Resistor Recognition using Eigenvectors

Thomas Jagielski
Quantitative Engineering Analysis
Olin College of Engineering
Needham, MA 02492
Email: tjagielski@olin.edu

Sparsh Bansal
Quantitative Engineering Analysis
Olin College of Engineering
Needham, MA 02492
Email: sbansal@olin.edu

***Abstract*—Color-bands are used in the manufacturing of resistors in order to ease the process of determining the resistance of a resistor. However, many of the color combinations used are commonly difficult for colorblind people to differentiate. Due to this, we created an optimization algorithm in which we are able to determine the scale multipliers for the red, green, and blue color channels (independently) as an attempt to recognize the variation between resistors. Each of these multipliers will change the contrast of an image along the channel. This increases the algorithm's ability to distinguish between test cases. The algorithm decomposes an image as a projection into the axis defined by the set's principal components. As each of these projections is defined as a set of weights, we can compare the test image weights to the train images in order to classify the test image's weights based on the minimum distance between them. By focusing on the contrast within the blue color channel, we were able to identify 52.5% of the resistors correctly.**

I. Introduction

Color is commonly used to distinguish between items with similar packaging. A specific example of this is the colored bands on resistors. When considering people who are colorblind we questioned, how can colorblind people read the bands of a resistor? This question further relates to the larger context of whether or not we are able to use facial recognition techniques to distinguish between subtle changes across features within the test set.

We created an algorithm that used the red, green, and blue color channels for this recognition of these differences. Many of the positive applications of this software tool are in assistive tech by aiding the colorblind. However, it can be applied to many other applications that utilize color images for recognition. We focused on resistors because many of the color combinations used are difficult for people who are colorblind to differentiate.

Although being able to distinguish between test images that have more subtle differences could yield more success in correctly identifying the image, there are negative uses that can arise with regard to the image processing that is done. The success of our algorithm depends on the scale multipliers, which results in the inherent bias of the user affecting these scale multipliers. Although our algorithm is an optimization in which each color channel (red, green, and blue) is multiplied by a scale factor, these factors can be input and swept throughout a range by the user. In choosing the weights for further contrasting a color track, a user of the program is able to control what the algorithm deems important. For example, if the user was to set the red scale higher than green and blue, this signifies that the red variation is more important to the user.

An additional negative consequence of our algorithm is when it incorrectly identifies a resistor. Electronics are components with specific values in order to make circuits operate properly. It can be harmful if our algorithm incorrectly identifies a resistor as components can burn with excessive current. These components can be harmful by burning the hands of people who touch them.

Our algorithm breaks down the variation of images using the principal components of the variation in the image set. Principal component analysis (PCA) is a technique that is used to reduce the dimensionality of the data-set. By finding the non-zero eigenvectors, vectors that do not transform during a linear transformation, we can find the principal components. Each of these components are represented as a vector in the direction of the spread of the data. This set of vectors provides a new basis axis in which we can represent each of the images. After each image is projected into this new axis, we are able to compare a new image to the training set and find the image that is the closest in distance.

A common way to compute the principal components is using singular value decomposition (SVD). SVD is the factorization or decomposition of a data matrix into its eigenvalues and eigenvectors. These eigenvalues and eigenvectors come from the resultant matrices of both M^TM and MM^T where M is a m by n matrix. In terms of singular values, the orthonormal - each vector is normal and has unit length - eigenvectors of M^TM yield the right-singular values. Conversely, the left-singular values are given by the set of orthonormal eigenvectors of MM^T. The eigenvalues are broken-down as the square-root of the eigenvalues of both M^TM and MM^T.

II. Methods

We created an optimization algorithm in which we multiplied each of the RGB-color channels by a scale factor in order optimal disparity within color tracks. The algorithm would take the input of a matrix of values and compute the percent correctly identified for that specific combination of multipliers.

A. Image Processing

We created a data set by taking photos of resistors and passing them through a Python script we wrote. The script uses OpenCV-Python to template match an image of a resistor with an error threshold of 99%. Using the script, the images are then cropped around the resistor and scaled to 600 by 250 pixels. Each resistor has four images. There are two images in each orientation. The two orientations after passing through the python script are shown in Figures 1 and 2.

Fig. 1. The first orientation of resistors.

Fig. 2. The second orientation of resistors.

After each of the train images are input into MATLAB, their red, green, and blue color channels are isolated; each of these sets are multiplied by a scale factor, which increases the contrast for a color throughout the set; and concatenated into a single column vector. The test images are then put into a matrix where each column represents an image. Thus, the size of this matrix is the number of pixels in the image times three by the number of images. For our training set, this matrix was 450,000 by 80. The same process is used for the test set of images.

B. Nomenclature

Variable	Significance
Γ_n	Image in the set
μ	Average resistor
η	The number of images
Φ_i	Mean-centered image
M_{train}	Matrix of the normalized images in the training set
U	Left-singular value
Σ	Singular values
V	Right-singular values
ω	A matrix of the weights of each image
M_{Test}	Matrix of the normalized images in the test set
δ	Euclidean distance between two image's weights

C. Eigenvector Computations

Let each image in the training set be expressed as $\Gamma_1, \Gamma_2, \Gamma_3, ..., \Gamma_n$. We can find the "average resistor" image by,

$$\mu = \frac{1}{\eta} \sum_{n=1}^{\eta} \Gamma_n \tag{1}$$

where μ is the "average resistor" and η is number of images.

The images were then normalized the images by mean-centering them. In doing this,

$$\Phi_i = \Gamma_i - \mu \tag{2}$$

where Φ_i is the mean-centered resultant image.

Let M_{train} represent the matrix of normalized images such that,

$$M_{train} = [\Phi_1, \Phi_2, \Phi_3, ..., \Phi_n] \tag{3}$$

We can then find the principal components of the image set in order to represent the images in a new axis. This will allow us to reduce the dimensionality of the set and express the variation of the images. In order to find the principal components we use SVD. M_{train} has a SVD form of

$$M_{train} = U \Sigma V^T \tag{4}$$

where U is the n by r matrix whose columns represent the eigenvectors of $M_{train}^T M_{train}$, Σ is a diagonal matrix in which the singular values (the square-root of eigenvalues) are the entries along the diagonal, and V is an m by r matrix whose columns represent the eigenvectors of $M_{train} M_{train}^T$.

In order to find the non-zero eigenvalues and eigenvectors, we use $M_{train}^T M_{train}$ as it yields an 80 by 80 matrix rather than a 450,000 by 450,000 matrix with many of the entries being eigenvalues of zero. As the singular values and vectors satisfy both,

$$\mathbf{M_{train}} \mathbf{v}_i = \sigma_i \mathbf{u}_i \tag{5}$$

and

$$\mathbf{M_{train}} \mathbf{u}_i = \sigma_i \mathbf{v}_i \tag{6}$$

simultaneously. After finding the eigenvalues of this matrix, we know the singular values of the matrix as the square roots of these values. They also can be used to determine the eigenvectors. Further, we can determine the left-singular values by rearranging equation 5 to form,

$$\mathbf{u}_i = \frac{\mathbf{A}\mathbf{v}_i}{\sigma_i}. \tag{7}$$

Once we determine the eigenvectors using SVD, we are able to preform PCA on the set. These vectors prove to be a convenient set of basis vectors for a new axis in which we can represent the images. To do this, we project each image into the new axis. We do this with,

$$\omega_{train} = U^T M_{train} \tag{8}$$

where ω represents a set of matrix weights for a linear combination of the new basis vectors to represent each image. Effectively, this operation computes the dot product of the two vectors. By multiplying the new set of weights by the original image, we can create eigenresistors, which represent the variation of images across the set. The fifth eigenresistor is shown in Figure 3.

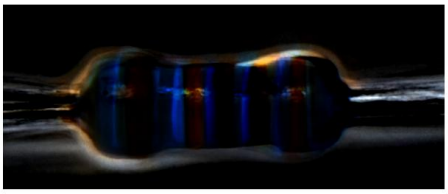

Fig. 3. The fifth eigenresistor image created from the weights.

After computing each of the weights for the training set we can compute the weights for the test set by,

$$\omega_{test} = U^T M_{Test} \tag{9}$$

where M_{Test} represents a matrix of the test images in the same format as the training set. Each of the weights can be compared to determine which images are the closest in euclidean distance,

$$\delta_k = || \ \omega_{test_k} - \omega_{train_k} \ ||^2 \tag{10}$$

When the weights are similar, the images can be classified as the same.

D. Summary of Computations

1) Load in the training set of images.
2) Isolate the red, green, and blue color channels.
3) Multiply each of the color channels by a scale factor.
4) Concatenate red, green, and blue to represent the image as a single column vector.
5) Normalize the data and compute the principal components of the variance within the face data. (While using MATLAB, 'svd' with the optional argument 'econ' is particularly useful for this).
6) Compute the weights for the images by projecting them into eigenspace.
7) Load the test set of images and repeat steps 2-4.
8) Project the test images into the eigenspace.
9) Compare the weights for the vectors that are the closest together to classify the test image.
10) Compute the test accuracy for the test set.

III. Results and Discussion

We invalidated the use of the eigenface algorithm presented in *Eigenfaces for Recognition* by Matthew Turk and Alex Pentland by implementing their algorithm on our train and test set of resistor images [1]. This method yielded a 30% correct identification.

We then included the red, green, and blue color channels for each of the images. This increased the accuracy to roughly 40%. A plot of the accuracy with respect to the number of eigenvectors used is shown in Figure 4. We can see that as we

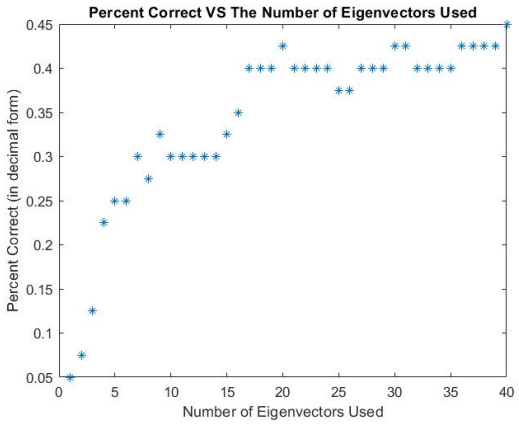

Fig. 4. Accuracy as a function of the number of eigenvectors used.

increase the number of eigenvectors used, the accuracy of the algorithm increases.

After adjusting the multipliers for red, green, and blue we found the weights,

Red Multiplier	Green Multiplier	Blue Multiplier
0.05	0	1
0.05	0	0.7

both yielded 52.5% accuracy. This was the highest accuracy we found through various weights with a trial and error method. The accuracy at various multipliers for red and blue (we found including green decreased the percentage correct) in Figure 5. From this, we can see when we increase the amount of red coloring used the accuracy tends to decrease. There is a similar trend for the blue color track. With any other weight for green, besides zero, the accuracy decreased. Thus, we can conclude the blue color track seemed to play the largest role in distinguishing between resistors. Colors like brown and black are very difficult to distinguish, but when modifying

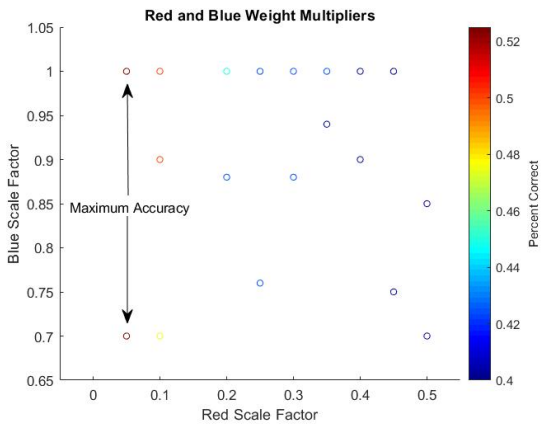

Fig. 5. Graph depicting the percentage correct with a change of scale for red and blue. The arrows indicate points of maximum accuracy (52.5%).

the contrast of various color tracks makes them more easily identifiable.

If the technology fails, and someone uses the incorrect resistor for a circuit, their circuit may malfunction and potentially cause harm. For example, if too much current goes through the resistor you may burn yourself and the breadboard. However, other ethical implications arise from misuse. Most specifically, the weights that determine which contrasts are the most important can be set by the user, and thus have the potential to incorporate bias.

For future iterations, we could potentially implement the Fisherfaces algorithm in order to create more variety in the lighting conditions for the train and test images. Additionally, if we were to use a micro-lens while taking images to make the images with better quality. We could also ensure that each resistor is in the same placement and aligned correctly. Another addition that would be useful would be a threshold in which we can determine if a resistor is not contained in the training set, and thus, cannot be identified.

IV. Conclusion

By utilizing the red, green, and blue color channels to perform the recognition of resistors, our accuracy increased by 10 percentage points (from 30% with black and white to 40% with the colors). It even further increased when we provided a set of weights for each of the color tracks. With weights of 0.05 for red, 0 for green, and 1 or 0.7 for blue, we were able to identify the correct resistor with 52.5% accuracy. This method, of applying weights to specific channels, can be applied to the recognition of color images. However, the algorithm will be limited in a variety of lighting conditions as it drastically influences the color of the object.

This algorithm can be used to separate out some resistors and aid in the process of determining if you are using the correct resistor. It can be especially useful for people who are colorblind. Many of the color combinations that are used as resistor bands, colorblind people may have an especially difficult time identifying correctly.

It would be interesting to consider surface mount component recognition or this algorithm in an industrial application. More specifically it could be used for sorting resistors during manufacturing or reusing resistors that were recycled. Assuming the surface mount components could be recognized, as a form of quality control to ensure the correct components are on a PCB.

References

[1]
Matthew Turk and Alex Pentland, ”Eigenfaces for Recognition”, The MIT Media Laboratory