Homework 2

Data Compression

2020311381 Sanghyeok Choi

**Goal of Project** : Understanding Linear Transformations

Practicing K-L Transform, SVD and Comparing with DCT in sense of data

compression.

**Tools :** DCT, K – L transform, SVD

1. DCT ( Discrete Cosine Transform )

Assume image is N N square matrix. Set DCT matrix SN = {s(m,n)}.

s(m,n) = . Where c(0) = and c(m) = 1 for .

Apply DCT in image matrix A : SNASN’. When DFT, DFT matrix is unitary so you don’t need transpose in second DFT matrix but in DCT, it in not unitary so you need transpose in second matrix.

2. K – L transform

A with eigenvectors ui of covariance matrix of image . It considers ensemble. In this case I divide image in partitions, so partitions can be ensemble. Let’s set = A(g-). is mean of image ensemble. Covariance matrix of : E() = E[A(g-)(A(g-)T)] = E[A(g-)(g-)TAT]. A is Deterministic matrix unlike (g-), so E[A(g-)(g-)TAT] = A \* E[(g-)(g-)T] \* AT = A \* Covariance matrix of (g - ) \* AT. You can change equation as

Covariance matrix of (g - ) = AT \* E() \* A. You can think E() is diagonal matrix. Then E() is become eigenvalue and AT become eigenvector. Now you get A matrix and . You can convert g to = A(g-). Also you can reconstruct g = AT + .

3. SVD ( Singular Value Decomposition )

You can set g = u∧v’. u and v is unitary matrix. ggT = u∧v’v∧u’ = u∧2u’. So eigenvector of ggT is u, square root of ggT’s eigenvalue is ∧. gTg = v∧u’u∧v’ = v∧2v’. So eigenvector of gTg is v, square root of gTg’s eigenvalue is ∧. However eigenvector can be 2, that only sign is different. So it can be error when you do not consider relation between u and v. gTu = v∧u’u = v∧. So v = gTu/∧.

**Process :**

In this project, I use picture of Lenna. And I convert gray scale



< Fig 1 >

This picture is 256 x 256 size, so it needs to divide in small patches. In this experiment I use 8 x 8 patch size that mean 256 x 256 image divided in 1024 8 x 8 patches.

There are 3 ways to compress data: DCT, K-L transform, SVD. There are some different data compress in those method.

First method is DCT. When you apply DCT by DCT matrix, right lower region has much less value than upper left region. So you can remove values in right lower region for data compress. If you apply inverse DCT by D’, you can get almost original image because you just remove small value.

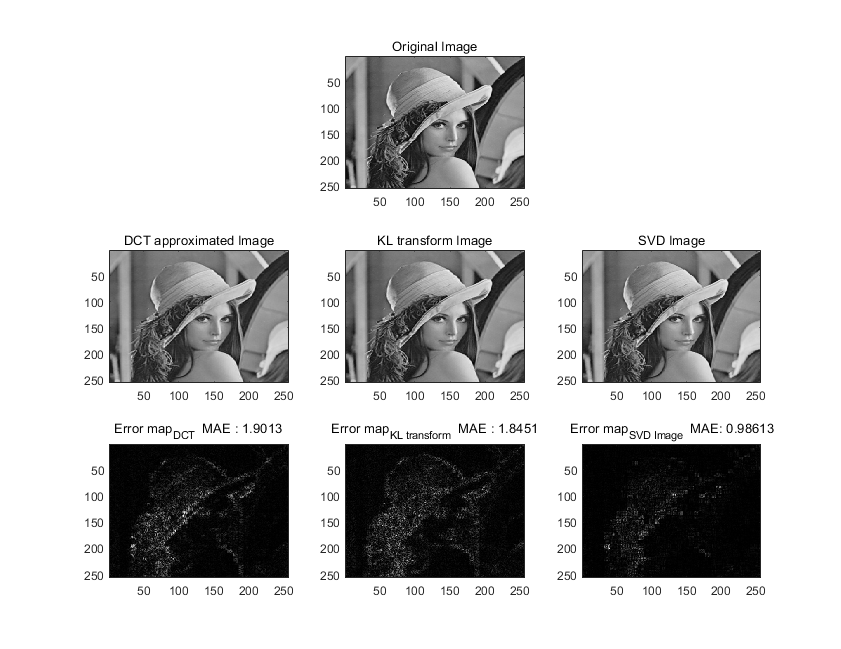
Second method is K – L transform. You can convert g to = A(g-). AT is eigenvector of covariance matrix of ( g – ). Remember and A. You can data compress by = A(g-). When you remove eigenvectors that have low eigenvalue, ’s size become shorter than original.

Also you can reconstruct by g = AT + .

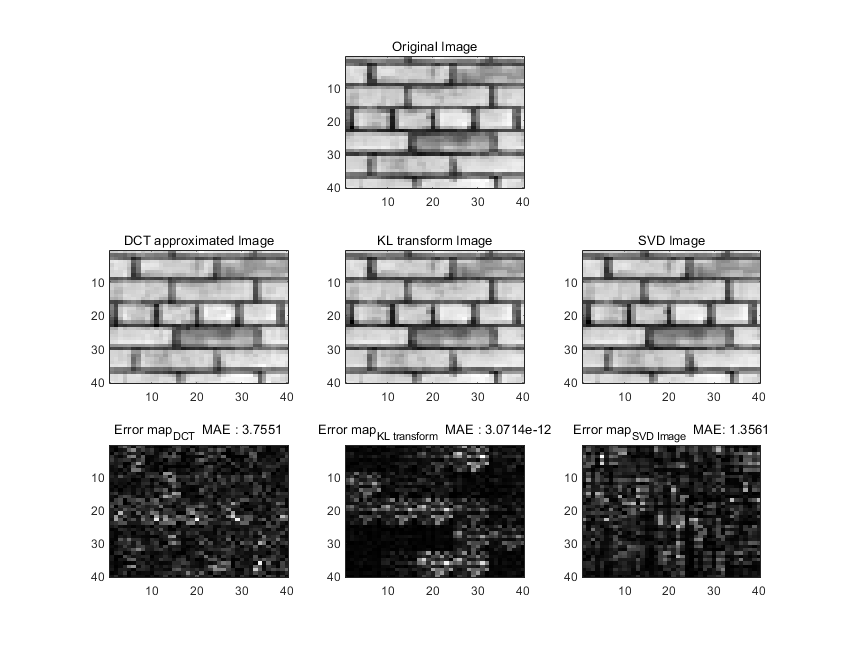
Third method is SVD. Set g = u \* ∧ \* v’. When eigenvalue is low, its effect could be low so delete correspond eigenvector in u and v. Also remove correspond eigenvalue in ∧. So you can reconstruct by small error.

I apply this method by “Matlab” tool.

**Result :**

****

< Fig 2 >



< Fig 3 >

MAE is absolute mean error. Take difference between original and reconstruct image in each pixel and get average of absolute difference. In first row in each figure, I plot original image to compare. In second row, I plot each reconstruct images. In third row, I plot absolute difference error between original and reconstruct images. In original image, SVD has least error in three method. On the other hands in texture image, K – L transform method has least error. Let’s see in each method.

First DCT remove high frequency. There is much more information in low frequency, so it does not matter to remove it. In original image. Compare with other method, high frequency has little value that could ignore but bigger than other method’s ignore value. In texture image, high frequency has lots of information than original image, so blurring effect happen. Error is much bigger than other method when texture image.

Second K-L transform is stationary & stationary method. It is not well in Original image. Of course it is better than DCT because DCT remove high frequency. However it is best on texture image.

Third SVD using eigen value and eigen vector. It is best on Original image. But it is not best on texture image. Although it is not best on texture image, it is good method compare with DCT method. But anyway, three methods are well reconstructed.

**Discussion :**

Does three method have mutual compatibility? I don’t think so. There are pros and cons between three method. As you can see SVD has least error in original image. If you focus only low error in usual image, use SVD method. When you do data compress, DCT is most simple. Just apply DCT matrix. So cost is cheap. Other methods are much more complicate because they need to calculate eigenvalue and eigenvector. If you want to use cheap cost in data compress, use DCT method. How about K – L method? Actually it is not good for data compression in usual image. It is not simple and error is bigger than SVD. KLT assume image as random field that satisfy ergodicity. But “Lenna” image is not random field but deterministic and does not satisfy ergodicity. It uses for recognizing object by dominant features. How about stationary and ergodic texture image? It has almost 0 error as you see. When you use for pattern, use this method So there are 3 method and pros and cons. So when you have to compress the data, consider the situation and select most proper method.

**Reference :**

Chapter3 of TextbookFile

<https://kr.mathworks.com/help/matlab/ref/eig.html>

https://ko.wikipedia.org/wiki/레나\_(이미지)

http://www.ux.uis.no/~tranden/brodatz.html