IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p-ISSN: 2278-8735
PP 61-71
www.iosrjournals.org

Face Anti-Spoofing using Robust Features and Fisher Vector Encoding based Innovative Real Time security system for Automobile Applications

R.Jayashalini[1], Ms.S.Priyadharshini[2] (Asst.Professor)
[1]*(M.E/ECE, M.A.M College of Engineering/ India)*
[2]*(M.E/ECE, M.A.M College of Engineering/ India)*

Abstract: *The vulnerabilities of face biometric authentication systems to spoofing attacks have received a significant attention during the recent years. Some of the proposed countermeasures have achieved impressive results when evaluated on intra-tests i.e. the system is trained and tested on the same database. Unfortunately, most of these techniques fail to generalize well to unseen attacks e.g. when the system is trained on one database and then evaluated on another database. This is a major concern in biometric anti-spoofing research which is mostly overlooked. In this paper, we propose a novel solution based on describing the facial appearance by applying Fisher Vector encoding on Speeded-Up Robust Features (SURF) extracted from from different color spaces. The evaluation of our countermeasure on three challenging benchmark face spoofing databases, namely the CASIA Face Anti-Spoofing Database, the Replay- Attack Database and MSU Mobile Face Spoof Database, showed excellent and stable performance across all the three datasets. Most importantly, in inter-database tests, our proposed approach outperforms the state of the art and yields in very promising generalization capabilities, even when only limited training data is used.*

I. Introduction

Nowadays, the security of face recognition systems is challenged. A face image printed on a paper can spoof the systems and then the access is granted to attackers. A practical face recognition system demands not only high recognition performance, but also the capability of anti-spoofing to differentiate faces from real persons (genuine face) and those from attackers (fake face). Recently, many papers on face anti-spoofing have been published. Meanwhile, competitions further promoted the development. In all these works, a generic classifier was used to detect spoofing attacks on all subjects. Because samples from different subjects have different distributions, it is difficult to obtain a generic classifier to well detect various spoofing attacks on all subjects. In person-specific anti-spoofing, the most challenging issue is the limited number of samples for training, especially the fake samples. In our framework, we develop anti-spoofing model for each enrolled subject specifically. However, many enrolled subjects have no fake samples in practice. To train face anti-spoofing models for these subjects, we propose a subject domain adaptation method to transfer the information provided by the subjects which have both genuine samples and fake samples (source subjects) to the subjects having no fake samples (target subjects) to synthesize fake samples .

II. Material And Methods

In this paper, we propose a novel solution based on describing the facial appearance by applying Fisher Vector encoding on Speeded-Up Robust Features (SURF) extracted from from different color spaces. The evaluation of our countermeasure on three challenging benchmark face spoofing databases, namely the CASIA Face Anti-Spoofing Database, the Replay- Attack Database and MSU Mobile Face Spoof Database, showed excellent and stable performance across all the three datasets. Most importantly, in inter-database tests, our proposed approach outperforms the state of the art and yields in very promising generalization capabilities, even when only limited training data is used

A. The Speeded-Up Robust Features (SURF)

The Speeded-Up Robust Features (SURF) is a fast and efficient scale and rotation invariant descriptor. It was originally proposed to reduce the computational complexity of the Scale Independent Feature Transform (SIFT) descriptor . Instead of using the Difference of Gaussian (DoG) filters to approximate the Laplacian of Gaussian, the SURF descriptor uses the Haar box filters. A convolution with these box filters can be computed rapidly by utilizing integral images. The SURF descriptor is obtained using the Wavelet responses in the horizontal and vertical directions.

The region around each interest point is first divided into 4 × 4 subregions. Then, for each sub-region j, the horizontal and vertical Wavelet responses are used to form a feature vector Vj as follows:

$$V_j = [\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|]. \qquad (1)$$

Where dx and dy are the Haar wavelet responses in the horizontal and vertical directions, respectively. The feature vectors extracted from each sub-region are concatenated to from a SURF descriptor with 64 dimensions:

$$SURF = [V_1, ..., V_{16}]. \qquad (2)$$

The SURF descriptor was originally proposed for grayscale images. Inspired by our previous finding showing the importance of the color texture in face antispoofing, we propose to extract the SURF features from the color images instead of the gray-scale representation. First, the SURF descriptor is applied on each color band separately. Then, the obtained features are concatenated to form a single feature vector (referred to as CSURF). Finally, Principal Component Analysis (PCA) is applied to de-correlate the obtained feature vector and reduce the dimensionality of the face description.

B. Fisher Vector (FV)

Extracting dense features has shown to be an essential component in many computer vision applications . In Fisher Vector (FV) encoding was shown to perform very well in many image recognition benchmarks. FV embeds a set of feature vectors into a high dimensional space more amenable to linear classification. The feature vectors are obtained by fitting a generative parametric model, e.g. Gaussian Mixture Model (GMM), to the features to be encoded. Let X = {xt, t = 1, ..., T} be a D-dimensional local descriptors extracted from a face Image I and let _ = {µk, _k,wk, k = 1, ...,M} be the means, the covariance matrices and the weights of the GMM model _ trained with a large set of local descriptors. The derivations of the model with respect of the mean and the covariance parameters (Equation 3 and 4) capture the first and the second order differences between the features X and each of the GMM components.

$$\phi_k^1 = \frac{1}{T\sqrt{w_k}} \sum_{t=1}^{T} \alpha_t(k)(\frac{x_t - \mu_k}{\sigma_k}) \qquad (3)$$

$$\phi_k^2 = \frac{1}{T\sqrt{2w_k}} \sum_{t=1}^{T} \alpha_t(k)[\frac{(x_t - \mu_k)^2}{\sigma_k^2} - 1], \qquad (4)$$

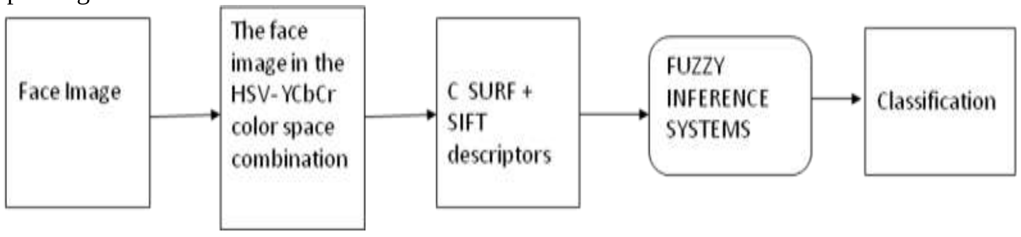
where, t(k) is the soft assignment weight of the feature xt to the GMM component k:

$$\alpha_t(k) = \frac{w_k u_k(x_t)}{\sum_{j=1}^{M} w_j u_j(x_t)} \qquad (5)$$

Here, ui denote the probability density function of the Gaussian component i. The concatenation of these two order differences represent the Fisher Vector of the image I described by its local descriptors X. The dimensionality of this vector is 2MD.

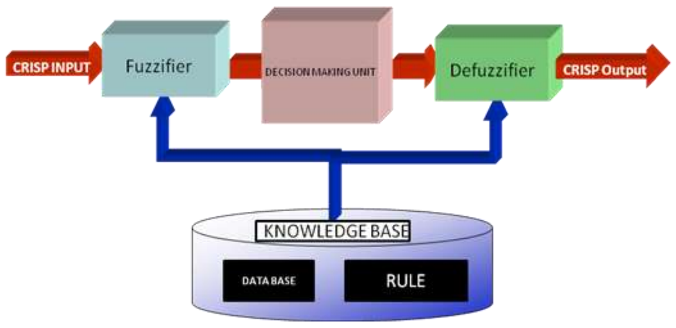
Procedure methodology

A Fisher vector represents how the distribution of the local descriptors X differ from the distribution of the GMM model trained with all the training images. To further improve the performance, the Fisher vectors are normalized using a square rooting followed by L2 normalization. Figure 1 depict the general block diagram of our face spoofing detection method


Face Image
The face image in the HSV- YCbCr color space combination
C SURF + SIFT descriptors
FUZZY INFERENCE SYSTEMS
Classification

.

FUZZY INFERENCE SYSTEMS



Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made, or patterns discerned. The process of fuzzy inference involves all of the pieces that are described in the previous sections: Membership Functions, Logical Operations, and If-Then Rules. You can implement two types of fuzzy inference systems in the toolbox: Mamdani-type and Sugeno-type. These two types of inference systems vary somewhat in the way outputs are determined. Fuzzy inference systems have been successfully applied in fields such as automatic control, data classification, decision analysis, expert systems, and computer vision. Because of its multidisciplinary nature, fuzzy inference systems are associated with a number of names, such as fuzzy-rule-based systems, fuzzy expert systems, fuzzy modeling, fuzzy associative memory, fuzzy logic controllers, and simply (and ambiguously) fuzzy systems. Fuzzy inference process comprises of five parts: fuzzification of the input variables, application of the fuzzy operator (AND or OR) in the antecedent, implication from the antecedent to the consequent, aggregation of the consequents across the rules, and defuzzification. These sometimes cryptic and odd names have very specific meaning that is defined in the following steps which is taken for the sample conditions.

Step 1. Fuzzify Inputs

The first step is to take the inputs and determine the degree to which they belong to each of the appropriate fuzzy sets via membership functions. In Fuzzy Logic Toolbox software, the input is always a crisp numerical value limited to the universe of discourse of the input variable (in this case the interval between 0 and 10) and the output is a fuzzy degree of membership in the qualifying linguistic set (always the interval between 0 and 1). Fuzzification of the input amounts to either a table lookup or a function evaluation.

This example is built on three rules, and each of the rules depends on resolving the inputs into a number of different fuzzy linguistic sets: service is poor, service is good, food is rancid, food is delicious, and so on. Before the rules can be evaluated, the inputs must be fuzzified according to each of these linguistic sets. For example, to what extent is the food really delicious? The following figure shows how well the food at thehypothetical restaurant (rated on a scale of 0 to 10) qualifies, (via its membership function), as the linguistic variable delicious. In this case, we rated the food as an 8, which, given your graphical definition of delicious, corresponds to μ = 0.7 for the delicious membership function

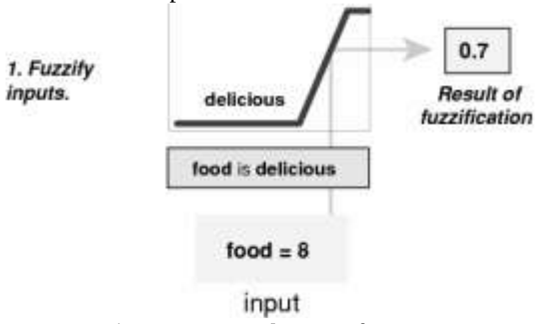


Figure 2 -Sample Fuzzify Inputs

In this manner, each input is fuzzified over all the qualifying membership functions required by the rules.

Step 2. Apply Fuzzy Operator

After the inputs are fuzzified, you know the degree to which each part of the antecedent is satisfied for each rule. If the antecedent of a given rule has more than one part, the fuzzy operator is applied to obtain one number that represents the result of the antecedent for that rule. This number is then applied to the output function. The input to the fuzzy operator is two or more membership values from fuzzified input variables. The output is a single truth value.

As is described in Logical Operations section, any number of well-defined methods can fill in for the AND operation or the OR operation. In the toolbox, two built in AND methods are supported: min (minimum) and prod (product). Two built-in OR methods are also supported: max (maximum), and the probabilistic OR method probor. The probabilistic OR method (also known as the algebraic sum) is calculated according to the equation

$$\mathbf{probor(a,b) = a + b - ab}$$

In addition to these built-in methods we can create your own methods for AND and OR by writing any function and setting that to be your method of choice.

The following figure shows the OR operator max at work, evaluating the antecedent of the rule 3 for the tipping calculation. The two different pieces of the antecedent (service is excellent and food is delicious) yielded the fuzzy membership values 0.0 and 0.7 respectively. The fuzzy OR operator simply selects the maximum of the two values, 0.7, and the fuzzy operation for rule 3 is complete. The probabilistic OR method would still result in 0.7.

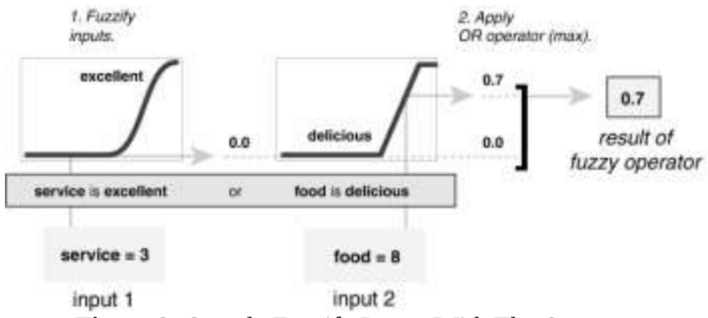


Figure 3- Sample Fuzzify Inputs With The Operator

Step 3. Apply Implication Method

Before applying the implication method, we must determine the rule's weight. Every rule has a weight (a number between 0 and 1), which is applied to the number given by the antecedent. Generally, this weight is 1 (as it is for this example) and thus has no effect at all on the implication process. From time to time you may want to weight one rule relative to the others by changing its weight value to something other than 1. After proper weighting has been assigned to each rule, the implication method is implemented. A consequent is a fuzzy set represented by a membership function, which weights appropriately the linguistic characteristics that are attributed to it. The consequent is reshaped using a function associated with the antecedent (a single number). The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Implication is implemented for each rule. Two built-in methods are supported, and they are the same functions that are used by the AND method: min (minimum), which truncates the output fuzzy set, and prod (product), which scales the output fuzzy set.

Step 4. Aggregate All Outputs

Because decisions are based on the testing of all of the rules in a FIS, the rules must be combined in some manner in order to make a decision. Aggregation is the process by which the fuzzy sets that represent the outputs of each rule are combined into a single fuzzy set. Aggregation only occurs once for each output variable, just prior to the fifth and final step, defuzzification. The input of the aggregation process is the list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable. As long as the aggregation method is commutative (which it always should be), then the order in which the rules are executed is unimportant. Three built-in methods are supported:

- max (maximum) ·
- probor (probabilistic OR) ·
- sum (simply the sum of each rule's output set)

In the following diagram, all three rules have been placed together to show how the output of each rule is combined, or aggregated, into a single fuzzy set whose membership function assigns a weighting for every output (tip) value

Step 5. Defuzzify

The input for the defuzzification process is a fuzzy set (the aggregate output fuzzy set) and the output is a single number. As much as fuzziness helps the rule evaluation during the intermediate steps, the final desired output for each variable is generally a single number. However, the aggregate of a fuzzy set encompasses a range of output values, and so must be defuzzified in order to resolve a single output value from the set. Perhaps the most popular defuzzification method is the centroid calculation, which returns the center of area under the curve. There are five built-in methods supported: centroid, bisector, middle of maximum (the average of the maximum value of the output set), largest of maximum, and smallest of maximum.

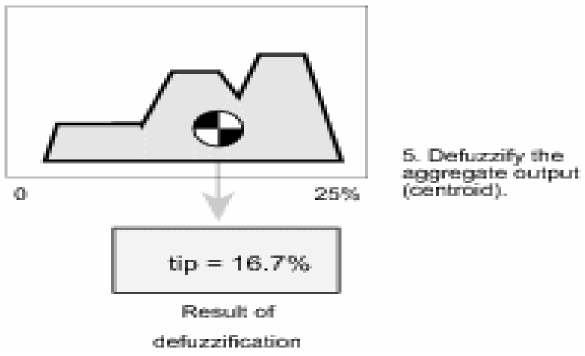

0
25%
5. Defuzzify the aggregate output (centroid).
tip = 16.7%
Result of defuzzification

Figure 4- De-fuzzification sample

III. Result

Effect of the color information

We begin our experiments by first evaluating the importance of the color SURF features (referred to as CSURF) compared to the gray-scale SURF features (referred to as SURF). In these experiments, we extracted the CSURF features from three color spaces: RGB, HSV and YCbCr. For the sake of simplicity, no dimensionality reduction or feature encoding; technique was applied at this point. These results clearly indicate the importance of CSURF descriptions compared to the original SURF descriptions extracted from the gray-scale images. Comparing the results obtained using the different color spaces, we observe that using HSV and YCbCr color spaces yields in better performance compared to the RGB color space. This confirms the importance of using separated luminance and chrominance color spaces. As the color (luminance and the chrominance) information in the HSV and YCbCr color spaces are different, we propose to fuse the features extracted from these two color spaces in order to benefit from their potential complementarity. As shown in tables I and II, this fusion improves the performance in both intra-database (except on MSU database where the use of the HSV color space gives the best performance) and inter-database scenarios compared to the performance obtained using each color space separately. In the rest of our experiments, the CSURF features are extracted from the combined HSV-YCbCr color space.

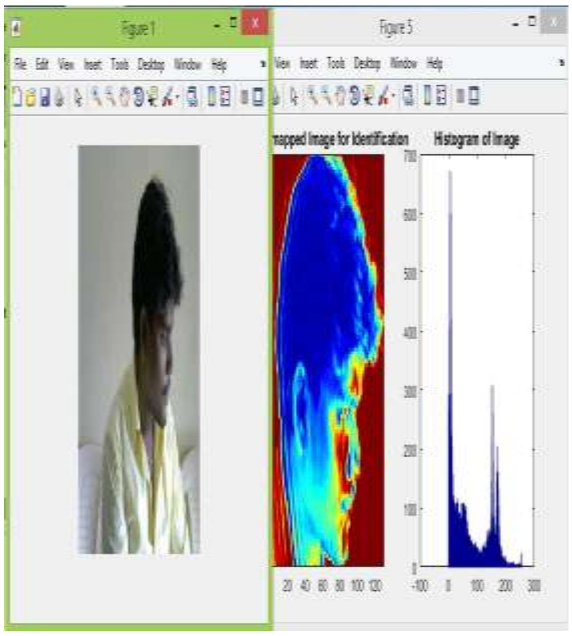


Figure 1
Figure 5
File Edit View Insert Tools Desktop Window Help
mapped Image for Identification
Histogram of Image

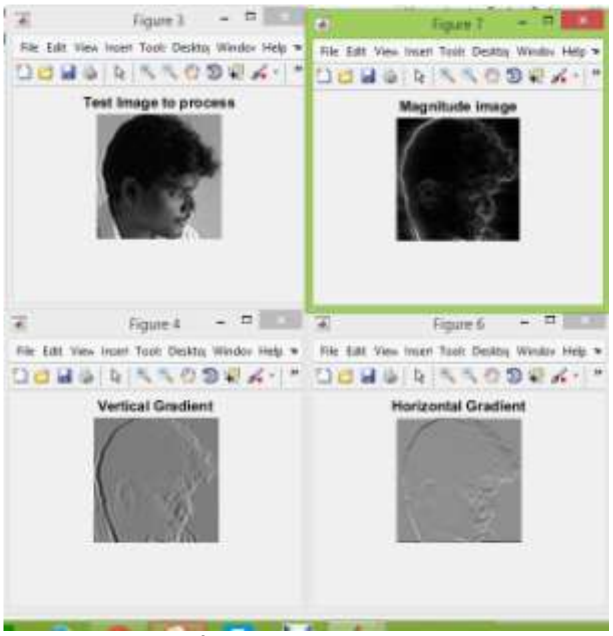
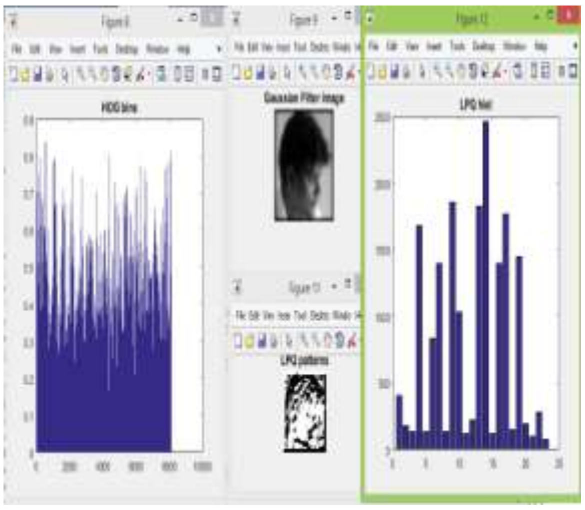


Fig 5- Input image and cropped image with histogram **Fig 6-** Image gradient of vertical and horizontal along with HOG



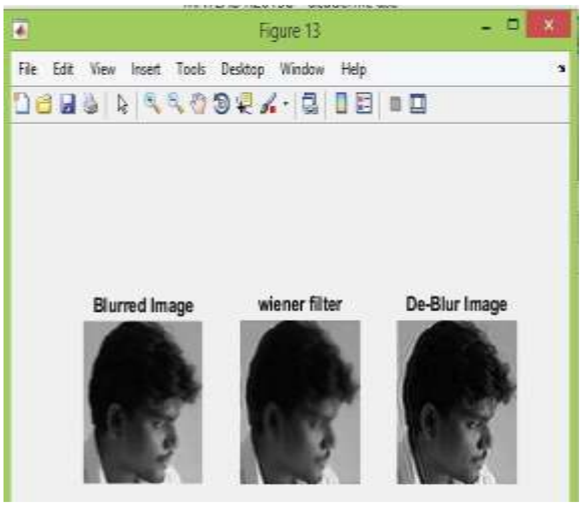
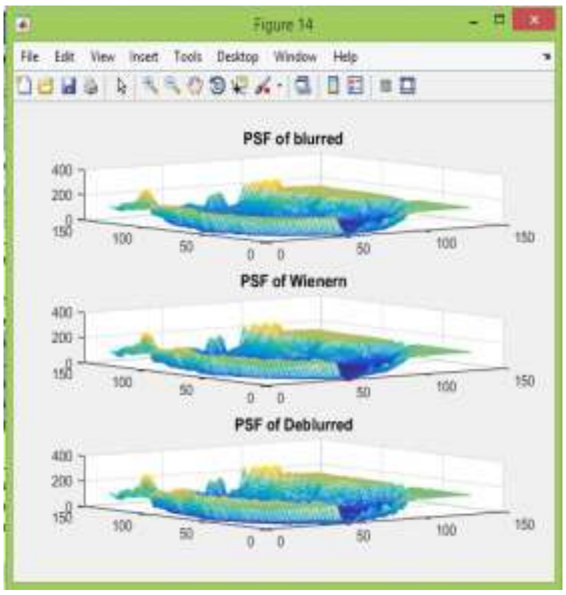


Fig 7- Extracted Features of HOG and LPQ pattern with histogram **Fig 8-** Image gradient of vertical and horizontal along with HOG



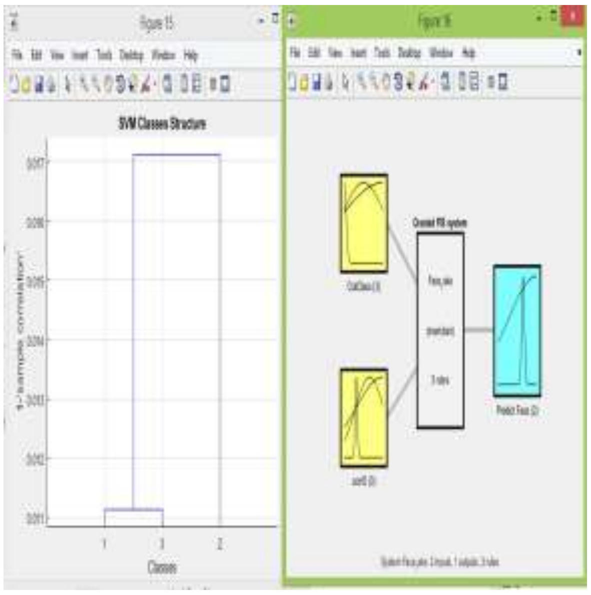
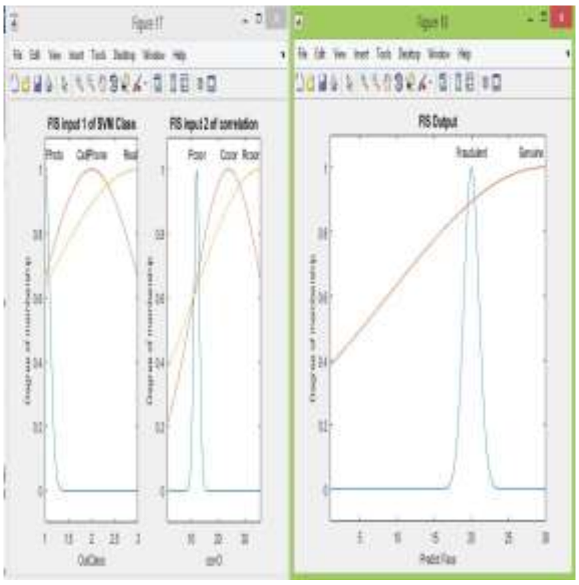


Fig 9- Noisy content PSF of blurred and Deblurred image **Fig 10-** SVM class prediction and FIS Class assignments



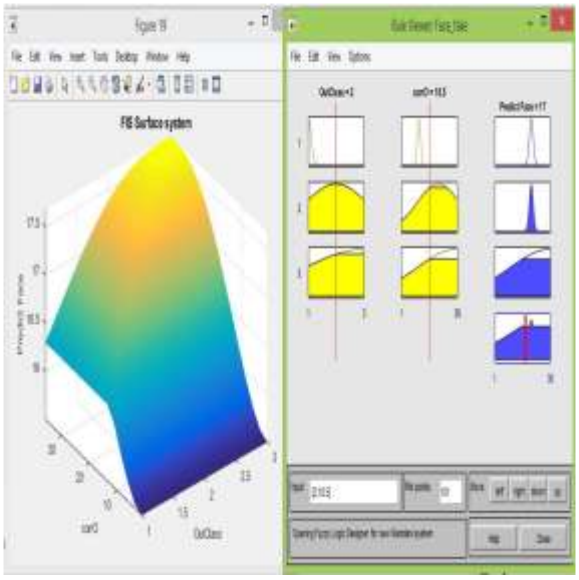
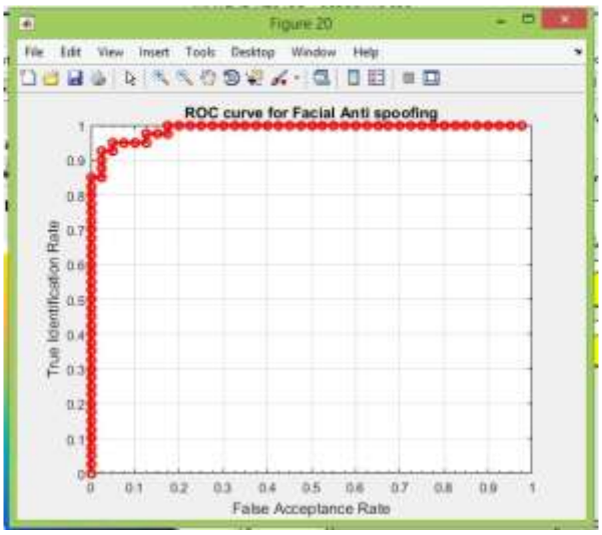


Fig 11- Membership function of FIS two inputs and one output **Fig 12**- FIS classification Layer Surface and Rule's List for each class



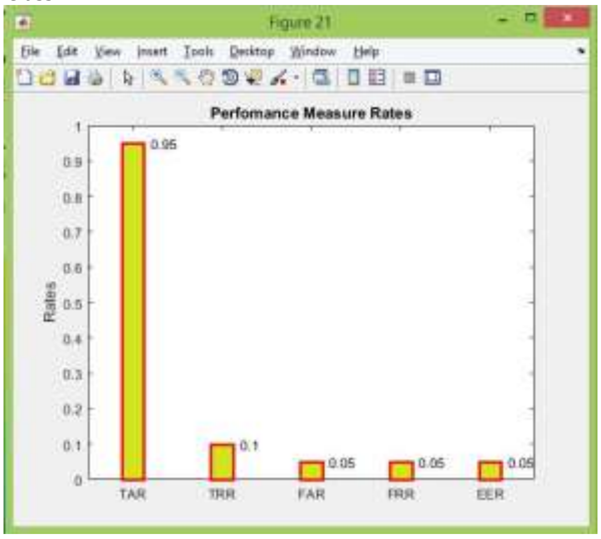


Fig 13- ROC curve for facial antispoofing **Fig 14**- Rates Measure of Classification

Table 1: Training Database for Genuine Object Faces.

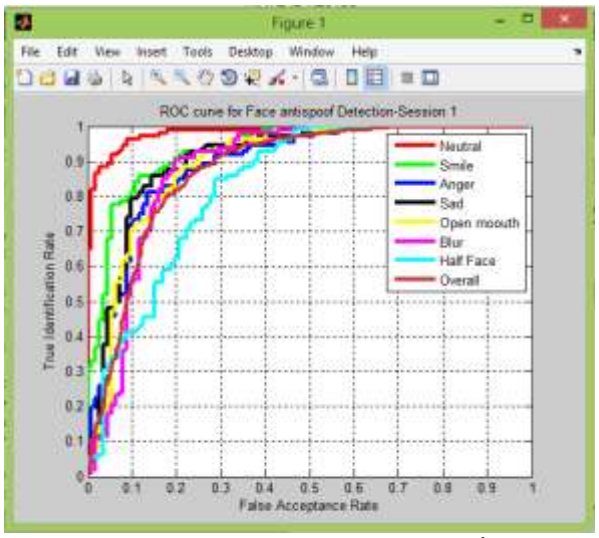
Training Database for Genuine Object Faces								
Session	**Ideal**	**smiley**	**Anger**	**Sad**	**Open mouth**	**Blur**	**half face**	**overall**
session 01	5	15	7	12	6	10	15	80
session 02	7	8	16	5	18	2	6	62
session 03	14	3	3	9	2	14	5	50
Total	**26**	**26**	**26**	**26**	**26**	**26**	**26**	**182**

Table 2 : Training Database for Photo Print attack Object Faces

Training Database for Photo Print attack Object Faces								
Session	**Ideal**	**smiley**	**Anger**	**Sad**	**Open mouth**	**Blur**	**half face**	**overall**
session 01	24	17	12	15	16	10	12	106
session 02	10	14	15	11	18	22	16	106
session 03	12	15	19	20	12	14	18	106
Total	**46**	**46**	**46**	**46**	**46**	**46**	**46**	**322**

Table 3 : Training Database for Video Reply attack Object Faces

Training Database for Video Reply attack Object Faces								
Session	**Ideal**	**smiley**	**Anger**	**Sad**	**Open mouth**	**Blur**	**half face**	**overall**
session 01	22	15	15	15	14	15	11	107
session 02	10	12	13	11	18	12	16	92
session 03	8	15	20	18	13	17	13	104
Total	**40**	**42**	**48**	**44**	**45**	**44**	**40**	**303**



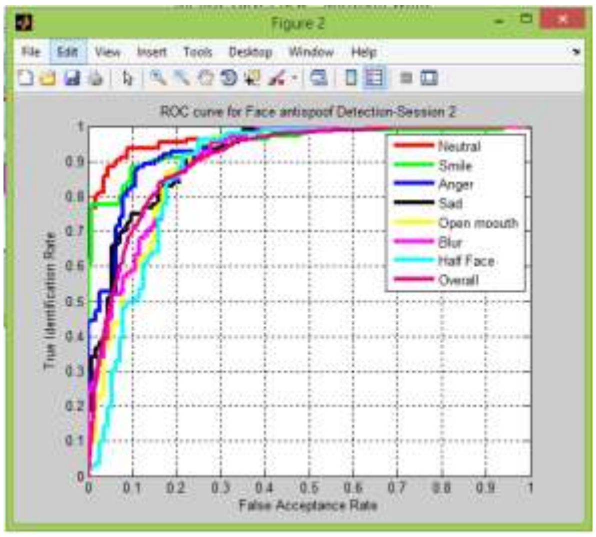


Fig 15- ROC curve for session **1** **Fig 16**- ROC curve for session 2

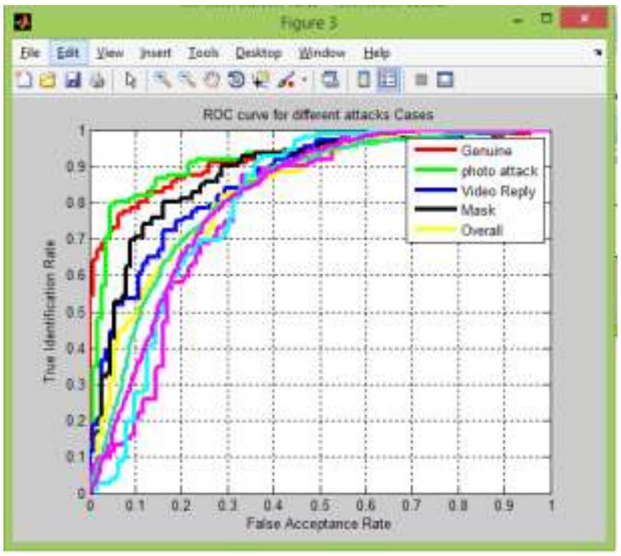
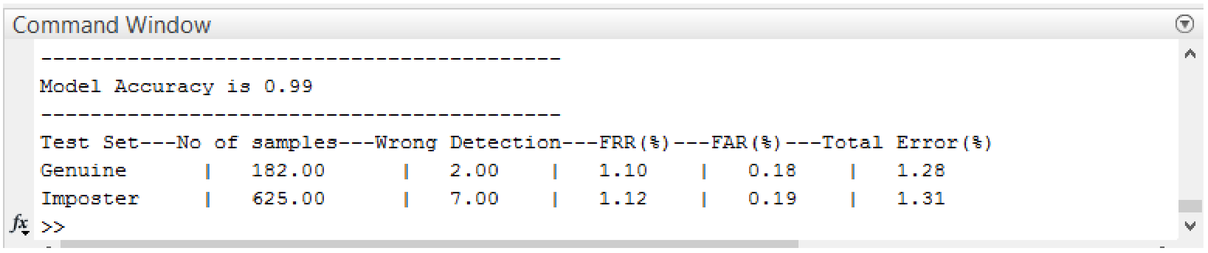


Fig 17- ROC curve for session 3

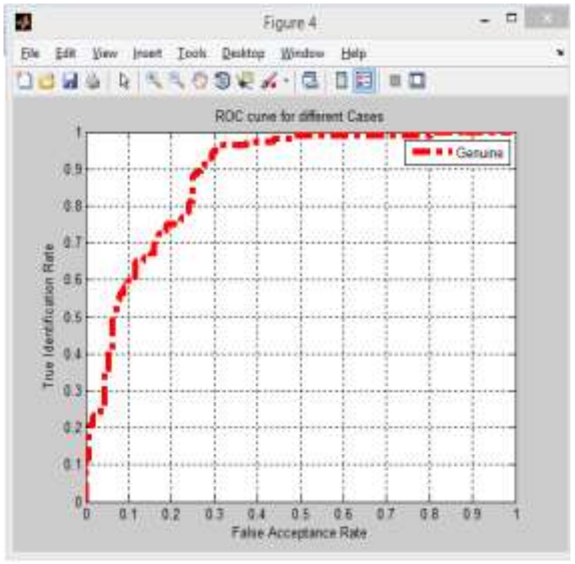
Table 4 : Proposed Approach Results

Test set	Total no. of images	Wrong Detection	FRR (%)	FAR (%)	Total Error (%)
Genuine	182	2	1.10	0.18	1.28
Imposter	625	9	1.44	0.24	1.68



```
Command Window
  -----------------------------------------
  Model Accuracy is 0.99
  -----------------------------------------
  Test Set---No of samples---Wrong Detection---FRR(%)---FAR(%)---Total Error(%)
  Genuine      |   182.00      |   2.00     |   1.10     |   0.18     |   1.28
  Imposter     |   625.00      |   7.00     |   1.12     |   0.19     |   1.31
fx >>
```

Generate Individual ROC curves for case studies



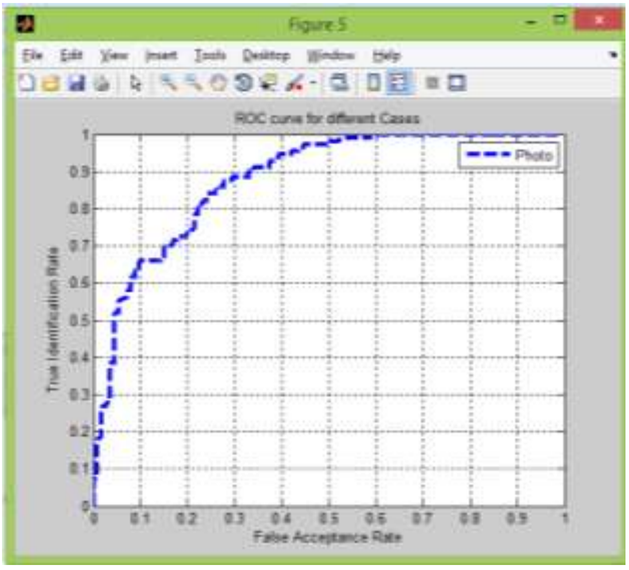
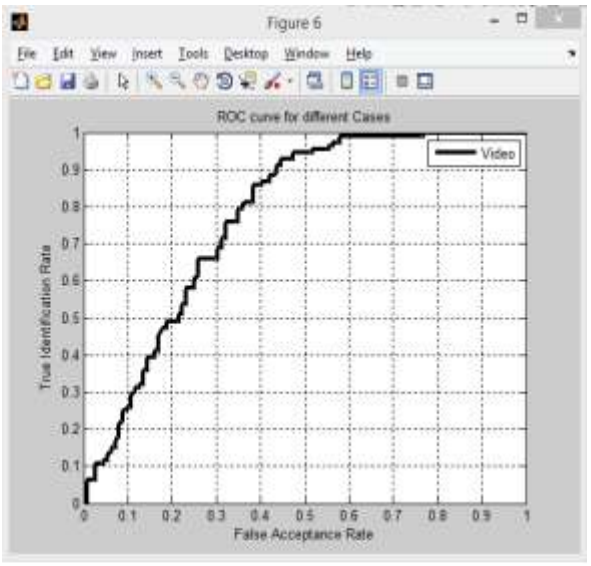


Fig 17- ROC curve for Genuine case **Fig 18-** ROC curve for Photo Attack case



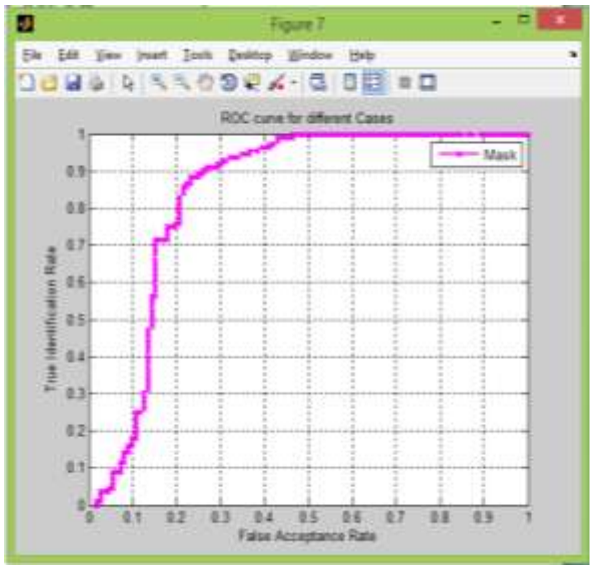


Fig 19- ROC curve for Video reply attack **Fig 20-** ROC curve for Mask

IV. Discussion

Performance metrics

The relationship between the input and output variables of a system understand by employing the suitable performance metrics like sensitivity and specificity. The general formula for calculating the sensitivity and specificity of face detection rate is given in the equation.

$$Sensitivity = \frac{Number\ of\ TP}{Number\ of\ TP + Number\ of\ FN} \times 100$$

$$Specificity = \frac{Number\ of\ TN}{Number\ of\ TN + Number\ of\ FP} \times 100$$

Where, *TP* is represented as true positive, *FP* is denoted as false negative, *TN* is represented as true negative and *FN*
is stated as false negative. In addition, the accuracy, precision and recall are the suitable evaluation metrics for finding the effectiveness of face detection rate. Precision and recall are the measure of statistical variability and a description of random errors. The general formula of accuracy, precision and recall for determining face detection rate is given in the equation

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100$$

$$Precision = \frac{TP}{TP + FP} \times 100$$

$$Recall = \frac{TP}{TP + FN} \times 100$$

F-score is the measure of accuracy test and it considers the both precision *P* and recall *R* of the test in order to calculate the score. The general formula for F-score measure is given in the equation.

$$F - score = \frac{2TP}{2TP + FP + FN}$$

V. Conclusion

We proposed a face anti-spoofing scheme based on color SURF (CSURF) features and Fisher Vector encoding. We extracted the SURF features from two different color spaces (HSV and YCbCr). Then, we applied PCA and Fisher Vector encoding on the concatenated features. The proposed approach based on fusing the features extracted from the HSV and YCbCr was able to perform very well on three most challenging face spoofing datasets, outperforming state of the art results. More importantly, our proposed approach yielded in very interesting generalization performance in the inter database experiments even when only limited training data was used. As a future work, we plan to investigate other strategies or creating more robust feature spaces for spoofing detection, including person-specific adaptation of anti-spoofing models.

References

[1]. Y. Li, K. Xu, Q. Yan, Y. Li, and R. H. Deng, "Understanding OSN-based facial disclosure against face authentication systems," in *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, ser. ASIA CCS '14. ACM, 2014, pp. 413–424.

[2]. A. Anjos, J. Komulainen, S. Marcel, A. Hadid, and M. Pietik¨ainen, "Face anti-spoofing: visual approach," in *Handbook of biometric anti-spoofing*, S. Marcel, M. S. Nixon, and S. Z. Li, Eds. Springer, 2014, ch. 4, pp. 65–82.

[3]. J. Galbally, S. Marcel, and J. Fi´errez, "Biometric antispoofing methods: A survey in face recognition," *IEEE Access*, vol. 2, pp. 1530–1552, 2014.

[4]. A. Anjos and S. Marcel, "Counter-measures to photo attacks in face recognition: a public database and a baseline," in *Proceedings of IAPR IEEE International Joint Conference on Biometrics (IJCB)*, 2011.

[5]. T. de Freitas Pereira, J. Komulainen, A. Anjos, J. M. De Martino, A. Hadid, M. Pietik¨ainen, and S. Marcel, "Face liveness detection using dynamic texture," *EURASIP Journal on Image and Video Processing*, 2013.

[6]. S. Bharadwaj, T. I. Dhamecha, M. Vatsa, and S. Richa, "Computationally efficient face spoofing detection with motion magnification," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Biometrics*, 2013.

[7]. S. Tirunagari, N. Poh, D. Windridge, A. Iorliam, N. Suki, and A. T. S. Ho, "Detection of face spoofing using visual dynamics," *IEEE Transactionson Information Forensics and Security*, vol. 10, no. 4, pp. 762–777, 2015.

[8]. J. Komulainen, A. Hadid, and M. Pietik¨ainen, "Context based face antispoofing, in *Proc. International Conference on Biometrics: Theory, Applications and Systems (BTAS 2013)*, 2013.

[9]. J. M¨a¨att¨a, A. Hadid, and M. Pietik¨ainen, "Face spoofing detection from single images using micro-texture analysis," in *Proceedings of International Joint Conference on Biometrics (IJCB)*, 2011.

[10]. J. Yang, Z. Lei, S. Liao, and S. Z. Li, "Face liveness detection with component dependent descriptor," in *IAPR International Conference on Biometrics, ICB*, June 2013.

[11]. J. Galbally and S. Marcel, "Face anti-spoofing based on general image quality assessment," in *Proc. IAPR/IEEE Int. Conf. on Pattern Recognition ICPR*, 2014, pp. 1173–1178.

[12]. D. Wen, H. Han, and A. Jain, "Face spoof detection with image distortion analysis," *Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 746–761, 2015.

[13]. T. de Freitas Pereira, A. Anjos, J. De Martino, and S. Marcel, "Can face anti-spoofing countermeasures work in a real world scenario?" in *International Conference on Biometrics (ICB)*, June 2013, pp. 1–8.

[14]. J. Yang, Z. Lei, and S. Z. Li, "Learn convolutional neural network for face anti-spoofing," *CoRR*, vol. abs/1408.5601, 2014. [Online]. Available: http://arxiv.org/abs/1408.5601

[15]. Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face anti-spoofing based on color texture analysis," in *IEEE International Conference on Image Processing (ICIP2015)*, 2015.

[16]. Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, "A face antispoofing database with diverse attacks," in *5th IAPR International Conference on Biometrics (ICB)*, 2012, pp. 26–31.

[17]. I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *International Conference of the Biometrics Special Interest Group (BIOSIG)*, Sept 2012, pp. 1–7.

[18]. T. Ojala, M. Pietik¨ainen, and T. M¨aenpaa, "Multiresolution grayscale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 7, pp. 971–987, Jul 2002.

[19]. F. Perronnin, J. S´anchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 143–156.

[20]. H. Bay, T. Tuytelaars, and L. Gool, *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, ch. SURF: Speeded Up Robust Features, pp. 404–417. [Online]. Available: http://dx.doi.org/10.1007/11744023 32

[21]. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[22]. Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 1818–1830, 2016.

[23]. I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[24]. K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher Vector Faces in the Wild," in *British Machine Vision Conference*, 2013.

[25]. J. S´anchez and F. Perronnin, "High-dimensional signature compression for large-scale image classification," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1665– 1672.