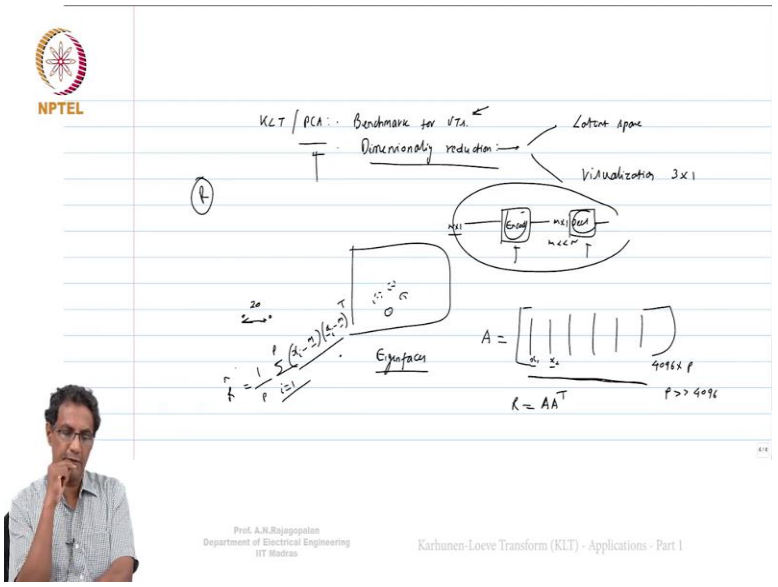
Image Signal Processing
Professor. A.N.Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture No. 53
Karhunen-Loeve Transform (KLT) – Applications – Part 1

(Refer Slide Time: 00:16)


KLT / PCA : · Benchmark for UTs.
· Dimensionality reduction :
Latent space
Visualization 3×1
n×1
Encoder
m×1
Decoder
m<<n
20
$\hat{R} = \frac{1}{p}\sum_{i=1}^{p}(x_i-m)(x_i-m)^T$
Eigenfaces
$A = $
x_1 x_2
4096×p
p>>4096
$R = AA^T$
NPTEL
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras
Karhunen-Loeve Transform (KLT) - Applications - Part 1

But then the PCA by itself has several other applications, so since we are on it, I just start in passing and also mentioned those so the PCA so one of the things is as a benchmark, so the KLT or PCA whatever you want to call it, one of the things is it is a so it is a benchmark for let us say unitary transforms. So, this is a benchmark for it for UT’s which is unitary transform but then a PCA also finds wide usage elsewhere here probably you simply limited to being a benchmark whereas places where it is used for example, one is a dimensionality reduction even today you will find you know people using PCA left and right for doing the dimensionality reduction.

In fact, if you if you have done if you have write something about let us auto encoders, especially the say deep learning type, so when they actually when they actually build a deep learning auto encoder which is typically nonlinear, so first the thing they will show is that if you constrain everything inside a deep network to be linear, then then the kind of the kind of auto the encoder decoder structure that you get will be exactly similar to that of the PCA. So, in fact identical you can show that you can show that you can show that what you end up doing there is in fact, it is simply a PCA.

So, if you force the deep learning thing to be totally linear, then it actually turns out that you will be implementing a PCA and the dimensionality reduction by itself people do for let us say various reasons, so for example, of course none of this can all be lousy that is I mean you know people are kind of willing to work with some work with some lousy you know sort of a reduction, because when you do this kind of of a dimensionality reduction, you may lose some information, but the idea is that you at least have an idea about how much you are losing and so on and therefore therefore you do not mind doing this kind of reduction.

And I know there are kind of two ways I mean you know one is so the dimensionality kind of say reduction, one thing is to is to be able to know go into a kind of a latent space, which is your feature space, where you say that now I am able to be able to bring it down so you can take some samples inside and then, you know, you can think of think of having some kind of an encoder it will actually take it down to down to a dimension which is much smaller, so so in this case, I think of let us say some like in cross 1 victor going in, then you get some like going to say m cross 1, where you know m is much less than n and then of course another then you send it through a decoder and then you see outcomes outcomes and you see estimator for whatever is input.

So, you may not always get back this but then the idea is that how much can you kind of go down by so that you can actually you know minimize minimize the error over it is a several several examples, so these kind of things people always do and here which is where I said I mean if this was nonlinear, this was nonlinear then of course you can do much better than a PCA, but cut if you restrict all these guys to be linear, then what you will end up implementing it simply a PCA.

On the other hand people also do dimensionality reduction, if you want to do some kind of this one what we call a visualization, what this means is that you know sometimes you do you try to solve some kind of a classification problem and you may have n classes and then I know you have whatever it could be it could some features that you I have got for goy for each class, but these even these features can still be in some high dimensional space, like for example, this m could still be in the 100s.

So, in a hundred dimensional space, how do I even see as to what is going on whether these groups are overlapping, whether these groups are actually separate I mean how well do they look and so on. So, so in that space you cannot you cannot even imagine what is going on, so what they will do, so they will finally bring it down to something like, you know 3 cross 1 sort of a vector which means that you can actually plot it and it is not for reconstruction purpose or anything because nobody typically will kind of bring it down all the way from 100 to a dimension of 3.

But then this 3 or 2 is typically done for example, sometimes people do do a kind of you know 2 cross 1 in order to simply see as to where all those point see for example if in that high dimensional space if you think about let us say, a person 1 his features are somewhere here, person 2 features somewhere here, person 3 features somewhere here we do not even know where they are. So, therefore if you do a PCA bring down all the features and then if you plot them, I mean sometimes, you know, even in this in this low dimensional space you might be able to see that there is there is a good deal of separation going on.

Even though this will not reflect exactly what is going on there, because in that dimensional space what may be separate may not be separate here, but then many a times people use it simply to be able to show that you know that these classes are maybe some overlap is there but then these are all of the intra-class variance is small the you know intra-class variance is still reasonably large, so all that kind of you know, this one is also done, so in fact that is that is the another place where they use, just bring it down, just crash the whole this on dimension down to some like 2 or 3.

But then that is not for reconstruction, that is simple because you visualize, visualize especially if you have features coming from (())(05:19) different classes, but you want to know how are they spaced out, one way to directly see could be like, you know within I say 3 cross 1, or 2 cross 1 kind of vector. Then people people the one, one application that let us PCA whenever whenever somebody talks about PCA is what is called Eigen faces, this is something that is that used to be a very sort of a popular thing, it is no longer because ever since deep learning networks came in your deep space net and all came, then this PCA kind of faded into the sun oblivion, especially for Eigen faces.

But but then this was one of the one of the one of the this was one of the algorithm that this is one algorithm that was talked about a lot, I mean you know during the days when let us say I do not know 85 90 tpyes, so at that time when let us say people started using PCA for doing face recognition. So, again the idea is like that, so if you are wondering, how they do this, do this R computation, so what they will do is you know, so basically imagine that you have lots of human faces, so you kind of stack all of them, right, I mean so for example in this case you can call this as some A, so here is a data matrix, so where you have let us say each vector is 4096 let us say suppose this 64 cross 64 and suppose you have some by by P, where P could be really large.

So, P will be much much greater than 4096, otherwise otherwise you cannot even learned anything and the sort of the thumb rule is like, you know, if m is the size of the data then at least m square but much more than that typically. So, so you need to have that many face images for example and also it was not easy because you know what face is come in the you know various sizes that I may have your face and you know I could have taken one from close maybe you know another face is from far, one is zoomed in another zoomed out, so what they would normally do is you know they would scale, so they will do some kind of a normalization.

So, what this means is that for every face they will make sure that the inter-eye sort of a separation is a kind of a fix that number, so they will say I do not know it was about 20 pixels or something like. So, they will fix that number, so they will take every face image they will make sure that there is a separation between the eye is that. So, if it is already zoomed out then it means that they will have to blow it up, if it is zoomed in they will have to shrink it a little bit so that because when you get a slap one above the other you want all of them to be aligned, otherwise what will you learn.

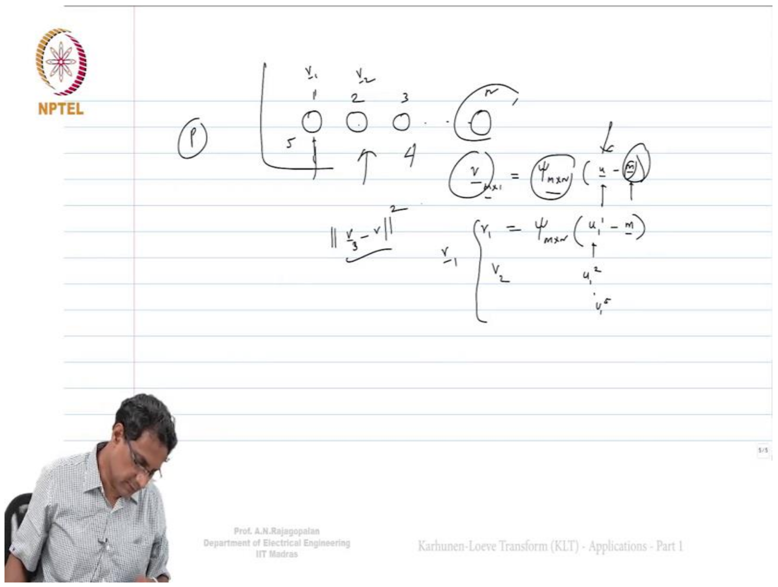
So, all this so it is not actually easy, so even today it is done by the way by except that deep network seem to be much more see robust a changes and so on. But then even now people actually do kind of you know create a data were in at least these variations can be removed, so that what is really need to be learned can be learned, so they will kind of do this normalization and after normalization it you will have all these faces stored up there, so we can think of this as.

So, then what you will do is you will actually compute this covariance a simply AA transpose, you can even do it like you can do it like that or or you can or you can actually take take the other way, the usual ways like R hat is equal to what is that, so if you have P number of example (())(08:27) say like 1 by P, then you go like i equal to 1 to P and then let us say xi minus m, xi minus m transpose, where xi xi xi is this is sample is xi this like x1 this like x2 and so on.

So, you can so the idea is that you so the more examples you have here the hope is that your estimate of R is going to be good, this is R hat, this is how this is how you typically get whether you have face or whether you have whatever else, if you have lots and lots of samples you compute the mean of that and then you do whatever you can then do this covariance thing or you can also do directly from that (())(09:10) itself.

And then once have R hat then you will go through an eigenvalue eigenvector kind of a decomposition, typically SPD is used for that, we have not done SPD yet, but after this you use a SPD to find the to find the eigenvalues and eigenvectors for this, then based on the eigenvalues, then you knock off whatever is not significant, then just use is a significant eigenvector, then you then you start. Now, for example, I mean terms a face recognition I mean their see again what you have what you wish to do later will change for example if you want to do face recognition, what you would do is a.

(Refer Slide Time: 09:45)


NPTEL
$\underline{v}_1$ $\underline{v}_2$
1 2 3
5 4
$\underline{v}_{M\times1} = \Psi_{M\times N}(\underline{u} - \underline{m})$
$\|\underline{v}_3 - \underline{v}\|^2$
$v_1 = \Psi_{M\times N}(u_1^1 - \underline{m})$
v_2
u_1^2
u_1^5
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras
Karhunen-Loeve Transform (KLT) - Applications - Part 1

So, imagine that let us say I have 1 person 1 a person 2 a person 3 something like whatever, I mean what this means is that out of those 4000 not out of those P image P number of images let us say that there are about n classes there, that this is person 1 this person 2, person 3 there are n number of them, n could be 1000 or something, it depends but now people can do millions and all let us say keep it as a small number, you have got let us say 1000 individuals all you want to do a do a kind of recognition.

Now, assume that for each people you have at least some examples, if you have 1 then of course very hard to do anything but assume that you know you have let us say 5 examples of each guy suppose which is all to assume. So, let us say for each person you have about 5 examples, so what you would do is, you know you will go back to the deep equation that we had which is like v you know m cross 1, so there you had psi you know which was m cross n and then u minus m.

So, so this m will be your will be your will be the average computed over over you see all the all the all the P, so that is really independent of the person, so this m is independent of which individual you are kind of looking at, this m comes from the entire the entire set and so the average of all the P examples is your m. Then then what you would do is, then then for each person, what you will do is you will actually take his image number 1.

Let us say if you are interested in computing, let us say suppose I want a transform we call it feature, suppose I want to find the feature for let us say for this guy, suppose I call that v1, now v1 I want features for this guy, then what I will do is, this psi is again same for everybody, so the Eigen faces will be the same for all. So, it is like the m cross n, because you know what you do not there is another way but then that will require too many images for each person which we cannot afford to have.

So, the psi is computed over the entire set again, this is not really dependent on a person, so psi comes from the entire set of face images, then let us say that let us say you have got like for this person I have got let us say u 1 1 minus m, m is again again coming from the whole class, so u 1 1 means image number 1 of this guy and then you will do v2 which will involve u 1 2 and so on. And then let us say I do this up to u 1 5, now what I can do is what is typically done is you take all these v1 v2 v3 v4 v5, which is with respect to the 5 images of this person that we have you take the average and then and then we can call this as let us say v1 or something, that will be the feature for this guy.

Similarly, you go to the second guy, you get some like v2 for this person, v3 and so on. So, we are like v1 v2 v3 for each person so it is like it is the feature of this guy which maybe which maybe which maybe of a much lower dimension than the original image, because this m depends upon how many eigenvectors you chose, so like I said at this can be as small as 100 dimensional. So, this can be just a 100 dimensional vector or even 50 dimensional vector.

And you take the average of all the features that you have got for that example and now you store all of this, so where each person, because you have captured all this a all this offline for each person you have captured all these images well well I ideally it cannot be 5 because if I have just n is 1000 then there only be 5000 images.

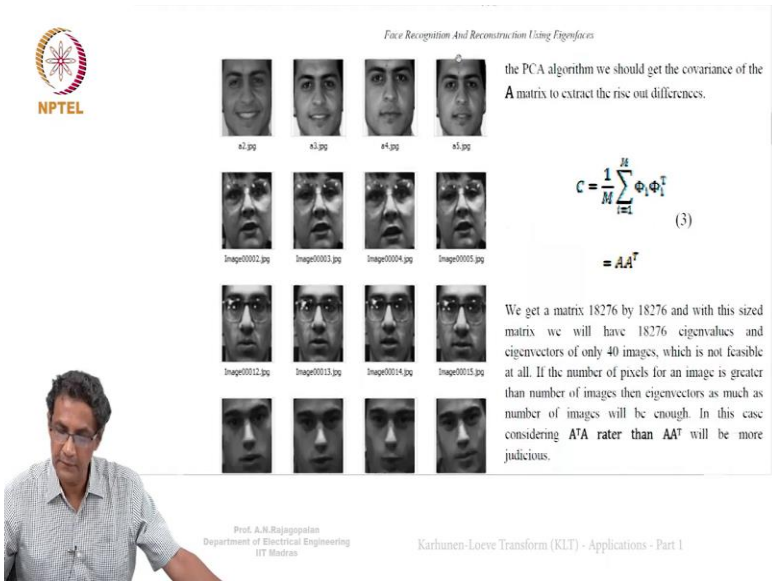
So, I assume that I have captured enough or maybe the n is large enough whatever, so that I get kind of P, so this number of individuals times the number of images that I have should be equal to P. Now, what you do is later now for example, you have stored all of this with you then someone else comes in, one of these guys comes in with a new face now, then you actually find out this v for that guy, again what will you do?

You will use the same psi and then you will be this new image sort of a test image that has come in and then m is still the same old m, that you had that you have already computed, then to compute this view, now you going to compare this v with respect to v1 v2 v3 which were it comes closest to, if it comes closest to let us say 3 in a simple kind of a nonsense v1 or whatever, so let us say v3 minus v, if this is if this happens to be the happens to be the smallest among all the I mean you compare with all the means, then you would say that he is that that person.

Now, this works very well then actually it is not such a bad idea it works very well if you give nice frontal faces, this struggles when there is the illumination variation, it struggles are now in there is a pose variation which is why it never you know it never flew far because they knew that there were some very very fundamental constraints you might say that why cannot we why cannot we take more images and all ask this guy to look this where or that where at that side this that they take but then you know people who will be willing to give you that many images.

So, illumination is again another issue for example shadows create problems and also expression somebody may give a new expression later and then it will so it was not robust, so good the main if you kind of used a constraint atmosphere, where you would ask may know people to straight away look at the camera do not give any expressions and so on then it was okay. But then those kind of things cannot be used so, I thought I will wanted to copy something, let me see if I have it here, so that you will get an idea to us how these things look like.

(Refer Slide Time: 15:22)



Face Recognition And Reconstruction Using Eigenfaces

the PCA algorithm we should get the covariance of the **A** matrix to extract the rise out differences.

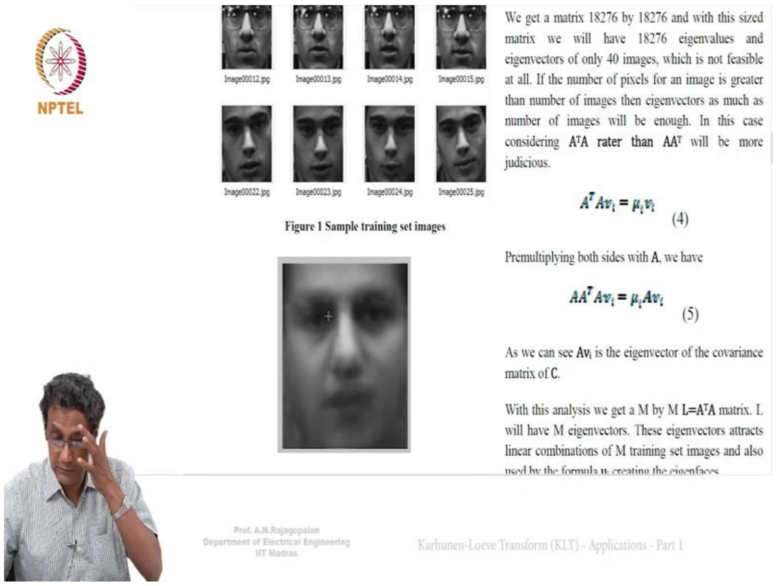
$$C = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T \quad (3)$$

$$= AA^T$$

We get a matrix 18276 by 18276 and with this sized matrix we will have 18276 eigenvalues and eigenvectors of only 40 images, which is not feasible at all. If the number of pixels for an image is greater than number of images then eigenvectors as much as number of images will be enough. In this case considering **A^T^A rater than AA^T^** will be more judicious.

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1



We get a matrix 18276 by 18276 and with this sized matrix we will have 18276 eigenvalues and eigenvectors of only 40 images, which is not feasible at all. If the number of pixels for an image is greater than number of images then eigenvectors as much as number of images will be enough. In this case considering **A^T^A rater than AA^T^** will be more judicious.

Figure 1 Sample training set images

$$A^TAv_i = \mu_i v_i \quad (4)$$

Premultiplying both sides with **A**, we have

$$AA^TAv_i = \mu_i Av_i \quad (5)$$

As we can see **Av**$_i$ is the eigenvector of the covariance matrix of **C**.

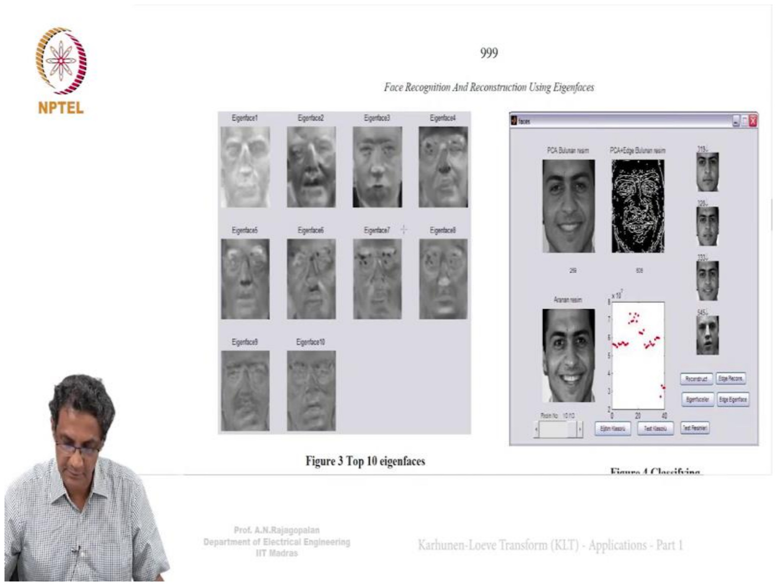
With this analysis we get a M by M **L=A**^{T}**A** matrix. L will have M eigenvectors. These eigenvectors attracts linear combinations of M training set images and also used by the formula $\mathbf{u}_i$ creating the eigenfaces

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

So, if you see these faces, so these are some people, so this images they have taken whatever male female all of them put together and then then then you construct something like an average face this is what I said this is your m, so this is like the average face.

(Refer Slide Time: 15:43)



999

Face Recognition And Reconstruction Using Eigenfaces

Figure 3 Top 10 eigenfaces

Figure 4 Classifying

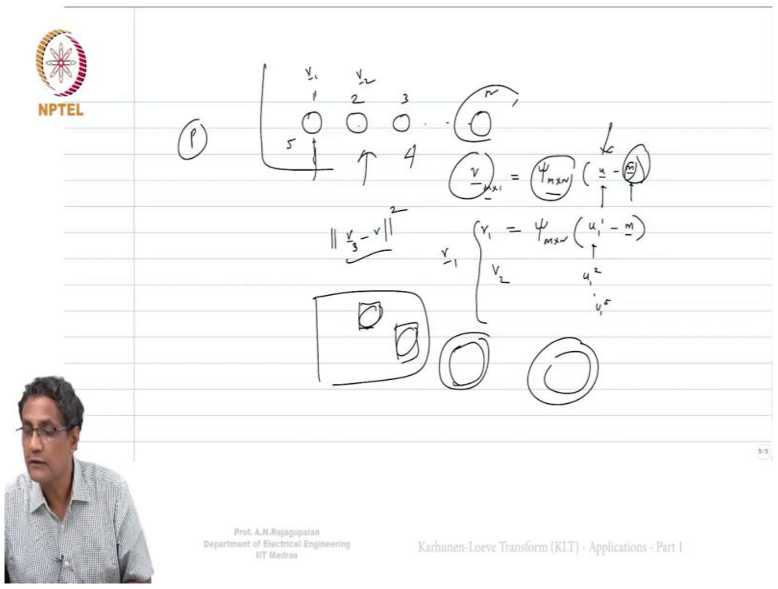
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

And then here are your Eigen faces, so I Eigen face because your mean subtracted them, so they look like a ghost face, so these are like this for example, the you know top Eigen face then the next most significant than the next most significant, so kind of order in some sense, in this case I do not know they have shown only first 10 Eigen faces and then so the plot that I was mentioning would be something like this where you would be able to show where these guys are at after you do when after you find the features you can reduce the dimensional and see how well separated they are and so on.

So, this as I said, this was, and then you know as long as you are willing to work in a constrained atmosphere constraint settings. Then I also wanted to show you right now another place where we had used it for paused here.

(Refer Slide Time: 16:38)


NPTEL
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras
Karhunen-Loeve Transform (KLT) - Applications - Part 1

So, until then people were using it for eigen, so this is face recognition what I said is Eigen face recognition, now you can also ask can I locate human people, so human faces, so suppose I have an image and suppose suppose I have somebody there somebody there normally human faces do not right they occur very sparsely in an image unless of course you are looking at a team you know the cricket team or something or a group photograph, otherwise normally when you look at look at an image there will be hardly like 2 or 3 people in an image mostly one person.

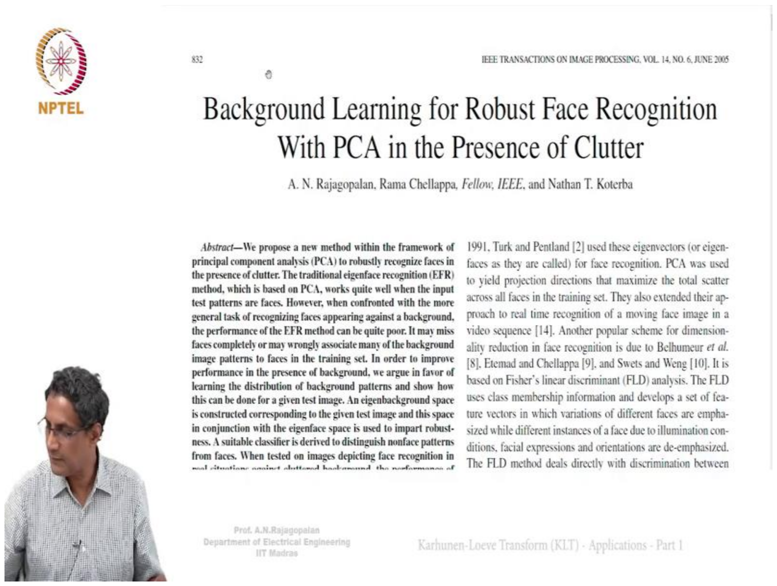
And then the idea is that can you use the PCA in order to be able to locate a face, where the where are human face is you are not try to recognize just want to know where it is, so this is also very important because it is in fact the first step before you can do kind of a recognition because you have to know where that face is and then you can probably pass that on to a this one recognize.

So, in that sense typically is like a first step, so what kind of say it, so people people what they are going to see did was they they they kind of learned what is called a universal background which is kind of difficult you know, I do not know I could never never except that it is like saying that whatever is not a face is all background.

Now, what is not a face is like, you know all over the place. Faces are that way very few, so there were groups that work on that kind of problem where where you know they would they would have a basket of images which should all be which they would all be which they would all classify as non faces and then they would have this basket where you have all faces and again again and they would kind of use the idea that I have I know how a background image looks like or a non-face looks like and then I know how her face looks like and based upon that I will try to do something. What happens is if you try to do something like that?

Then it hurts you because of the fact that some right like I told you I will show you I will show the next one, which will give you an idea about what will happen, if you try simply doing I am showing these just as you know just as additional applications I know that we are going to veering away from image transforms now, but that is now because you should not think that this is the only thing it is used for.

(Refer Slide Time: 18:52)



832 IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 14, NO. 6, JUNE 2005

Background Learning for Robust Face Recognition With PCA in the Presence of Clutter

A. N. Rajagopalan, Rama Chellappa, *Fellow, IEEE*, and Nathan T. Koterba

***Abstract*—We propose a new method within the framework of principal component analysis (PCA) to robustly recognize faces in the presence of clutter. The traditional eigenface recognition (EFR) method, which is based on PCA, works quite well when the input test patterns are faces. However, when confronted with the more general task of recognizing faces appearing against a background, the performance of the EFR method can be quite poor. It may miss faces completely or may wrongly associate many of the background image patterns to faces in the training set. In order to improve performance in the presence of background, we argue in favor of learning the distribution of background patterns and show how this can be done for a given test image. An eigenbackground space is constructed corresponding to the given test image and this space in conjunction with the eigenface space is used to impart robustness. A suitable classifier is derived to distinguish nonface patterns from faces. When tested on images depicting face recognition in**

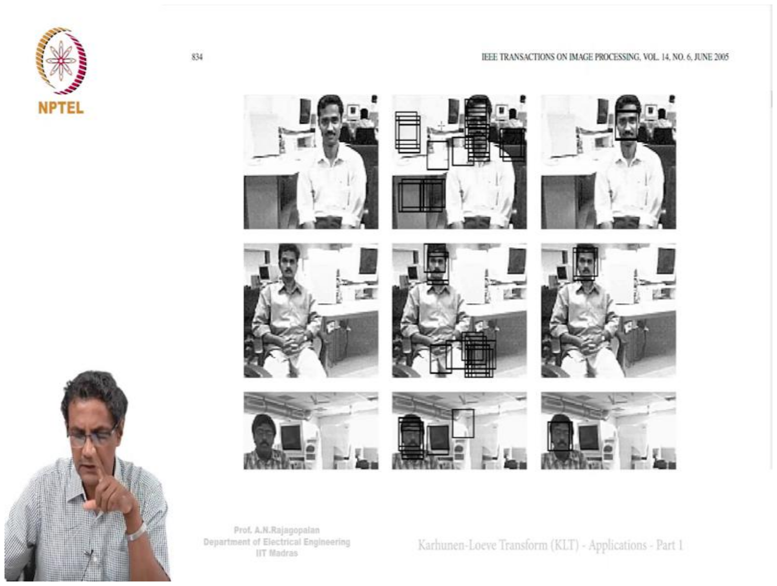
1991, Turk and Pentland [2] used these eigenvectors (or eigenfaces as they are called) for face recognition. PCA was used to yield projection directions that maximize the total scatter across all faces in the training set. They also extended their approach to real time recognition of a moving face image in a video sequence [14]. Another popular scheme for dimensionality reduction in face recognition is due to Belhumeur *et al.* [8], Etemad and Chellappa [9], and Swets and Weng [10]. It is based on Fisher's linear discriminant (FLD) analysis. The FLD uses class membership information and develops a set of feature vectors in which variations of different faces are emphasized while different instances of a face due to illumination conditions, facial expressions and orientations are de-emphasized. The FLD method deals directly with discrimination between

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

So for example, this was a paper that I look at when I send.

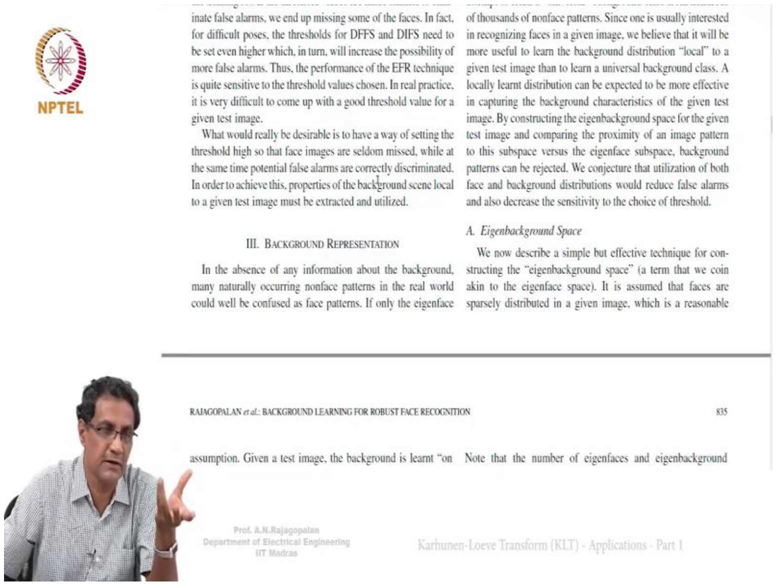
(Refer Slide Time: 18:56)


834
IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 14, NO. 6, JUNE 2005
NPTEL
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras
Karhunen-Loeve Transform (KLT) - Applications - Part 1

But this guy did some very good work. So, the idea was this, so if you try to use something like that, normally you will have like people 1 or kind of 2 people in a scene, what will happen is, it will start to classify even other things as faces, but even if it puts see these multiple boxes, you can remove you can do what is called a non-maximum separation remove, so you will get maybe 3 or 4 box, so you do not need to see so many overlapping ones, but the idea is that even if you show one extra, it looks very jarring because there is only 1 person and then if you say that is that guy and then and then is 1 more guy and if you say that this is a human face, then you will wonder what the hell is this algorithm doing then.

So, you have to be very accurate, so when you especially when you localized faces you cannot afford to throw up something which is not a face and call it a face. So, as I said, so learning kind of a universal background did not at all appeal at least to me, so what we did was we learnt a background with respect to that particular image, that that seems to make a lot more sense to me, why should I worry about what the rest of the universe looks like, if I have this image and if had to locate a face in that image one had to simply worried about what is the background in that image and learn that.

(Refer Slide Time: 20:05)



inate false alarms, we end up missing some of the faces. In fact, for difficult poses, the thresholds for DFFS and DIFS need to be set even higher which, in turn, will increase the possibility of more false alarms. Thus, the performance of the EFR technique is quite sensitive to the threshold values chosen. In real practice, it is very difficult to come up with a good threshold value for a given test image.

What would really be desirable is to have a way of setting the threshold high so that face images are seldom missed, while at the same time potential false alarms are correctly discriminated. In order to achieve this, properties of the background scene local to a given test image must be extracted and utilized.

III. Background Representation

In the absence of any information about the background, many naturally occurring nonface patterns in the real world could well be confused as face patterns. If only the eigenface

of thousands of nonface patterns. Since one is usually interested in recognizing faces in a given image, we believe that it will be more useful to learn the background distribution "local" to a given test image than to learn a universal background class. A locally learnt distribution can be expected to be more effective in capturing the background characteristics of the given test image. By constructing the eigenbackground space for the given test image and comparing the proximity of an image pattern to this subspace versus the eigenface subspace, background patterns can be rejected. We conjecture that utilization of both face and background distributions would reduce false alarms and also decrease the sensitivity to the choice of threshold.

A. Eigenbackground Space

We now describe a simple but effective technique for constructing the "eigenbackground space" (a term that we coin akin to the eigenface space). It is assumed that faces are sparsely distributed in a given image, which is a reasonable

RAJAGOPALAN *et al.*: BACKGROUND LEARNING FOR ROBUST FACE RECOGNITION 835

assumption. Given a test image, the background is learnt "on

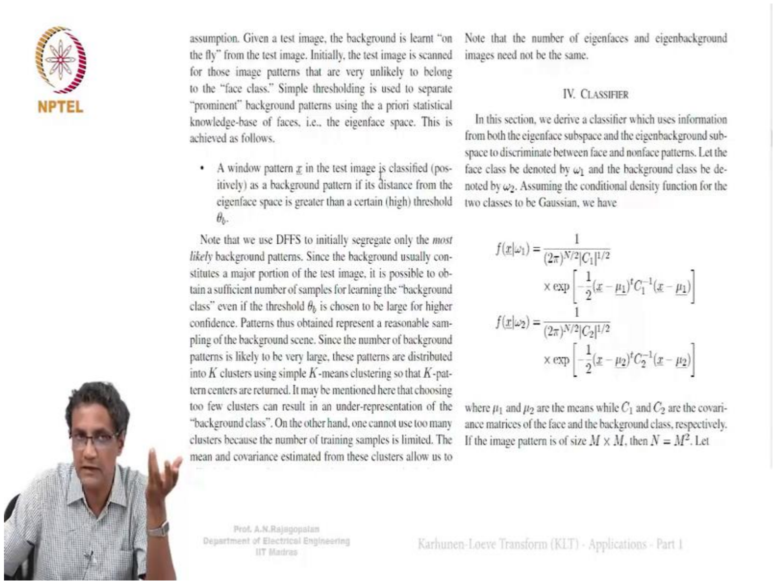
Note that the number of eigenfaces and eigenbackground

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

But if you do that that right then what you have is something like an Eigen background space and then you have what is called an Eigen face space, the face space is still the same thing, whatever we did for face recognition all those faces you can use to get an idea about how the Eigen faces look like, but for Eigen background, you have to pick patterns from this image from its own background.

(Refer Slide Time: 20:23)



assumption. Given a test image, the background is learnt "on the fly" from the test image. Initially, the test image is scanned for those image patterns that are very unlikely to belong to the "face class." Simple thresholding is used to separate "prominent" background patterns using the a priori statistical knowledge-base of faces, i.e., the eigenface space. This is achieved as follows.

- A window pattern $\underline{x}$ in the test image is classified (positively) as a background pattern if its distance from the eigenface space is greater than a certain (high) threshold θ_b.

Note that we use DFFS to initially segregate only the *most likely* background patterns. Since the background usually constitutes a major portion of the test image, it is possible to obtain a sufficient number of samples for learning the "background class" even if the threshold θ_b is chosen to be large for higher confidence. Patterns thus obtained represent a reasonable sampling of the background scene. Since the number of background patterns is likely to be very large, these patterns are distributed into K clusters using simple K-means clustering so that K-pattern centers are returned. It may be mentioned here that choosing too few clusters can result in an under-representation of the "background class". On the other hand, one cannot use too many clusters because the number of training samples is limited. The mean and covariance estimated from these clusters allow us to

Note that the number of eigenfaces and eigenbackground images need not be the same.

IV. Classifier

In this section, we derive a classifier which uses information from both the eigenface subspace and the eigenbackground subspace to discriminate between face and nonface patterns. Let the face class be denoted by ω_1 and the background class be denoted by ω_2. Assuming the conditional density function for the two classes to be Gaussian, we have

$$f(\underline{x}|\omega_1) = \frac{1}{(2\pi)^{N/2}|C_1|^{1/2}} \times \exp\left[-\frac{1}{2}(\underline{x}-\underline{\mu_1})^t C_1^{-1}(\underline{x}-\underline{\mu_1})\right]$$
$$f(\underline{x}|\omega_2) = \frac{1}{(2\pi)^{N/2}|C_2|^{1/2}} \times \exp\left[-\frac{1}{2}(\underline{x}-\underline{\mu_2})^t C_2^{-1}(\underline{x}-\underline{\mu_2})\right]$$

where μ_1 and μ_2 are the means while C_1 and C_2 are the covariance matrices of the face and the background class, respectively. If the image pattern is of size $M \times M$, then $N = M^2$. Let

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

Now, which is actually tricky because you do not know where they face is.

(Refer Slide Time: 20:30)

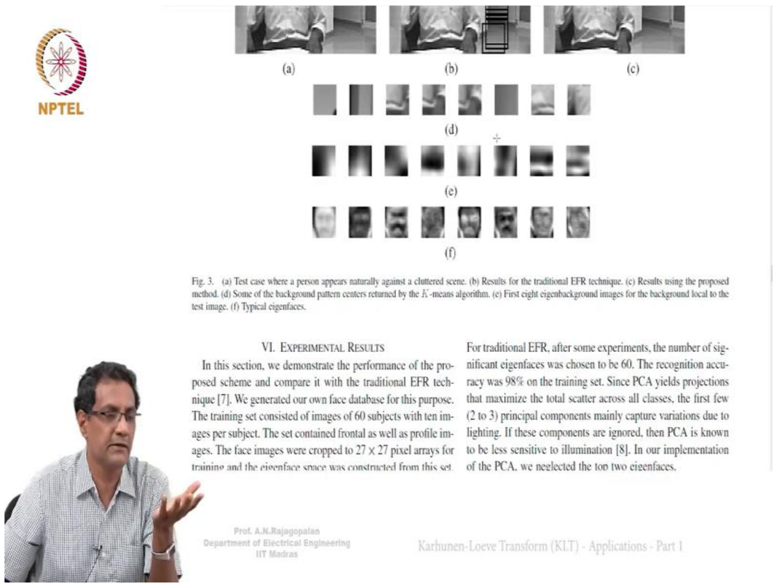


Fig. 3. (a) Test case where a person appears naturally against a cluttered scene. (b) Results for the traditional EFR technique. (c) Results using the proposed method. (d) Some of the background pattern centers returned by the K-means algorithm. (e) First eight eigenbackground images for the background local to the test image. (f) Typical eigenfaces.

VI. Experimental Results

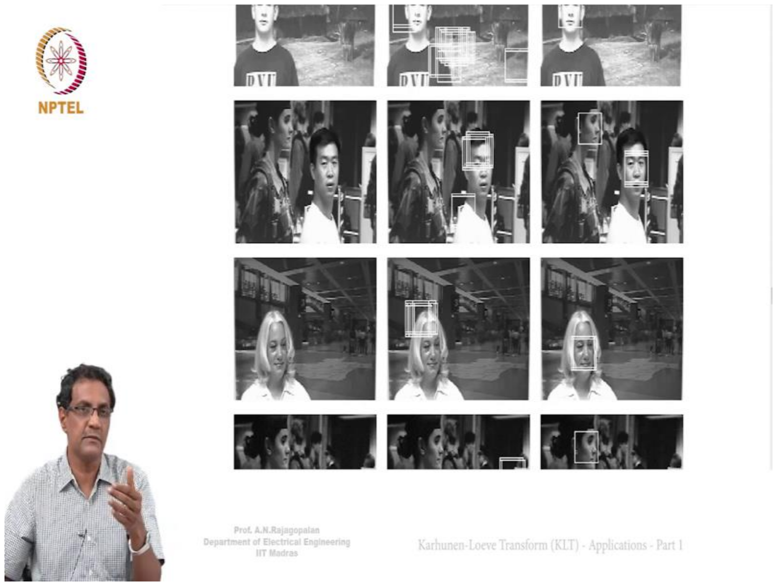
In this section, we demonstrate the performance of the proposed scheme and compare it with the traditional EFR technique [7]. We generated our own face database for this purpose. The training set consisted of images of 60 subjects with ten images per subject. The set contained frontal as well as profile images. The face images were cropped to 27×27 pixel arrays for training and the eigenface space was constructed from this set. For traditional EFR, after some experiments, the number of significant eigenfaces was chosen to be 60. The recognition accuracy was 98% on the training set. Since PCA yields projections that maximize the total scatter across all classes, the first few (2 to 3) principal components mainly capture variations due to lighting. If these components are ignored, then PCA is known to be less sensitive to illumination [8]. In our implementation of the PCA, we neglected the top two eigenfaces.

Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

So, then what then what we can show is, see these Eigen background images will look like that, this is with respect to some particular image, again it will change if you change the image. So, it will be you know it is like learning background on the fly, you change the you change the image you give me another image then then my again background will change my faces will remain the same, I do not change my Eigen face but my Eigen background keeps changing.

(Refer Slide Time: 20:50)



Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras

Karhunen-Loeve Transform (KLT) - Applications - Part 1

And therefore, now what you can do is you know earlier if you had a result like this then now you would have a result like that, where where the boxes is exactly on where there is a person, so here for example it puts an additional box, here the plenty of boxes there overlapping once of course, again here it is not even putting it correctly on this guy whereas you know, it comes back on and actually in those days people never talked about sparsity, the word sparsity was not actually used.

In fact that was the thing which was actually exploited, what was done was in this case, it was simply a reconstruction error that we had to find out and what we realized was if you fix the fix the basis images to be few like for example, if you say that Eigen background, I literally only used 30 and similar Eigen faces also, I will use only so many 30 let us say. then the idea was that if you had a face somewhere then your ability to reconstruct face with eigenfaces is far higher is far superior than let us say when you use a few of these bases images.

It it is very, you know, it is if you start increasing the number of eigenvectors that you are starting that you start to use then if you sort of throw away this notion of sparsity, then it becomes harder and harder, but if you say that I will only use a few of them which is like sparsity we never use the words sparsity even if you read that paper never was the word sparsity used, but then later on people ended up showing exactly something similar.

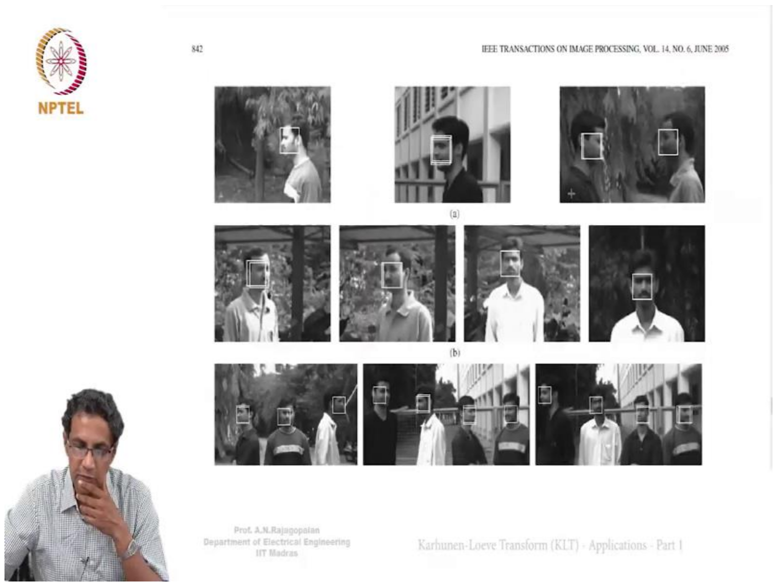
So, the idea is that if you just use a few eigenvectors, then your ability to reconstruct that particular class is very high. So, for example, if I use a background image and if I try because I do not know right whether it is a background or whether it is a face, so I kind of reconstructed using my Eigen background images 30 of them, I also reconstructed using my Eigen faces again let us say so many of them 30 of them, now because it is a background, the Eigen background can actually favourably reconstructed much better than what the face can, because because I am using only so many.

So, if you if you throw in this kind of notion of sparsity and try to reconstruct, it will just show up. So, so earlier if you had a problem earlier the problem was if you had only one sort of single metric where you know you had only face and then you had an image you did not know whether it was a background or a face, for face you got a value for let us say you know so for example, if you said that if it is closer than this value, I will declare it as a face then what would what would happen is a face with a minor variation it will end up having a slightly higher error and then you would say it is not a face.

So, what will happen is either you end up declaring too many things as faces even things that are not a face you declare along with faces you declare them also as faces which is wrong or if you make this very tight then what will happen is you will end up showing you will end up not even showing faces as faces, you will miss. So, so this kind of a threshold setting became a problem, whereas, with this kind of thing what happens you have this Eigen background on this side setting the Eigen faces sitting now you are not only computing one one sort of a one value one sort of metric, you also trying to find out what is it respective background.

What happens if this guy turns out to be a background then it is error with respect to background Eigen face reconstruction is very small compared to with faces, so you have some notion of how well you are doing, that is what was used and.

(Refer Slide Time: 24:07)


842
IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 14, NO. 6, JUNE 2005
(a)
(b)
Prof. A.N.Rajagopalan
Department of Electrical Engineering
IIT Madras
Karhunen-Loeve Transform (KLT) - Applications - Part 1

Then you know we can even show it, I think I am even other examples or even we took it here it is all there, but even we came back here and took it outside our building here outside EHB so multiple people and all so did pretty well and of course we train even for sides faces and all. We did and wan to say restricted to so even here this is not front till in fact totally side, so all of that it could pick and you see that right there are hardly any false false from pictures the false boxes.