To calculate the precision of the results, we checked the highest probability in each row and assigned each image to a color. We calculated the accuracy by dividing the number of correct matches by the total number of testing images.

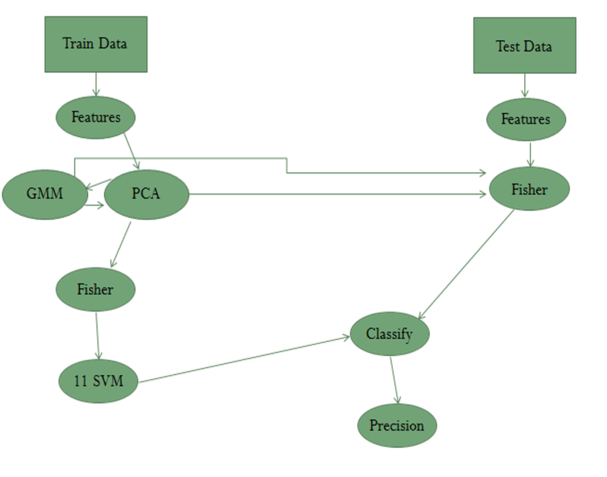


Figure 5. Visualization of Pipeline

After our color descriptor showed favorable results, we continued with our existing code while employing a larger dataset and incorporating Dense SIFT (Scale Invariant Feature Transform) along with our color moment calculations. Dense SIFT calculations are more time consuming because they include 128 key points per image against the color moment’s 9 calculations per image. This means that each Dense SIFT feature matrix will contain 128 rows instead of 9 and each matrix will require more time to compute and more memory to store.

To test our final descriptor, we will be using the Birds 200 data set [4]. This set of images contains 200 species of birds with 11,788 images total (see Figure 6). Birds of the same species should be recognized as the same category.



*Figure 6. Example images from the Birds 200 dataset [4]*

1. **Results and Analysis**

After our first experiment of using our color moment calculations with the datasets of Google and EBay images, the program was successful in distinguishing colors from one another. We tested the program on three separate three-channel color spaces and calculated the precision of each color space. The average precision (AP as seen in figures 7-9) is defined by the number of colors correctly determined by the descriptor divided by the total number of images in that category and then finding the average of all of the categories combined. CIELAB returned an average precision of 42%, HSV images had an accuracy of 45%, and RGB images were the most successful with 50% accuracy. Each of the color space’s results were significantly higher than a random guess which would have 9% accuracy, but the RGB color space proved to be the most accurate.

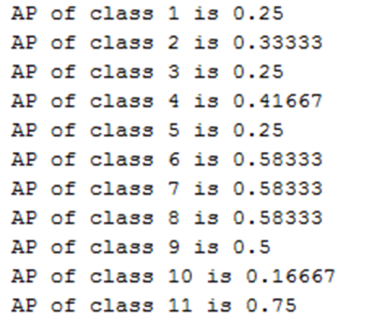


Figure 7. CIELAB Results for each color

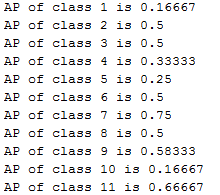


Figure 8. HSV Results for each color

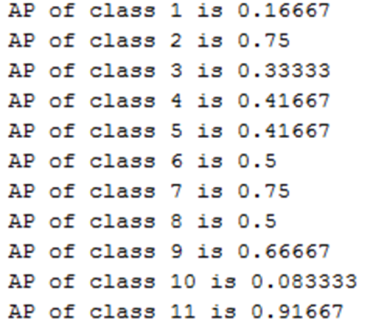


Figure 9. RGB Results for each color

Once the code was adjusted to handle a change in image number and format, we were able to acquire more results with the combination of our color moments and Dense SIFT. In order to obtain results before the end of the REU summer program, we used 20 of the 200 bird species. Our color moment was able to classify the images with a 21.36% accuracy, which was very similar to Dense SIFT’s accuracy of 21.75%. Both accuracies were well above the random guess which would have been 5% accurate. Our descriptor was most successful when combined with Dense SIFT, which improved the accuracy to 25.44%.

1. **Conclusion**

Color descriptors are most advantageous when combined with another descriptor. Our results greatly exceeded any random guessing as well as the current standard of Dense SIFT. The blockwise color moment feature accounts for scene accidental events in real world images, such as shadows and differenced in lighting, and therefore more accurately depicts images seen in everyday life.

Our color descriptor was very successful in its own right when learning the Birds 200 dataset, producing results nearly the same as those of Dense SIFT. When both descriptors were combined, the result was the clear winner, increasing the precision by about 4%. These results prove that when shape descriptors include the added element of color into their calculations, they can more accurately learn and categorize images.

1. **Future Work**

Due to the limited amount of time allotted from the REU program, there is still more progress to be made for our project. To begin with, we would continue to compare our color moment with Dense SIFT on the full Birds 200 dataset. Also, throughout the study we used a box size of 8 pixels by 8 pixels. We also want to study the results of increasing or decreasing the box size. Finally, we would like to incorporate object detection and image retrieval to fully test the ability of our color descriptor.

1. **References**

[1] A. Vedaldi and B. Fulkerson. 2008. Dense SIFT as a faster SIFT. http://www.vlfeat.org/.

[2] A. Vedaldi and B. Fulkerson. 2008. VLFEAT: An Open and Portable Library of Computer Vision Algorithms. http://www.vlfeat.org/.

[3] C. Chang and C. Lin. 2013. LIBSVM: A Library for Support Vector Machines. National Taiwan University. http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf.

[4] C. Wah, S. Srandon, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Computation and Neural Systems Technical Report, CNS-TR-2011-001.

[5] D. Reynolds. Gaussian Mixture Models\*. MIT Lincoln Laboratory. https://www.ll.mit.edu/mission/communications/ist/publications/0802\_Reynolds\_Biometrics-GMM.pdf.

[6] F. Shahbaz Khan, R.M. Anwer, J. van de Weijer, A.D. Bagdanov, M. Vanrell, and A.M. Lopez. Color Attributes for Object Detection. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012.

[7] J. van de Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning Color Names for Real-World Applications. IEEE Transactions in Image Processing, 2009.

[8] K.,Noah and B. Fisher. 2005. Color Moments. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\_COPIES/AV0405/KEEN/av\_as2\_nkeen.pdf.

[9] K. Simonyan, P. Omkar, A. Vedaldi, and A. Zisserman. Fisher Vector Faces in the Wild. British Machine Vision Conference, 2013.

[10] R. Fergus, L. Fei-fei, P. Perona, and A. Zisserman. Learning Object Categories from Google’s Image Search. In Proc. Of the IEEE Int. Conf. on Computer Vision, Beijing, China, 2005.

[11] R. Khan, J. van de Weijer, F. Shahbaz Khan, D. Muselet, C. Ducottet, and C. Barat. Discriminative Color Descriptors. Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. IEEE, 2013. (2)

[12] S. Raschka. 2014. Implementing a Principal Component Analysis (PCA) in Python Step by Step. http://sebastianraschka.com/Articles/2014\_pca\_step\_by\_step.html.