INTERNATIONAL JOURNAL OF ENGINEERING AND MANAGEMENT SCIENCES

www.scienceandnature.org

A COMPARATIVE EXPERIMENTAL STUDY OF IMAGE COMPRESSION ALGORITHM BASED ON VECTOR QUANTIZATION

Kumari Mithlesh
Department of Computer Science & Engineering, USIT Guru Gobind Singh Indraprastha University, New Delhi, India

ABSTRACT
Image compression is a main field in the development of numerious multi-media computer application and telecommunication services such as video conferencing, and many others areas like interactive education, medical, space research etc . Image compression techniques main aim at removing (or minimizing) redundancy in data, but maintains acceptable image reconstruction.Vector quantization is an effective technique for data compression and has been efficiently used in numerious different fields. The via-media usually used to develop a codebook are Linde, Buzo, Gray (LBG) algorithm, fuzzy vector Quantization (FVQ) algorithm. Although if the divisible boundaries in the codebook development are non linear their output can debase fast. In this research I present three alforthm LBG, FCM, and KFL algorithm. KFL takes mileages of the distance kernel trick and the gradient-based fuzzy grouping method to generate a codebook automatically. Experiment with real life data authenticates that the suggestive KFL algorithm is more well suited quality in its resultant. Vis- a-vis to that of the other LBG and FVQ.

KEYWORDS: Image Compression, Radial Basis Function, Neural Network, Codebook design, Vector quantization, Kernel fuzzy Learning, Gradient based.

INTRODUCTION

With the fast paced emerging technology in computer related spheres, it is essential to explore and develop well suited quality data compression methods [4,9,16, 17]. One of basic targets of data compression is to lower the bit rate for data set, as well as balancing the data quality. According to the VQ method which has been currently applied in this segment, it targets on mapping input vectors by smaller size of codebook vectors. Several latest developments in kernelised clustering that perform FCM [20] in a higher dimensional and possibly infinitely dimensional kernel space by use of the 'kernel trick' [21–22]. Kernels implicitly map patterns from the input space to a higher dimensional space with the hope of simplifying the geometry of the clustering space and with help of kernel I try to improve the performance of grouping and classification methods [1, 2].

I provide the nutshell of the introduction idea of VQ method. In this methodology we see a vector quantizer produces a result as index of weight vector when unidentified input vector is received. Arithmetically, the working of a vector quantizer consists a group of training vector $\mathbf{X} = [x_1, x_2,.....,x_{n.....}x_N]$; a distortion measure , a centroid calculative method and a categorization approach. Depending upon these factors VQ can be summarize by mapping a m- dimensional vector into a limited set of vector $\mathbf{V} = [v_1, v_{2,}..,v_{m,}.....,v_C]$ where n>>c .Each vector v_m is noted as codeword. Set of all codeword together called codebook. In co-ordination with each codeword there is a vicinity neighbor area known as encoding region ω_m , in the encoding territory Ω. Normally the VQ design can be concluded as follows: In a scenario of training set of input vectors **X** , a codebook size C, calculate a group of centroid and the encoding area Ω , in a fashion that the average distortion d is given in eq. (1) should be minimized:

$$d[x_n, C(x_n)] = \frac{1}{NM}\sum_{n=1}^{N} d[x_n, C(x_n)] \qquad (1)$$

Where $C(x_n) = v_m$, if $x_n \in \omega_m$

The codebook can be formulated by above method. Segmentation of an arbitrary vector **x** can be obtained by a thorough search through the codebook. Without loss of normalcy the vector **x** is deputed to the class b*, which is sampled design by the codebook $\mathbf{v}_{m..}$

$$b^* = \arg\min_{1 \le b \le C} d(X, Vb) \qquad (2)$$

The average distortion measure is

$$d[x_n, C(x_n)] = \frac{1}{C}\sum_{C=1}^{C} d[x_n, C(x_n)] \qquad (3)$$

The codebook structure is the basis of VQ, different learning algorithm. For codebook generation for instance Linde,

Buzo, Gray(LBG) algorithm [3], Fuzzy c-means algorithm (FCM) [6,7] had already been developed. But in these algorithms if the dividing boundaries are not linear then we will not obtain the desired optimum performance. In this scientific study, I Compare LBG, FCM and a kernel fuzzy learning (KFL) algorithm which enables to get the max^{m} benefit of distance kernel trick and gradient based for developing the codebook self-regulating. The suggestive algorithm tallied with LBG, FCM by numerous true instances to elaborate its efficacies.

The balance of the study is compiled as follows: The different algorithm LBG, FCM and KFL for codebook creation is initiated in sec. 2. Various real life examples are applied to check the efficacies of the KFL in sec. 3. The final portion throws the light on the summary aspect.

METHODOLOGY

1. LBG Design Algorithm

In a case of training data **X** a codebook of size C*, and a noted small number Ɛ > 0. The LBG algorithm can be put into action by the following procedure.[1, 23]

1. Let N =1 and

$$v_c = \frac{1}{N}\sum_{1}^{N} x_n \tag{4}$$

Calculate

$$D^{*}_{avg} = \tfrac{1}{NM}\sum_{n=1}^{N} d[x_n - v_m] \tag{5}$$

2. **Dividing:** For i=1,2,.....C, set

$$\mathbf{v}_i^{(0)} = \mathbf{v}_i(1+\varepsilon) \tag{6}$$

$$\mathbf{v}_{N+i}^{(0)} = \mathbf{v}_i(1-\varepsilon) \tag{7}$$

Set C = 2C.

3. **Repetition:** Let $D^{(0)}_{avg} = D^{*}_{avg}$ .Set the iteration index i=0.
   a. For n=1,2,….N, find the minimum value of

$$\left\| x_n - c_k^{(i)} \right\|^2 \tag{8}$$

over all k= 1, 2,… C. Let k* be the index which achieves the minimum. Set

$$Q(x_n) = c_{k^*}^{(i)} \tag{9}$$

   b. For k= 1, 2,… C, update the codevector

$$c_k^{(i+1)} = \frac{\sum_{Q(x_n)=c_{k^*}^{(i)}} x_n}{\sum_{Q(x_n)=c_{k^*}^{(i)}} 1} \tag{10}$$

   c. Set i = i+1.
   d. Calculate

$$D^{i}_{avg} = \tfrac{1}{NM}\sum_{n=1}^{N} d[x_n - Q(x_n)] \tag{11}$$

   e. If $\left(D^{(i-1)}_{avg} - D^{(i)}_{avg}\right) \Big/ D^{(i-1)}_{avg} > \varepsilon$, go back to Step (i).
   f. Set. $D^{*}_{avg} = D^{(i)}_{avg}$ .For , k=1, 2,….C set

$$c_n^{*} = c_n^{(i)} \tag{12}$$

as the final codevectors.

4. Repeat Steps 3 and 4 until the desired number of codevectors is obtained.

2. FCM Design Algorithm

The purpose of the algorithm is to segmenting the training set **X** into C fuzzy set by reducing the following targeted function.

$$J\tau(U,V) = \sum_{m=1}^{C}\sum_{n=1}^{N} (\mu_{mn})^{\tau} (d_{in})^2 \tag{13}$$

$$d_{in} = d(x_n - v_i) = \left[\sum_{j=1}^{M} (x_{nj} - v_{ij})^2\right]^{1/2} \tag{14}$$

μ_{mn} is the membership of $\mathbf{X}_n$ in class m, and τ is the no. Of controlling grouping fuzziness. The matrix U with mn-th entry u_{mn} having limitation to contain element value between [0, 1] such that $\sum_{m=1}^{C} \mu_{mn} = 1$. The function J_t can be decrease using the well known iterative algorithm[12].

An effective algorithm for fuzzy classification ,called iterative optimization. The steps in this algorithm are:

1. Fix C (2 < C < N) and select a value for parameter ɼ. Initialize the partition matrix, U^{0}. Each step in this algorithm will be labeled r, where r = 0, 1, …….

2. Calculate the c center $\{v^r_m\}$ for each step.

$$v_m = \frac{\sum_{n=1}^{N} \mu_{mn}^{\tau} x_n}{\sum_{n=1}^{N} \mu_{mn}^{\tau}} \quad (15)$$

3. Update the partition matrix for the *r*th step, U^r

$$\mu_{mn}^{(r+1)} = \left[\sum_{j=1}^{C} \left(\frac{d_{in}^{(r)}}{d_{jk}^{(r)}} \right)^{2/(\tau-1)} \right]^{-1} \quad (16)$$

4. If $\| U^{r+1} - U^r \| < \varepsilon$,stop ; otherwise set r = r + 1 and return to step 2.

3. THE KFL Algorithm

Inspite of LBG and FCM which have been proven to be quite applicable in some selective application , they are unusable with a condition when divisible boundaries are nonlinear. I club the distance kernel trick with gradient based fuzzy clustering method [6,7,11,14,18] to self regulatory generate the codebook.

Distance kernel trick: Let us recount the mercer kernel theory[15], in that each kernel function can be indicated as

$$K(\mathbf{x}, \mathbf{v}) = \Phi(\mathbf{x})^{\mathrm{T}} \Phi(\mathbf{v}) = \Phi(\mathbf{v})^{\mathrm{T}} \Phi(\mathbf{x}) \quad (17)$$

where **x** and **v** are emptyless vectors, where Φ results in mapping from the input space to a high dimensional feature space, and the kernel function K is elaborated as the inner product in the prescribed feature space and essential thing about kernel function is that the Euclidean distance is calculative unknowing elaborately.

$$\begin{aligned} \| \Phi(\mathbf{x}) - \Phi(\mathbf{v}) \|^2 &= \left(\phi(x) - \phi(v)\right)^T \left(\phi(x) - \phi(v)\right) \\ &= \Phi(\mathbf{x})^T \Phi(\mathbf{x}) - \Phi(\mathbf{v})^T \Phi(\mathbf{x}) - \Phi(\mathbf{x})^T \Phi(\mathbf{v}) + \Phi(\mathbf{v})^T \Phi(\mathbf{v}) \\ &= K(\mathbf{x}, \mathbf{x}) + K(\mathbf{v}, \mathbf{v}) - \mathbf{2}K(\mathbf{x}, \mathbf{v}) \end{aligned} \quad (18)$$

This is termed as distance kernel trick. The Gaussian radial basis function (RBF) kernel function

$$\mathrm{K(X, V)=exp}\left(\frac{-\| x - v \|^2}{\sigma^2} \right) \quad (19)$$

where σ is an adjustable parameter.

we recall (13), which is used for learning M codewords from a dataset **X**.

U = $[u_{ik}]$ denotes a membership matrix with satisfying value

$$U \in \left(\mu_{mn} \in [0,1] \,|\, \sum_{m=1}^{C} \mu_{mn} = 1, \forall n; 0 < \sum_{n=1}^{N} \mu_{mn} < N, \forall m \right) \quad (20)$$

The factor τ is a weighing component of each fuzzy membership and establishes the quantity of fuzziness of the resulting categorization. In case to deal with a more common dataset (13) is written again

$$J_{\tau}^{\phi}(U,V) = \sum_{m=1}^{C} \sum_{n=1}^{N} (\mu_{mn})^{\tau} \left\| \phi(x_n) - \phi(v_m) \right\|^2 \quad (21)$$

Eq (21) is the objection function of the KFL algorithm. It is to be noticed that $\varphi(\mathbf{v}_i)$ indicates a mapped point of $\mathbf{v}_m$ in the true original input space. After solving eq. (18) and (21)

$$J_{\tau}^{\phi}(U,V) = 2\sum_{m=1}^{C} \sum_{n=1}^{N} (\mu_{mn})^{\tau} \left(1 - K(x_n, v_m)\right) \quad (22)$$

Here I take amount of fuzziness τ =2. In case to shorten the objective function $J\phi(U,V)$, I implement the steepest gradient descent algorithm. In nutshell the learning rate can be as defined follows:

$$\Delta \mathbf{v}_m = \eta(\mathbf{v}_m - \mathbf{x}_n) = \eta \frac{\partial J_{\tau}^{\phi}}{\partial v_m} \quad (23)$$

Pertaining to the Gaussian kernel function, the objective function can be jotted down as

$$J_{\tau}^{\phi} = 2\sum_{m=1}^{C} \left(\mu_{mn}\right)^2 \left(1 - \exp\left(\frac{-\| x_n - v_m \|^2}{\sigma^2} \right) \right) \quad (24)$$

Put (24) into (23),I get

$$\nabla \mathbf{v}_m = 4\eta \left(\mu_{mn}\right)^2 \sigma^{-2} K\left(\mathbf{x}_n, \mathbf{v}_m\right)\left(\mathbf{x}_n - \mathbf{v}_m\right) \quad (25)$$

$$\mathbf{v}_m^{\,t+1} = \mathbf{v}_m^{\,t} - \nabla \mathbf{v}_m \quad (26)$$

Where t indicates the different interval time index.

The constrained optimization in (20) can be sorted out by using the Langrange multiplier

$$P_q = \sum_{m=1}^{C} \left(\mu_{mn}\right)^2 \left(1 - \exp\left(\frac{-\| x_n - v_m \|^2}{\sigma^2} \right) \right) - \lambda \left(\sum_{m=1}^{C} (\mu_{mn})^2 - 1 \right) \quad (27)$$

Considering the initial derivative of P_q w.r.t. to μ_{mn} and put the result to zero,

$$\mu_{mn} = \frac{\lambda}{4(1-K(x_n, v_m))} \quad (28)$$

In reference to (20), I find that

$$\lambda = -\frac{1}{\sum_{j=1}^{C} \frac{1}{4(1-k(x_n, v_j)}} \quad (29)$$

Replacing (29) into (28), the membership are explained as

$$\mu_{mn} = \frac{1}{\sum_{j=1}^{C} \left(\frac{1-k(x_n, v_m)}{1-k(x_n, v_j)} \right)} \quad (30)$$

After this eq. [1] I give crux of KFL

1. Select $\tau = 2$ and select no. Of cluster C, select $\max^{im}$ iteration steps t_{ax} and take a small value for iteration termination error $\varepsilon > 0$.
2. Set the initial value of the membership matrix μ^0_{mn}.
3. For $t=1, 2 \ldots\ldots\ldots\ldots t_{ax}$ put into action:
   (a) Perform updating process for all codeword's $\mathbf{v}_e^t$ according to (26)
   (b) Compute new membership value μ^t_m from eq. (30)
   (c) Calculate iteration error for stopping criteria ET $=mad_{men}| \mu^t_m - \mu^{t-1}_{mn}|$, and if $E^T < \varepsilon$, *stop* updating process.
4. At the end, I got the perfect codebook **v.**

RESULTS

To show the effacies of the KFL, I do numerous experiments and respective results are shown in table1. These algorithms are coded in MATLAB language. For the time being MATLAB language is selected for it's comfortably rather than speedy rate of response.

Fig. 1 Training image of size 256 x256

Maintaining the generality different images such as 'Baboon' , 'Lina' , 'Pepper' , 'Airport' etc. With resolution 256 x 256 pixels. 'Lina' image is used to initiate the codebook of various sizes with measurements P = 16(4096 blocks of size 4 x 4). Other images are applied to evaluate the effacies of final results. The resulting images are tested by the peak signal to noise ratio (PSNR), defined as

$$PSNR = 10 \log_{10} \frac{255^2}{\left(1/NM \sum_{n=1}^{N} || x_n - v_m ||^2 \right)} \quad (31)$$

The kernel function leads to dissimilar results . There is no general theory to navigate the condition of Kernel-based algorithm.This is an issue which can be discussed. Different cross validation process are applied on different dataset. This is done in two phases, first taking a long interval to locate a good initial assumption of the parameters, and then minimizing it slowly the interval to again find the factors in second phase.

The results of experiments for training image are depicted in Table .

TABLE I

PSNR for training image

Code Book	LBG	FCM	KFL
	PSNR(db)	PSNR(db)	PSNR(db)
64	25.3581	22.9520	26.8658
128	26.8754	23.0068	26.8754
256	29.0716	23.5058	29.0716
512	31.7355	23.7382	31.7355

TABLE II

MSE for training image

Code Book	LBG	FCM	KFL
	MSE(db)	MSE(db)	MSE(db)
64	189.3538	329.5211	189.3538
128	133.5178	325.3904	133.5178
256	80.5230	290.0708	80.5230
512	43.6042	274.9541	43.6042

Fig. 2 Test Image of size 256 x 256

The results of experiments for test image are depicted in Table

TABLE III

PSNR VALUE FOR TEST IMAGE

Code Book	LBG	FCM	KFL
	PSNR(db)	PSNR(db)	PSNR(db)
64	26.8658	25.2987	26.8658
128	28.7374	25.3749	28.7374
256	30.7053	25.5137	30.7053
512	33.2163	26.0810	33.2163

TABLE IV

MSE VALUE FOR TEST TMAGE

Code Book	LBG	FCM	KFL
	MSE(db)	MSE(db)	MSE(db)
64	133.8134	191.9608	133.8134
128	86.9651	188.6227	86.9651
256	55.2773	182.6884	55.2773
512	31.0065	160.3181	31.0065

Fig. 3 Comparison of image reconstructed from different algorithm LBG, FCM, KFL recpectively for Codebook size = 512 and original image size = 256x 256

Fig.5 Histogram for original image of size = 256x 256

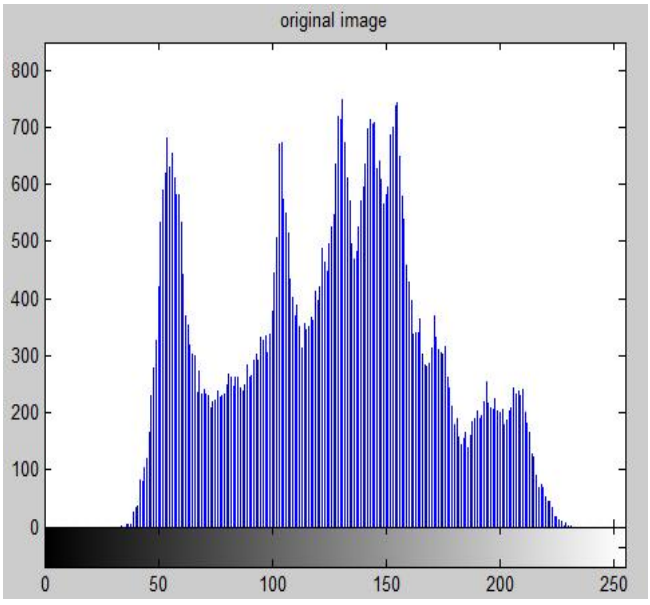
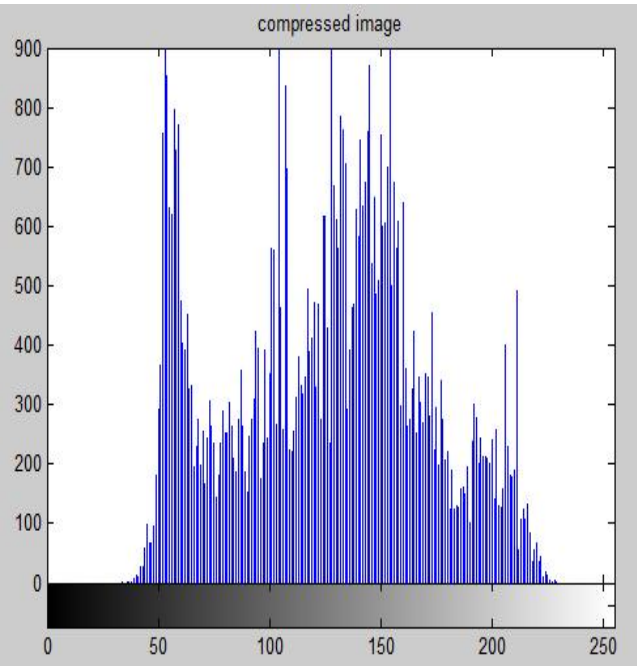


Fig.5 Histogram for compressed image from KFL algorithm for codebook size =512 and original image size = 256x 256



CONCLUSION

For the solutions of VQ codebook designing issue, I suggest a KFL algorithm, where the following two methods namely distance kernel trick and the gradient-based fuzzy clustering algorithm are very well interconnected. By virtue of experimental results obtained it indicates that PSNR value along with expansion of codebook size is far superior comparative to LBG, FCM. This impressive result is achievable due to the KFL method's superiority, in which the image reconstructed from codebook is accepted by human eye. Therefore giving impression that no alteration has been taken place in the said image. Thus we are having an advantage of access space due to the compression which took place in the above method. When I analyze LBG, FCM I find our KFL algorithm is definitely is the most appropriate and result oriented modus-operandy for image compression.

ACKNOWLEDGMENT

The author would like to thanks my friends for their valuable discussions and help us to improve presentation.

REFERENCES

[1] Zongbo Xie. Jiuchao Feng, "Codebook design for vector quantization based on a kernel fuzzy learning algorithm", Circuits Syst Signal Process (2011) 30:999–1010

[2] Y.K. Chan, H.F. Wang, C.F. Lee, "A refined VQ-based image compression method". Fundam. Inform. **61**, 213–221 (2004)

[3] C.H. Chang, P.F. Xu, R. Xiao, T. Srikanthan, "New adaptive color quantization method based on selforganizing maps". IEEE Trans. Neural Netw. **16**, 237–249 (2005)

[4]. C.C. Chang, W.L. Tai, C.C. Lin, "A reversible data hiding scheme based on side match vector quantization". IEEE Trans. Circuits Syst. Video Technol. **16**, 1301–1308 (2006)

[5] R.N. Dave, R. Krishnapuram, "Robust clustering methods: a unified view". Neural Comput. **5**, 270–293 (1997)

[6] D. Graves, W. Pedrycz, "Performance of kernel-based fuzzy clustering". Electron. Lett. **43**, 25–26 (2007)

[7] D. Graves, W. Pedrycz, "Fuzzy c-means, Gustafson-Kessel FCM, and kernel-based FCM: a comparative study". Adv. Soft Comput. **41**, 140–149 (2007)

[8] H.C. Huang, J.S. Pan, Z.M. Lu, S.H. Sun, H.M. Hang," Vector quantization based on genetic simulated annealing". Signal Process. **81**, 1513–1523 (2001)

[9] H.B. Kekre, T.K. Sarode, "Vector quantized codebook optimization using K-means". Int. J. Comput. Inf. Sci. Eng. **1**, 283–290 (2009)

[10] H.B. Kekre, T.K. Sarode, "Fast codebook search algorithm for vector quantization using sorting technique", in *Proceedings of the International Conference on Advances in Computing, Communication and Control*, vol. 1 (2009), pp. 317–325.

[11] D. Kim, K. Lee, D. Lee, K.H. Lee, "Evaluation of the performance of clustering algorithms in kernel induced featureSpace". Pattern Recognition. **38**, 607–611 (2005)

[12] J.F. Kolen, T. Hutcheson, "Reducing the time complexity of the fuzzy C-Means algorithm". IEEE Trans. Fuzzy Syst. **10**, 263–267 (2002)

[13] J. Leski, "Towards a robust fuzzy clustering. Fuzzy Sets" Syst. **137**, 215–233 (2003)

[14] K. Mizutani, S. Miyamoto, "Possibilistic approach to kernel-based fuzzy c-means clustering with entropy regularization", in *Lecture Notes in Computer Science: MDAI*, vol. 3558 (2005), pp. 144–155

[15] K.R. Muller, S. Mika, G. Ratsch, K. Tsuda, B. Scholkopf, "An introduction to kernel-based learning algorithms". IEEE Trans. Neural Netw. **12**, 181–202 (2001)

[16] J.S. Pan, F.R. McInnes, M.A. Jack, "Fast clustering algorithms for vector quantization". Pattern Recognit. **29**, 511–518 (1996)

[17] J.S. Pan, F.R. McInnes, M.A. Jack, "VQ codebook design using genetic algorithms". IEE Electron. Lett. **32**, 194 (1996)

[18] D. Park, C.N. Tran, S. Park, "Gradient based fuzzy c-means algorithm with a mercer kernel", in *Lecture Notes in Computer Science: ISNN*, vol. 3971 (2006), pp. 1038–1043

[19] D. Jacques Vaisey and Allen Gersho, "Variable block image coding", University of California Santa Barkma, **CA** 93106

[20] Bezdek, J.C.: "Pattern recognition with fuzzy objective function algorithms" (Plenum, New York, 1981)

[21] Chiang, J.H., and Hao, P.Y.: "A new kernel-based fuzzy clustering approach: support vector clustering with cell growing", IEEE Trans.Fuzzy Syst., 2003, 11, (4), pp. 518–527

[22] Zhou, S., and Gan, J.: "Mercer kernel fuzzy c-means algorithm and prototypes of clusters". Proc. Conf. Int. Data Engineering and Automated Learning, Exeter, UK, 2004, Vol. 3177, pp. 613–618

[23] Y. Linde, A. Buzo, and R. M. Gray, ``An Algorithm for Vector Quantizer Design," *IEEE Transactions on Comm*