

# PROJECT REPORT

## Quality Control using PCA Eigenfaces



CHEMENG 765: Multivariate Statistical Methods for Big Data Analysis and Process  
Improvement

Submitted to: Dr. Brandon Corbett

Submitted by: Livanshu Kashyap

#400286794

[kashyapl@mcmaster.ca](mailto:kashyapl@mcmaster.ca)

## 1. ABSTRACT

Using PCA in Machine learning and quality control: Scanning images from the conveyer belt of a submersible pump impeller and detecting if the product is under quality specifications. Data was collected from a company's original data. The algorithm would use PCA as a major constituent to decrease dimensions of image eigen data from 90000x90000 to a much lower 200x200 matrix. The algorithm was coded and tested in Python environment. Also, the data was processed in SIMCA to compare results.

## 2. INTRODUCTION & DATABASE

The data was a real-life data from a company named Pilot Technocast and was sourced from Kaggle database [\[1\]](#). It includes images of top view of the pump impellers manufactured through casting process. A total of 7348 photos are in the database which includes 6633 training and 715 testing images. Training and testing data both are divided into 'defective' and 'okay' images. The images were well captured with enough lighting and are cropped accordingly by the provider. Every image has a size of 300x300 pixels and 96dpi. Images were converted into Grayscale because in the objective of this project, RGB colors do not play any vital role. Grayscale magnitude ranges from 0 to 255 and every single pixel has its own single grayscale value. Whereas, in RGB, every single pixel has 3 different color values, thus, making it a 3x1 matrix for 1 pixel.

For computational efficiency and limited processing memory in PC, I only took 200 images for training the system (i.e. 100 defective and 100 okay). For preparation of database, the images were saved into the Python code directory. Due to some reason, the python function does not input images if they have a special character in it. Therefore, all the images were renamed as they have a special character when downloaded from database.

### 3. OBJECTIVE

The main objective of the project is to make a system which automatically detects if the product is under quality specifications or not, by scanning and processing its image. In this era of automation, quality control systems via AI and Machine learning are in huge demand. The other aspects are to keep the system as simple and cost effective as possible. For real time application of the project a continuous scanning camera is needed but for the project and algorithm, only still images are used in testing phase.



*Figure 1 Process flow of testing phase*

## 4. METHODOLOGY

The methodology behind this is to train the system to detect the status of quality of the product and then test it in real time. PCA Eigen face is one the simplest approach used in image recognition. It transforms higher dimensional data in lower dimensional. It retains only the useful datapoints from the image datasets and neglect others. The advantages of using this technique is simplicity and higher speed of processing. The process is divided into 2 parts:

- i) Training the system
- ii) Testing the system

### 4.1. Training Phase:

- Collect the data from images and convert them into useful data.
- Calculate Eigen faces using PCA and keep top K eigen faces with higher value.
- Calculate weights of each image by projecting them into the eigen space.

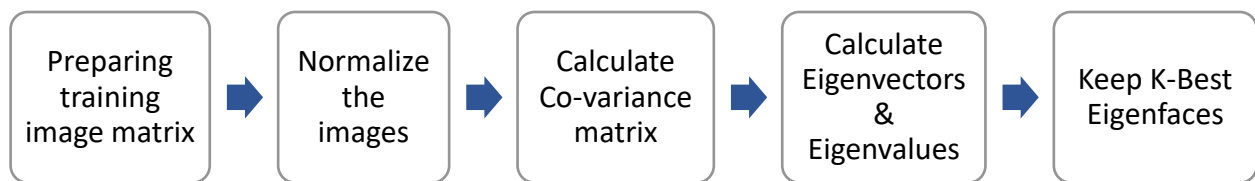
### 4.2. Testing Phase:

- Compare the weights of testing image with eigen space to find the difference the datapoints.
- Result is shown by using a threshold between testing image and eigen space (closest to the training image variant).

#### 4.3. Eigen face calculation Algorithm

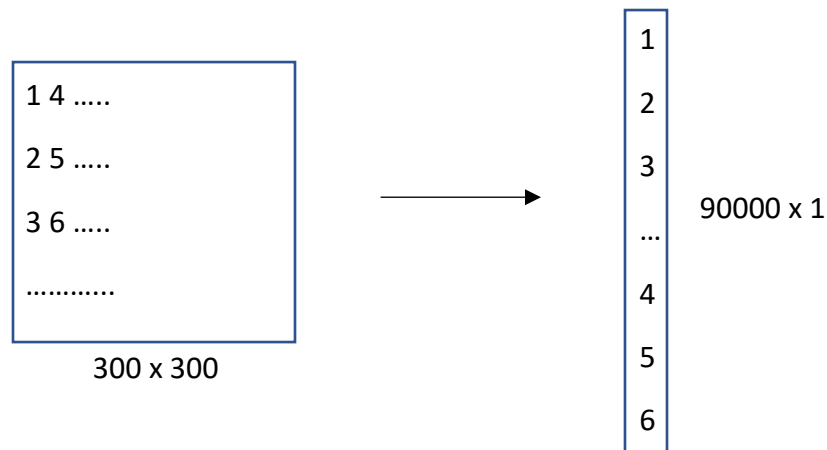
Images are 2-Dimensional array of color intensity values. The intensity value depends upon the nature of colors in image. If the image is Grayscale type then the data points value would range from 0 to 255 and for RGB, each datapoint would be a 3x1 matrix containing values of Red, Green and Blue constituents. In this project, Grayscale images are used because color specifications are not required. Grayscale data is less in size and can be computed faster.

The following algorithm is used for computation:



##### 4.3.1. Preparing training image matrix

Suppose an image has  $M \times N$  dimensions, therefore, it can be seen as  $\text{Image}(M,N)$ . In this project, 300x300 pixel images are used. Therefore,  $M = N = 300$ .



First step is to convert image into grayscale datapoints and flatten all training images. All the images are put in a matrix and final matrix(A) would be 90000 x 200, as 200 images are flattened.

#### 4.3.2. Normalizing image vector

For normalization, average of the original image data is calculated and then the average vector is subtracted from the original data to make normalized vector.

#### 4.3.3. Covariance matrix

A covariance matrix is calculated using the formula,

$$C = AA^T \quad \text{where } A \text{ is image data matrix}$$

The size of the matrix after multiplication with transpose in this project is to be of a dimension 90000x90000 because our A matrix is of 90000x200.

Computing this big size of matrix is undesirable. Therefore, PCA is to be done to reduce the dimensions to keep the required important data and eliminate others.

#### 4.3.4. Principal Component Analysis and Eigen face generation

PCA is a technique used in Multivariate statistics to reduce dimensionality of datasets while preserving as much variance in data as possible [\[5\]](#). PCA is applied for Eigen face approach to reduce dimension of our covariance matrix. This would result in reduction of variables into smaller ones i.e. Principle components. As we know, dimension reduction also reduces information, for this, best low dimension space is to be determined by best principle components.

Eigen face approach is an efficient approach for image recognition as it uses the variation in the dataset and use the information to process and compare images of different variants. In our case, there are 2 variants i.e. Defective and Okay.

PCA with eigen face approach is done to reduce dimensions of covariance matrix and calculating eigen values & eigen vectors.

Consider a matrix,

$$Z = A^T A$$

Compute eigen vectors  $v_1$  of Z,

$$A^T A v_1 = u_1 v_1$$

Multiply A on both sides,

$$A A^T A v_1 = u_1 A v_1$$

$$Z A v_1 = u_1 A v_1 \quad [Z = A^T A]$$

$$Z U_1 = u_1 A v_1$$

Thus, we can say that both Z and C have same eigenvalues and their eigenvectors are related as  $U_1 = u_1 A v_1$

We know that  $C = A A^T$  can have up to 9000 eigen values and eigen vectors and  $Z = A^T A$  can have up to 200 eigen values and eigenvectors.

By the relation above, we can say that the 200 eigen values of C corresponds to 200 largest eigen values of Z.

Therefore these 200 eigen vectors of  $Z = A^T A$  are used to find 200 eigen vectors of C and form eigen faces using the formula,

$$X_i = v_i A$$

Where  $X_i$  are eigen faces.

\*PCA analysis is explained in Appendix file.



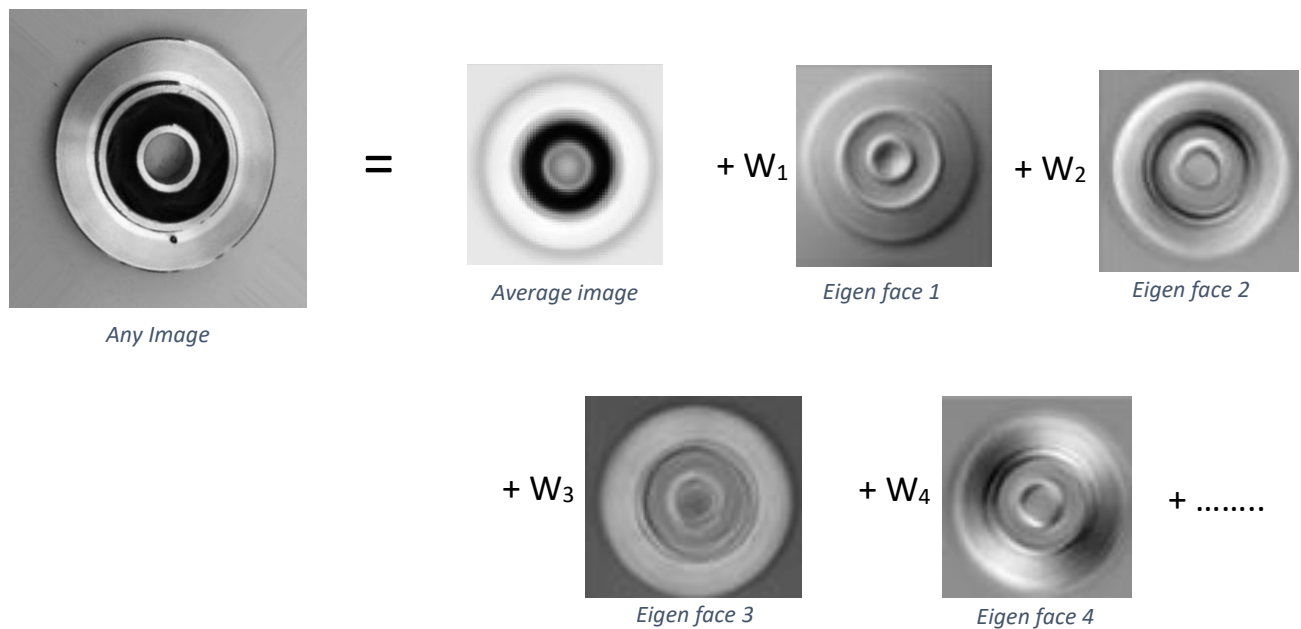
#### 4.3.5. K-Eigen Vectors

Eigen faces with larger values are to be retained and lower value eigen faces are deleted because they contain only a negligible part of characteristic of data.

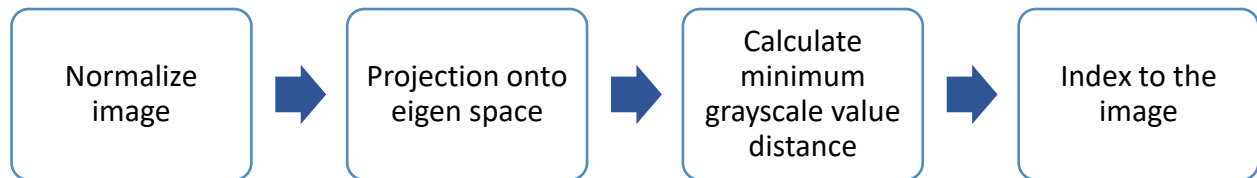
For this project, K value has been tested with range 5 to 20 and 11 is found out to be the most efficient value. K value can be depicted as principle components.

This is then confirmed by doing another PCA on Simca. Result came out with 11 principle components as most efficient.

After this, training sample is projected on Eigenface space and weights are calculated. These weights define how much of an eigenface describes an image. An example is shown below:


$$\text{Any Image} = \text{Average image} + W_1 \text{ Eigen face 1} + W_2 \text{ Eigen face 2} + W_3 \text{ Eigen face 3} + W_4 \text{ Eigen face 4} + \dots$$

#### 4.3.6. Testing the images



Testing phase includes normalizing the image and then projecting it on eigen space of training images. Projection is done to find the difference between the grayscale values. The datapoints with minimum difference is selected and image related to the datapoint is identified. The images related to it are indexed and are the result of the testing.

## 5. RESULTS

In testing phase, images were randomly picked from the data base to test if the system can detect the quality of product. A demo of the process was shown in the presentation. The testing results and visual inspection were compared to check the accuracy of the system. Due to quality of data the results were quite good. A total of 50 random photos were processed and compared with visual sight.

Testing Results	No. of Images in random testing set	No. of correct result w.r.t. input	Percentage
Defective	23	18	76.26 %
Okay	27	26	96.29 %

Defective images have lower accuracy percentage because of more variation in defective pixels whereas result of okay images was almost perfect. The reason of lower accuracy might be because of a smaller number of trained images (200 only).

Overall, the data was well produced by the provider and that is the reason the accuracy was good as compared to number of images tested. The accuracy would much more if more images are trained. I believe it might cross 95% if system is trained with all 7000+ images.

\*PCA Analysis on Simca is explained in Appendix file.

## 6. FUTURE WORK

- For future, the testing phase can have a continuous scanning camera which detects the image and shows results on the spot. OpenCV which is a programming function for Python supports continuous camera scanning.
- A sorted system can be attached which can sort defective and okay images from conveyer using pneumatic cylinders or any other process from the conveyer by communicating with the system.
- The system should be made in such a way that it keeps training itself by adding a very small threshold to it. For example, if the system detects the product is defective, the data should be added to the training data set. In this way, more and more variation can be stored in system and would improve accuracy.

## 7. REFERENCES

1. Casting product image data for quality inspection Version-1 by *Ravirajsinh Dabhi* uploaded on Jan 2020. Link: <https://www.kaggle.com/ravirajsinh45/real-life-industrial-dataset-of-casting-product>
2. Face Recognition Using Principal Component Analysis Method by *Liton Chandra Paul, Abdulla Al Sumam*, "International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 1, Issue 9". Link: <http://ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-9-135-139.pdf>
3. Image Recognition with Eigenfaces, Python Machine Learning, Tutorials on Python. "Machine Learning, Data Science and Computer Vision by ZENVA". Link: <https://pythonmachinelearning.pro/face-recognition-with-eigenfaces/>
4. Face recognition based on PCA and logistic regression analysis, by *Changjun Zhou, Lan Wang, Qiang Zhang & Xiaopeng Wei*. "Key Laboratory of Advanced Design and Intelligent Computing (Dalian University), Ministry of Education, Dalian 116622, China". Link: <https://www.sciencedirect.com/science/article/abs/pii/S0030402614008511>
5. Principal component analysis: A review and recent developments by *Ian T. Jolliffe and Jorge Cadima* Phil.Trans. R. Soc. A.37420150202 Link: <https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202>