International Journal of Applied Engineering Research, ISSN 0973-4562 Vol. 10 No.63 (2015)

International Conference on Computational Systems in Engineering and Technology, March 7 & 8 2014, Chennai

A Comparative Study on different sparse algorithms and its application on Image compression

Rashmita Routray
Department of Telecommunication
SRM University
Tamilnadu, India
rashmitaroutray@gmail.com

Dr Uma Ranjan Jena
Electronics and Telecommunication dept,
Veer Surendra Sai University of Technology
Odisha,India
urjena@rediffmail.com

Abstract*—we presented a comparison of different sparse algorithms and its application in image compression. In part I we have compared with the existing sparse algorithm such as Matching Pursuit, Orthogonal Matching Pursuit, Order Recursive Matching Pursuit and Matching Pursuit with Simulated Annealing. In this part we have discussed the dictionary used for sparse algorithm and its properties. The result of compressed image using the above technologies are compared. In part II we have proposed two new algorithms, Those are Matching Pursuit using Genetic Algorithm and Matching Pursuit with wavelet decomposition. A comparison is done between the existing algorithms and proposed algorithms with respect to their mse, PSNR and time consumption.

***Keywords*—matching pursuit, omp, ormp ,GA ,Gabor transform**

I. INTRODUCTION (*Image compression*)

Image compression has been a widely tackled field for a long time. Several methods are available, and standards exist that provide high compression ratios for a given subjective quality. Although storage media is becoming cheaper, the requirement for compressed images still holds for communication through limited capacity networks, or for storage on pervasive computing devices with limited storage such as hand held devices. Several applications also suffer from large image data sets. Properties such as high resolution and high pixel depth lead to large images. Examples of such applications are space images and medical images.Take for example, a high resolution x-ray image. Most of the image is a black background, with large areas of white bone. It makes sense to try to encode most of the background as one entity rather than encode every pixel in the image. The main objective of any data compression algorithm is to exploit any correlation or similarities within the data to be represented once rather than for every occurrence. Image compression also utilizes the fact that the human visual system is less sensitive to high frequency changes (e.g. edges) than low frequency changes.

Image compression using sparse algorithms:

Representing a given signal as a linear sum of a few signals from a signal set is commonly known as sparse approximation in the signal processing literature. Here we have represented any signal using dictionary based approximation. The signals which represent the columns of the matrix A are commonly called atoms and the set of atoms is called a dictionary.

A. Dictionary Based Representation

Any signal b can be represented as

$$b = \sum_k a_k \quad \ldots\ldots\ldots\ldots\ldots\ldots (1.1)$$

Or

in matrix form $b = Ax$ ……(1.2)

$A \in R$, R^{m*n} where R is called a dictionary. a_k is called the atoms of the dictionary. The signal b is represented as the weighted sum of basis vectors which are called atoms. In general, we require to obtain the coefficient vector x that minimizes the error. For the rest of the discussion we will try to minimize the least square error.

$$\text{Min } \|b - Ax\|_2 \leq \delta \qquad (1.3)$$

or

$b = Ax + R$, where R is the residual. (1.4)

In this case x may be obtained using the pseudo-inverse of A .

$$Ax = b \qquad (1.5)$$

$$x = A^{-1} b \qquad (1.6)$$

$$x = (A^T A)^{-1} A^T b \qquad (1.7)$$

After that, the signal b may be reconstructed using equation (1.2) . If m = n (and assuming that A is full rank), then we get a perfect reconstruction of b. If n > m, then we get a sparse vector x. If n < m then we will not get perfect reconstruction for the signal b.

If the dictionary elements cover the signal space, and have similar properties as the signal, then we may indeed require

less basis elements than signal sample to represent the signal with acceptable distortion.

B. Properties of the dictionary:

Several properties should be considered when selecting the basis function for
the dictionary. The dictionary properties may be:

1. Full space coverage
2. Time-Frequency localization
3. Orthogonality
4. Orthonormal
5. Scale invariance
6. Shift/ rotation invariance

II. SPARSE ALORITHMS

Introduction on sparse alorithms

The area of sparse approximations has seen tremendous research progress in the recent years. At the core of many of these activities are results that ensure the efficient recovery of sparse solutions, i.e., solutions having few nonzeros , to linear systems Ax = b with A ∈ R , Rm*n , b ∈ R , Rm . The basic idea is to let 'A' be a composition of several bases of m and hence n > m . This implies that b has several representations, often guaranteeing sparse solutions to the equation system: Under appropriate assumptions on A and if b is known to have a sufficiently sparse representation.

A generalization to the sparse representation problem is to allow Ax deviate from b with respect to some ℓp-norm, i.e., to look for the sparsest solution x satisfying $\|Ax - b\|_p \leq \delta$ for given $\delta > 0$, where $p \in [1,\infty]$. The literature investigates conditions on A and b such that a solution minimizing the ℓp-norm under such constraints is near a sparsest solution.

For example, in the case of compression, the vector b represents a signal to be compressed and the columns of A represent a redundant set of basis signals. The solution to the above problem yields the fewest coefficients to represent the signal under consideration thus leading to maximal compression efficiency. Representing a given signal as a linear sum of a few signals from a signal set is commonly known as sparse approximation in the signal processing literature. The signals which represent the columns of the matrix A are commonly called atoms and the set of atoms is called a dictionary. Since the signal set is larger than necessary (i.e., N >K) and the signals span Rk, the signals in the dictionary are said to form a redundant basis. In this paper we compare so many greedy algorithms those are developed till now and its progress in image processing field. Our main aim is to find out the differences and the advantage of one upon another

A. Matching Pursuit

Matching pursuit [1] is an iterative greedy algorithm that searches for the sparse representation of a signal through a sequence of mono-atomic approximations. Each iteration of the algorithm consists of two steps:

(1) an atom selection step
(2) a residual update step.

The atom selection step finds the atom which has the highest correlation with the current residual error, where the correlation is measured as the length of the orthogonal projection. The update step updates the residual error by subtracting the correlated part from it.

Let ai, $1 < i < N$, denote the ith column of the dictionary matrix A. Assume that all atoms are normalized with unity magnitude, i.e., $\|ai\|2 = 1$; for all i.

At the jth iteration, the algorithm finds

$\alpha j = \arg\max | < rj\text{-}1, ai > |$ [MP: Atom selection]

where < : > denotes the inner-product operation, and rj-1 denotes the residual at the

(j-1)th iteration with r0 =b. The inner product represents the coefficient associated with the selected atom αj. Let us denote it as cj . The algorithm then updates the residual as

$rj = rj\text{-}1 - cj\ \alpha j$ [MP: Residual update]

The approximation at the jth iteration is given as : $bj = \sum ck\ \alpha k$ (for k=1 to j)

The algorithm terminates when a halting criterion is satisfied, such as when the norm of the residual falls below a desired approximation error bound, or when the number of distinct atoms in the approximation equals a desired limit.

The Dictionary used here consists of gabor basis functions

$$g(x,y) = K \exp(-\Pi(a2(x-x0)r2 + b2(y-y0)r2))* \exp(j(2\Pi(u0x+v0y)+p))$$

Despite its simplicity, the algorithm is sub-optimal. Several examples can be given that demonstrate that the greedy nature of the algorithm is not sparsity preserving. Because of its greedy nature, the matching pursuit algorithm is not optimal. Several extensions to the algorithm have been made that enhance the performance (sparsity, reconstruction error) of the algorithm, reduce complexity, memory requirements.

An intrinsic feature of the algorithm is that when stopped after a few steps, it yields an approximation using only a few atoms. When the dictionary is orthogonal, the method works perfectly. If the object is made up of only $m << n$ atoms and the algorithm is run for m steps, it recovers the underlying sparse structure exactly. When the dictionary is not orthogonal, the situation is less clear. Because the algorithm is myopic, one expects that, in certain cases, it might choose wrongly in the first few iterations and, in such cases, end up spending most of its time correcting for any mistakes made in the first few terms. In fact this does seem to happen.

The matching pursuit algorithm is very simple. But because of the sub-optimality, it suffers from slow convergence and poor sparsity result .

B. Orthogonal Matching Pursuit :

Orthogonal Matching Pursuit is developed by Krishna Prasad, P. S., and Rezaiifar, Y. C. P. R , in 1993. The basic matching pursuit, although guarantees asymptotic convergence, it does not necessarily provide the optimal approximation with respect to the current selected subset. This optimality is only achieved if the rth residual is orthogonal to all the selected basis.

Orthogonal Matching Pursuit was developed as an improvement to Matching Pursuit .It therefore shares many of the properties of Matching Pursuit

ri = ri-1 –□i (□iT□i)-1□iTri-1

Where ϕi is the atom and ri is the current residual. Orthogonal matching pursuit is an attempt to improve the performance of the basic matching pursuit. At each step r, the algorithm solves the least square problem .

|| i ai ||

At each iteration, the coefficients are updated to ensure backward orthogonality of the current residual. In each iteration, orthogonal Matching Pursuit calculates a new signal approximation xn. The approximation error is then used in the next iteration to determine which new element is to be selected. In particular, the selection is based on the inner products between the current residual and the column vectors.

In many multi scale bases (e.g. wavelets), signals of interest (e.g. piecewise-smooth signals) not only have few significant coefficients but also those significant coefficients are well-organized in trees. In fact, these embedded tree structures for significant wavelet coefficients have been used successfully in compression, modeling, and approximation tasks. We propose to exploit the sparse tree representation as additional prior information for signal reconstruction with limited numbers of measurements.

Intuitively, a general sparse representation with K coefficients can be described with 2K numbers: K for the values of the significant coefficients and another K for the locations of these coefficients. If the K significant coefficients are known to be organized in trees then the indexing cost is significantly reduced and hence the total description of the unknown signal. Exploiting this embedded tree structure in addition to the sparse representation prior in inverse problems would potentially lead to: (1) better reconstructed signals; (2) reconstruction using fewer measurements; and (3) faster reconstruction algorithms.

C. Order Recursive Matching Pursuit :

Order Recursive Matching Persuit is developed by Natarajan in 1995. It provides a novel algorithm for the solution of the approximation problem . It is very similar to the matching pursuit algorithm .It greedily selects the basis function that best approximates the current residual signal. Initially, each column in dictionary A is normalized. The residual R0 is initially set to b. Assume the set of selected signals Γ is initially empty :

$\Gamma = \{\ \}$,

At each iteration of the algorithm, the index k is chosen

$Max_k |\langle R0, akr\rangle|$, $k=1,2,\ldots n-\Gamma$

$\Gamma = \Gamma \cup k$

where ak is the column k in dictionary A in iteration r. The residual is then projected onto the subspace orthogonal to akr

$Rr+1 = Rr - \langle Rr, akr\rangle akr$

Then, each unselected dictionary element is projected to the space orthogonal to ak then it is normalized.

$ajr+1 = ajr - \langle ak\ r, ajr\rangle ak\ r$, $j= 1,2\ldots n-\Gamma$

$ajr+1 = ajr / \| ajr\|$, $j= 1,2\ldots n-\Gamma$

The algorithm iterates until the norm of the residual ||RR|| is less than a predefined error.

D. Matching Pursuit with Simulated Annealing :

Several enhancements are available that improve the performance of greedy algorithms. Simulated annealing is one of them, following the process of physical annealing. Simulated annealing allows the greedy algorithm, depending on certain conditions, to select a solution element that is not the best, or one that worsens the overall solution. This allows the algorithm to explore different areas of the search space to reduce the chances of falling into a local minimum. It is desirable that the probability of selecting the non-best element be higher at the early stages of the algorithm , and lower when the solution starts to converge. It is also desirable that this probability is adaptive,

in the sense that the current state plays a role in calculating this probability.

The main equation for simulated annealing is given by

$\rho = e^{(-\Delta E)/T}$ .....(2.2.9.1)

where ΔE is the change in energy and T is the current annealing coefficient .

After calculating ρ, we generate a random variable p[0 , 1]. If the value of p is greater than ρ, we perform an annealing step, or allow the algorithm to make a non-optimal selection. Otherwise, the algorithm proceeds in a normal greedy fashion.

Image compression using over-complete dictionaries is usually performed using greedy algorithms. This means that they are all sub-optimal. By using the above mentioned algorithms we reduce this complexity, but may fall into a non optimal solution, or a local minimum. By using a technique based on simulated annealing, we allow the selection algorithm to explore a larger space. Following on the discussion in the previous section, we are essentially designing an algorithm that is the marriage between a greedy forward selection algorithm, a greedy backward selection algorithm, and the concept of simulated annealing. An outline of the algorithm is shown below. The forward selection step may be performed by any of the forward selection algorithms (Matching pursuit, orthogonal matching pursuit, order recursive matching pursuit, . . .). The backward selection may be performed by any of the backward selection algorithms. We are left with defining the energy change criterion ΔE and the annealing schedule T.

Algorithm requirements :

In order to obtain a proper algorithm for the solution of the subset selection problem, the encoding algorithm should posses several properties.

1 .The algorithm should terminate (converge) in a finite number of step

Algorithm 4.1 Subset selection using simulated annealing

```
Initialization
Γ ← {}, ξ ← ∞
Subset selection phase
while ξ > ϵ do
  ρ ← exp^{-E/T}
  Generate random number α[0, 1]
  if α < ρ then
    Perform backward elimination algorithm
  else
    Perform forward selection algorithm
  end if
  Calculate new E
  Update T
end while
```

2. The algorithm should provide a solution that is better than current selection algorithms
(or in the worst case the same) in terms of the number of selected basis functions
3. The algorithm should provide a solution that is better than current selection algorithms
(or in the worst case the same) in terms of the reconstructed signal quality
4. The output of the algorithm should be done in a way that minimizes the complexity of
the decoder.

All of the forward selection algorithms discussed above are candidates as the selection method upon which simulated annealing may be applied. Since the purpose of this research is to investigate the effect of combining the heuristic simulated annealing method with a selection algorithm, the basic matching pursuit is of primary interest. Also, it is advantageous due to its simplicity and because it is of the lowest complexity. The other methods are modifications to the basic matching pursuit, therefore the effect of adding simulated annealing may be less evident than matching pursuit. They are also much more complex in terms of computation and implementation. For the rest of this chapter we will use the basic matching pursuit algorithm as the forward selection algorithm, and we will use the primary concept of backward elimination algorithms for the backward step.

Inputs:

The inputs to the algorithm should be

- Signal b < Rm to be approximated.
- Dictionary A < Rm*n ; m << n that is full rank. The dictionary elements are desired, but not required to be affine in the sense of providing scale, rotation and translation invariance.
- Error tolerance Є for the reconstructed signal.

Or

- Compression factor c which is the maximum percentage of coefficients

used to the current signal m
i.e. coefficients = (c/100) * m .

ΔE :

From equation 2.2.9.1 , as ΔE increases, the probability ρ decreases. We define FΔE as a function that calculates ΔE. FΔE can be

$F(r)\Delta E = \|R(r)\|2$ .....(2.2.9.2)

Where F (r)ΔE is the value of FΔE after iteration r, and $\|R(r)\|2$ is the l2 norm of the residual. Since the forward selection algorithm reduces the residual at each iteration, FΔE is a decreasing function in the number of iterations, thus the probability of performing a backward selection increases. This may be regarded as a correction phase towards the end of the algorithm, where several backward removal steps are made to eliminate bad choices and improve the results. However we need an annealing schedule that guarantees convergence of the algorithm.

Another choice would be

$F(r)\Delta E = \|x(r)\|2$ .....(2.2.9.3)

where $\|X(r)\|2$ is the l2 norm of the coefficient vector x. At each iteration, we add a coefficient to the coefficient vector, thus FΔE is an increasing function in the number of iterations.

Since we need to explore the subspace early enough in the selection process, and less often at the end, every addition to the coefficient vector will increase the value of ΔE , hence we decrease the probability of making a backward selection. This addition guarantees convergence of the algorithm. Similar to the above functions, we may define
two more that take the difference of the generating functions at the current and previous iterations:

$F(r)\Delta E = \|R(r)\|2 - \|R(r-1)\|2$ .....(2.2.9.4)

$F(r)\Delta E = \|x(r)\|2 - \|x(r-1)\|2$ .....(2.2.9.5)

The absolute value is taken because backward elimination steps will result in a negative change in energy.

Annealing schedule T :

An increase in T will increase the probability ρ until it saturates at a certain value. One possibility would be

$T = (k * s) / r$

where k is a constant (> 0) input to the algorithm. Also, since we may be performing backward elimination, the number of currently selected basis s is not necessarily equal to r. The more backward elimination steps we make, the smaller T will become, therefore the probability ρ will decrease, thus the algorithm will eventually terminate.

Result and analysis

Matching Pursuit (using gabor dictionary)

original image

image after ½ reduction
mse=38.2357
psnr=32.2959
cpu time=9.1719 sec

image after ¼ reduction
mse=67.1125
psnr=29.8628
cpu time=6.6563

Matching Pursuit(using Gabor and Haar dictionary)

image after ½ reduction
Mse=33.7667
Psnr=32.863
Cpu time=13.9688

Image after ¼ reduction
Mse=68.623
Psnr=29.7661
Cpu time=7.078

Orthogonal Matching Pursuit

image after ½ reduction
Mse=35.1091
Psnr=32.6766
Cpu time=14.5938

Image after ¼ reduction
Mse=58.5571
Psnr=30.455
Cpu time=10.037

Order Recursive Matching Pursuit

image after ½ reduction
Mse=35.1091
Psnr=32.6766
Cpu time=20.5938

image after ¼ reduction
Mse=58.5571
Psnr=30.455
Cpu time=14.557

Matching Pursuit with Simulated Annealing

image after ½ reduction
Msc=38.2357
Psnr=32.2959
Cpu time=9.1719

Image after ¼ reduction
Msc=67.1125
Psnr=29.8628
Cpu time=6.75 sec

III. PROPOSED ALGORITHM

E. Matching Pursuit using Genetic Algorithm:

The basics of this algorithm is to choose randomly an initial population, estimate and make the fittest pass untouched in the next generation, while the other individuals fight by pairs, and the fittest of each pair goes to the recombination pool. Then each of these individuals is randomly recombined with the others and the "sons" pass to the next generation with the fittest. This process is repeated until stability is reached, when only the fittest is kept and the other atoms randomly reinitialized. The algorithm implemented here has some modifications with respect to the one explained previously to maximally adapt it for the scope of this application .Stability is reached not by bit to bit comparison, but by making the decimal difference between the atoms and the best current solution. This is because in the specific definition of dictionary atoms, when individuals are really close is when their parameters have close definition values, and a bit to bit comparison looses this sense of proximity. Mutations are not bit to bit mutations, but are also implemented by randomly adding a decimal number that must be comprised into the variance intervals chosen by the user depending on their interests (big variance implies more mobility, but if the best solution is near the actual one, it may not appear in the next generation, while small variance gives more accuracy when the solution is near but less mobility, so less probabilities to reach the surroundings of the optimal result). All the parameters are discrete to be able to have a discrete dictionary of functions and an easiest way to code them.

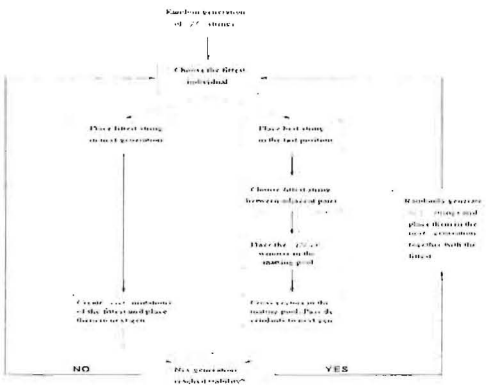
The algorithm follows the steps described below, which try to imitate the genetic evolution system :

1. Randomly generate *N* strings for the initial population (N _is a small odd number).
2. Evaluate the fitness of each string, select the best solution based on fitness, place itin position N and send it unchanged into the next generation.
3. Force strings that are adjacent to each other in the population to compete directlywith each other for the right to survive. Similarly, the fittest string between stringsU and U1 is placed in the mating pool (where U goes from 1 to N-1). When all the local competitors are held, we have (N-1)/2 strings in the mating pool, which are crossed with a probability of 1.0 and placed in the next generation. The localcompetitions enhance the Darwinian "survival-of-the fittest" aspect of the GA.They help ensure that the best solutions thrive in the population.
4. At this point there are ((N-1)/2 +1)strings in the next generation and the remaining\(N-1)/2 are generated in a random fashion, by mutating the best string with aprobability of p_{mutate} that is related with the variance interval introduced by the user.
5. Return to step 2 until convergence is achieved (that is when none of the strings inthe population differs from the population's current best solution by more than four units in every gene) or until the maximum number of iterations fixed by theuser is reached. If the maximum number of iteration is reached, the algorithm finishes here and returns the fittest individual in the actual generation. If stabilityhas been reached, the sixth step is executed.
6. Take the best solution and place it in a second "initial generation", generate the other N-1 strings in this second initial generation at random, and begin the cycle again until the maximum number of generations allowed is reached. This is necessary to have an evolutionary population, and so a population that is able of adapting to the environment. When a population has reached stability, it has no possibility of

adapting to the environment. Stability may be reached because the current solution is the best possible solution (possible, but not probable) or simply because in the population there were only weak individuals.

Reinitializing the population makes it more dynamic, and so more adaptive. So reinitializing the population, when keeping the stronger individual, will give more possibilities to reach a better solution by giving more dynamism to the population.

A general block diagram of this algorithm is given below:



F. Matching Pursuit with wavelet decomposition

Wavelet and Wavelet technique have recently generated much interest, both in applied areas as well as in mathematical ones. The class of Wavelet techniques is not really precisely defined and it keeps changing. Hence, it is virtually impossible to give a precise definition of wavelet that incorporates all different aspects. It is equally hard to write a comprehensive overview of wavelets. In this paper, we restricted to focus on wavelets which satisfy the important properties for image compression like orthogonal and ortho normal (db N and bior Nd, Nr class of wavelets). Wavelet theory involves representing general functions interims of simpler, fixed building blocks at different scales and positions. This has been found to be a useful approach in several different areas, like sub band coding, quadrature mirror filters, pyramid schemes,etc. Further, wavelets permit multi resolution analysis with the following properties: orthogonality, compact support, rational coefficients, symmetry, smoothness, number of vanishing moments and analytic expressions. Fig. 4 shows structural properties of db9 and bior6.8 as a specific case. These are the low pass and high pass filters used in the wavelet decomposition of image.

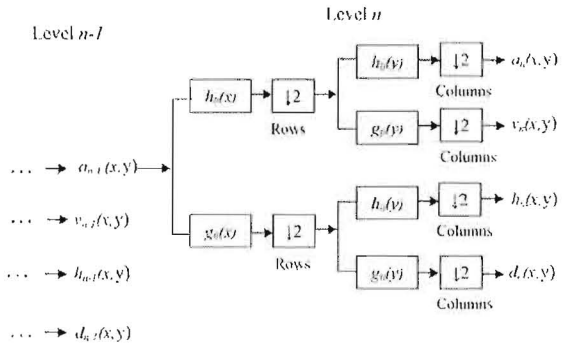
Wavelet Transform.:

Discrete Wavelet transform requires two sets of related functions called scaling function and wavelet function. Most of the scaling and wavelet functions are fractal in nature and iterative methods are required to see its shape. Most good and modern image compression schemes are based on the wavelet transform. Wavelets have been identified as basis functions or building blocks that can be used to produce all possible functions. They can also quickly de correlate data which can lead to a more compact representation than the original data. In addition, wavelets have the special feature that all the wavelet functions can be constructed from a single mother wavelet w(t), which is effectively a small wave. The other wavelet functions, wij(t) are obtained from the mother wavelet by translation and compression or dilation as shown below.

$$W_{ij}(t) = w(2^i t - j), \quad (5)$$

Where i and j represent the scaling (compression or dilation) and the translation parameters, respectively. The compression and translation allow the wavelet functions represent signals in multiple levels of time-frequency resolutions. The multi resolution representation of wavelets along with their orthogonal properties makes them attractive for representation of various functions. The wavelet transform is implemented in the discrete time domain by the discrete wavelet transform (DWT). The DWT can be implemented by passing the signal through a combination of low pass and high pass filters and down sampling by a factor of two to obtain a single level of decomposition. Multiple levels of the wavelet transform are performed by repeating the filtering and down sampling operation on low pass branch outputs. The 1-D wavelet transform can be extended to a two dimensional wavelet transform using separable filters. The 2-D transform is performed by applying a 1-D transform along the rows and then along the columns. An n-level 2-D wavelet decomposition of a signal is shown in Figure 2. In the figure, h0(x) and g0(x) represent the low pass and high pass filter responses, respectively. The filter outputs at level n are given by an, hn, vn, and dn, respectively. The low pass filter output from the previous level serves as the input for the next level of decomposition. The use of bi orthogonal wavelets has been found to be the most suitable for obtaining image wavelet transforms for compression applications. The bi orthogonal

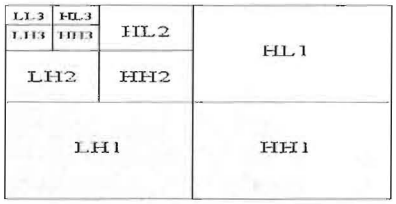
filters have linear phase and symmetric properties that are useful in avoiding boundary artifacts in images [8]. One of the most popular set of bi orthogonal filters are the Daubechies' 9/7 analysis and synthesis filters, used quite commonly for image compression applications. The 2-D wavelet transform, when applied to images, provides de correlation and yields a spatial-frequency distribution of the image.



n- level wavelet decomposition of a 2-D input (fig - 18)

where,

a (x, y) - approximation coefficient
v (x, y) – vertical coefficient
h (x ,y) – horizontal coefficient
d (x, y) – diagonal coefficient



3-Level dyadic wavelet pyramid(fig-19)

In general, most of the energy in the image tends to be concentrated in the low frequency regions with the detail sub bands containing the edge information. Furthermore the energy in the high frequency bands is concentrated into a relatively small number of coefficients. The probability distribution of wavelet coefficients in the high frequency sub bands has been shown to have a generalized Gaussian-like distribution with a high concentration of near zero valued coefficients . The spatial-frequency distribution of images due to the wavelet transform leads to an effective grouping of coefficients for compression. Above figure gives the 3- level dyadic wavelet pyramid representation of an image. Where suffix 1 indicates first level 2 is second level and 3 is third level of decomposition. For bi orthogonal property we consider only DbN and Bior Nd, Nr family of wavelets in our simulation.

After doing the decomposition the 4 coefficients are encoded using the general Matching Pursuit algorithm. After encoding it is again reconstructed by using wavelet reconstruction.

RESULT AND ANALYSIS

Matching Pursuit using Genetic Algorithm

Original image

Image after ½ reduction
Mse= 128.1421
Psnr= 27.0539
Cpu time= 93.5781

Image after ¼ reduction
Mse= 160.0850
Psnr= 26.0873
Cpu time= 35.9063

Matching Pursuit with wavelet decomposition

image after ½ reduction
Mse=(1.441)004
Psnr=6.5427
Cpu time=10.8594

Image after ¼ reduction
Mse=(1.441)004
Psnr=6.5427
Cpu time=10.8594

IV. CONCLUSION

In this paper, a survey of existing image compression methods using dictionary based approaches is presented. Different dictionary properties are also discussed and the effect of combining more than one dictionary is evaluated. A heuristic approach for signal compression is also proposed, that makes use of the well established matching pursuit algorithm, backward elimination approaches, and the simulated annealing heuristic search strategy, all combined in a novel approach. Different parameters for the approach are discussed. Results after applying the algorithm in image compression has shown superior performance in terms of reconstructed image quality, both in terms of PSNR and subjective quality, over the basic matching pursuit algorithm. This improvement is due to the exploration of the search space using simulated annealing, and reducing the adverse effect of greedy algorithms.

Modifications are made to the proposed algorithms that lead to even better results. Inspired by the orthogonal matching pursuit, the selected set of basis from the first algorithm are used to calculate the orthogonal projections to the selected basis.

Improvements are made to reduce the time consumption by taking the same dictionary using GA based approach and using wavelet decomposition. But the result is not better.

Several areas for improvement still exist for the novel approach presented. By incorporating linear algebraic methods, the backward elimination step in the algorithm will greatly reduce the computational complexity, and reduce execution time. The mpsa algorithm may also be experimented with different forward selection algorithms, such as Natarajan's algorithm or the orthogonal matching pursuit. These algorithms, which are better than the basic matching pursuit in terms of reconstructed quality, may provide better results, at the cost of increased computation. Basis selection is

perhaps the most time consuming process in the image compression sequence. The complexity will increase with the size of the dictionary. The performance of the algorithm may be enhanced by guiding the selection process based on the image block to be coded.

REFERENCES

[1] S. Mallat and Z. Zhang, "Matching Pursuit with time frequency dictionaries", IEEE Trans. on Signal Processing, Dec. 1993.

[2] Krishnaprasad, P. S., and Rezaiifar, Y. C. P. R. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. IEEE Computer (May 05 1993).

[3] Natarajan, B. K. Sparse approximate solutions to linear system. SIAM Journal on Computing 24, 2 (Apr. 1995), 227 {234}

[4] Ziyad, N. A., Gilmore, E. T., and Chouikha, M. F. Dictio-nary approaches to image compression and reconstruction. In IASTED International Conference on Signal and Image Processing (Sept. 1998).

[5] Chen, S. S., Donoho, D. L., and Saunders, M. A. Atomic de-composition by basis pursuit. SIAM Journal on Scienti¯c Computing 20, 1 (1999), 33-61.

[6] Tao Gan, Yanmin He and Weile Zhu ,SPARSE APPROXIMATION USING FAST MATCHING PURSUIT(2007)

[7] Thomas Blumensath, Member, IEEE, and Mike E. Davies, Member, IEEE, Gradient Pursuits(2008)

[8] Matching Pursuit through Genetic Algorithms by Rosa M. Figueras i Ventura, Pierre Vandergheynst

[9] Cotter, S. F., Adler, J., Rao, B. D., and Kreutz-Delgado, K. Forward sequential algorithms for best basis selection. IEE Proc. Vis. Imae Signal Process. 146, 5 (October 1999), 235{244.

[10] Cotter, S. F., Kreutz-Delgado, K., and Rao, B. D. Backward sequential elimination for sparse vector subset selection. Signal Process. 81, 5 (2001), 1849{1864.

[11] F. Bergeaud and S. Mallat, MATCHING PURSUIT OF IMAGES (1995)

[12] Michael J. Horowitz and David L. Neuhoff, Image Coding by Matching Pursuit and Perceptual Pruning by (1997)

[13] Przemysław Czerepin'ski, Nishan Canagarajah, David Bull. Matching Pursuits Video Coding (2000)

[14] Yang Liu and Deva K. Borah. Estimation of Time-Varying Frequency-Selective Channels Using a Matching Pursuit Technique (2003)

[15] Rosa M. Figueras i Ventura. Pierre Vandergheynst. Pascal Frossard and Andrea Cavallaro .COLOR IMAGE SCALABLE CODING WITH MATCHING PURSUIT(2004)

[16] UNDERDETERMINED NOISY BLIND SEPARATION USING DUAL MATCHING PURSUITS by Paul Sugden and Nishan Canagarajah (2004)

[17] Fast Matching Pursuit Video Coding by Combining Dictionary Approximation and Atom Extraction by Jian Liang Lin , Wen Liang Hwang . Soo-Chang Pei (2007)

[18] A New Method of weak signal detection based on Improved Matching Persuit Algorithm by Gang Xu and Jia Gao (2008)

[19] IRIS RECOGNITION BASED ON MATCHING PURSUITS by Yuch-Shiun Lee. Jinn Ho. Wen-Liang Hwang. and Chung-Lin Huang

[20] Matching Pursuit-Based Region-of-Interest Image Coding by Abbas Ebrahimi-Moghadam and Shahram Shirani. Senior Member. IEEE (2007)