I.J. Image, Graphics and Signal Processing, **2014, 10, 36-42**
Published Online September 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2014.10.05

Block-Based Compressive Sensed Thermal Image Reconstruction using Greedy Algorithms

Usham V. Dias
Dept. of Electronics & Telecommunication,
Padre Conceicao College of Engineering, Goa, India
Email: ushamdias@gmail.com

Milind E. Rane
Dept. of Electronics & Telecommunication,
Vishwakarma Institute of Technology, Pune, India
Email: me_rane@yahoo.com

Abstract—This paper implements a block based compressive sensing technique for thermal image reconstruction using greedy algorithms. A total of fourteen different sensing patterns were tested for data acquisition. Orthogonal Matching Pursuit (OMP) and Regularized Orthogonal Matching Pursuit (ROMP) with two different thresholds were implemented for image reconstruction with OMP having an edge over ROMP in terms of error and PSNR. ROMP was faster in terms of iterations needed for reconstruction. As the threshold for ROMP was increased the number of iterations needed decreased. Gaussian, Bernoulli and Hadamard patterns were the best for reconstruction. Hadamard matrix, Bernoulli matrix with +/-1 entries and Bernoulli matrix with 0/1 entries have the added advantage of being more conducive for hardware implementation. This paper used Discrete Cosine Transform as the sparsifying basis for reconstruction.

Index Terms—Compressive sensing, greedy reconstruction, sensing pattern, Regularized Orthogonal Matching Pursuit.

I. Introduction

Compressive sensing is a data acquisition technique where data is sampled using linear projections at a rate much below than the Nyquist rate. This is possible if the signal is sparse in a known domain, and accordingly sensing matrices can be designed to take linear projections of the data or signal under consideration[1-2].

In [3], fourteen different sensing matrices have been considered for acquiring the signal, some of which have the potential for object specific reconstruction. This paper tests all the fourteen sensing patterns using Orthogonal Matching Pursuit (OMP) and Regularized Orthogonal Matching Pursuit (ROMP) as reconstruction algorithms.

A brief survey of compressive sensing was conducted in [4]. Object specific reconstruction based on the reconstructed algorithm for surveillance was proposed in [5]. The different sensing matrices implemented in [3] are based on [6-9]. The combinatorial algorithms for reconstruction are not efficient for real time applications hence this paper implements greedy algorithms for reconstruction [10-12].

The next section explains the block based compressive sensing process, section III explains the greedy reconstruction algorithm, section IV calculates the sparsity of the database, followed by experimental results and conclusion in section V and VI respectively.

II. Block Based Compressive Sensing

This paper focuses on block based compressive sensed image reconstruction. A 256x256 image is divided into 8x8 blocks which gives a total of 1024 blocks in a single frame. Each of these blocks is reconstructed based on the compressive sensing paradigm. All the reconstructed blocks are combined to generate the frame. This process is repeated across all frames in the video under test. This paper tests fourteen different sensing patterns described in [3] for acquiring the data. Greedy reconstruction algorithms implemented in this paper include Orthogonal Matching Pursuit and Regularized Orthogonal Matching Pursuit. The sparsifying basis assumed is Discrete Cosine Transform.

III. Greedy Reconstruction Algorithm

A. Orthogonal Matching Pursuit

Orthogonal Matching Pursuit (OMP) algorithm is given below based on [10].

Let Φ be the sensing matrix.

Let Ψ the sparsifying basis.

Let $A=\Phi\Psi^{-1}$ be an MxN matrix.

Let $A_\lambda \epsilon R^M$ be the λth column of A.

Let $A_S \epsilon R^{MxS}$ be the Matching Pursuit estimation matrix where S is the sparsity which is also equal to the number of iteration.

Let $y \epsilon R^M$ be the acquired data

Let $yp \epsilon R^M$ be the estimate of y.

Let $yr \epsilon R^M$ be the residual error in estimating y.

Let x' ϵ R^N be the estimate of x.
Let x'$_k$ ϵ R be the k^{th} element of x'.

1. A_S =0, *yr*=y, k=0.
2. k=k+1;
3. Take dot product of *yr* with every column of A and find the index λ which corresponds to the highest dot product value and which was not selected earlier.
4. Augment A_S with A_λ
5. Perform least squares estimation to calculate *x'* using A_S.

$$x' = (A_S^T A_S)^{-1} A_S^T y$$

6. Take the projection of *x'* onto A_S to calculate *yp*

$$yp = A_S x'$$

7. Update the residual *yr*

$$yr = y - yp$$

8. Perform steps 2 to 7 until k=s.

Finally we get *x'* which is the sparse signal in Ψ domain using sensing matrix Φ.

B. Regularized Orthogonal Matching Pursuit

Regularized Orthogonal Matching Pursuit (ROMP) algorithm is given below based on [10].
Let Φ be the sensing matrix.
Let Ψ the sparsifying basis.
Let $A=\Phi\Psi^{-1}$ be an MxN matrix.
Let $A_\lambda \epsilon R^M$ be the λth column of A.
Let $A_S \epsilon R^{MxS}$ be the Matching Pursuit estimation matrix where S is the sparsity.
Let y ϵ R^M be the acquired data.
Let yp ϵ R^M be the estimate of y.
Let yr ϵ R^M be the residual error in estimating y.
Let x' ϵ R^N be the estimate of x.
Let x'$_k$ ϵ R be the kth element of x'.

1. A_S=0, yr=y, k=0, index= empty.
2. k=k+1;
3. Take dot product of yr with every column of A and sort the correlated data along with its index in descending order.
4. Select S highest correlated data along with indices (intermediate indices)
5. For each of these intermediate indices, find all those indices which lie about it based on some threshold and calculate its norm.
6. Select that set of indices which has the highest norm (current indices).
7. Find the union: index= index $\cup$ current indices.
8. Update A_S: $A_S=A_{index}$
9. Perform least squares estimation to calculate x' using A_S.

$$x' = (A_S^T A_S)^{-1} A_S^T y$$

10. Take the projection of x' onto A_S to calculate yp

$$yp = A_S x'$$

11. Update the residual yr

$$yr = y - yp$$

12. Perform steps 2 to 11 until number of supports is less than s.

Finally we get x' which is the sparse signal in Ψ domain using sensing matrix Φ. Two thresholds i.e. 10% and 20% about the intermediate index under consideration are used for testing.

IV. Sparsity Calculation For Database

IEEE OTCBVS WS Series Bench is an OTCBVS Benchmark Dataset Collection, which has been used for the experiments. The codes were tested for Dataset 01: OSU Thermal Pedestrian Database which had 10 different sequences [13].

The sparsity in DCT domain for the above mentioned 10 different video sequences for different block sizes viz. 8x8, 16x16, 32x32 and 64x64, and different energy compactions viz. 99.9%, 99%, 95% and 90% are given in table I. The parameters calculated are defined below.

Average Frame Sparsity (FS*ave*), indicates on an average how many coefficients are needed to reconstruct the frame with a given energy compaction. Average Block Sparsity (BS*ave*), indicates on an average how many coefficients are needed to reconstruct the block with a given energy compaction. Least Sparsity of Block across the video sequences (LS*block*), indicates that even though the average sparsity is low, there are some blocks which are not very sparse. It gives the maximum number of coefficients needed to reconstruct the block with a given energy compaction.

The defined parameters are calculated for each of the 10 video sequences and for each of the energy compaction value. The results in table I are the average across all the 10 video sequences for each block size.

V. Experimental Results

The parameters used for quantitative analysis have been defined below:

1. **Relative Error:** Relative error gives an indication of how good the reconstructed image is compared to the original, relative to the size of the image under consideration.

$$\text{Error}_{\text{block}} = \frac{\text{norm}(x - xs)}{\text{norm}(xs)} \quad (1)$$

where *x* is the reconstructed signal and *xs* is the original signal. Error $_{block}$ is the relative error to reconstruct the block.

Average Relative error across the video:

$$\text{Error}_{ave} = \frac{\sum_1^f \text{Error}}{f} \quad (2)$$

where f is the number of frames corresponding to a particular video and Error is the sum of the relative error for each block in the frame. Error $_{ave}$ is the average relative error across the video

Average Relative error across the Database:

$$\text{Error}_{database} = \frac{\sum_1^v \text{Error}_{ave}}{v} \quad (3)$$

where v is the number of videos and Error $_{database}$ is the average relative error across the database.

2. **Peak Signal-to-Noise Ratio(PSNR) in db:** PSNR gives a measure of the mean square error with respect to the maximum signal value. The higher the PSNR, the better the reconstructed image

Table 1. Average Sparsity of the entire database in terms of 8x8, 16x16, 32x32 and 64x64 blocks for different energy compactions

Energy →	99.9%			99%			95%			90%		
Block size ↓	FS_{ave}	BS_{ave}	LS_{block}	FS_{ave}	BS_{ave}	LS_{block}	FS_{ave}	BS_{ave}	LS_{block}	FS_{ave}	BS_{ave}	LS_{block}
8x8	13960.63	13.633	49	2016.1	1.969	30	1162.44	1.14	12	1070.97	1.05	7
16x16	13081.22	51.099	157	988.01	3.859	72	307.88	1.2	19	261.507	1.02	6
32x32	12647.5	197.62	488	602.08	9.407	120	80.62	1.26	19	65.3757	1.02	7
64x64	12401.33	775.08	1529	414.33	25.9	271	19.2218	1.2	21	16.1123	1.01	4

matches the original image and the better is the reconstruction algorithm.

$$\text{PSNR}_{block} = 10\log_{10}\frac{(2^r-1)^2}{MSE} \quad (4)$$

Where r=8, is the number of bits required to represent the original image and MSE is the Mean Square Error between the original and the reconstructed signal. PSNR $_{block}$ is the peak signal to noise ratio of the block.

Average PSNR across the video:

$$\text{PSNR}_{ave} = \frac{\sum_1^f P}{f} \quad (5)$$

Where f is the number of frames corresponding to a particular video and P is the sum of the peak signal-to-noise ratio for each block in the frame. PSNR $_{ave}$ is the average PSNR.

Average PSNR across the Database:

$$\text{PSNR}_{database} = \frac{\sum_1^v \text{PSNR}_{ave}}{v} \quad (6)$$

Where v is the number of videos and PSNR $_{database}$ is the average PSNR across the database.

3. **Amount of Saving*:***

Savings per frame (in pixels):

$$S_{per\ frame} = (I_{rows} \times I_{cols}) - (blocks \times m) \quad (7)$$

Where $\boldsymbol{I_{rows}}$ is the number of rows in the image, $\boldsymbol{I_{cols}}$ the number of columns, ***blocks*** the number of blocks per frame and ***m*** is the number of measurements per block.

Savings per video (in pixels):

$$S_{per\ video} = S_{per\ frame} \times \text{no. of frames} \quad (8)$$

Percentage saving per frame:

$$S_{\%\ per\ frame} = \frac{S_{per\ frame}}{I_{rows} \times I_{cols}} x100 \quad (9)$$

4. **Algorithm Reconstruction Time**

Let T_R be time in sec taken by the algorithm to reconstruct the entire video then,

Average time taken to reconstruct each frame in the video (TRave):

$$T_{Rave} = \frac{T_R}{f} \quad (10)$$

Where f is the number of frames.

Average time taken to reconstruct each frame in the database (TRdatabase):

$$T_{Rdatabase} = \frac{\sum_1^v T_{Rave}}{v} \quad (11)$$

Where v is the number of videos.

5. Number of Iterations for Reconstruction

Let k be the number of iterations to reconstruct a block,
Let k_{frame} be the number of iterations to reconstruct a frame,
Let k_{video} be the number of iterations to reconstruct a video,

Iterations needed to reconstruct a frame averaged across the entire video ($k_{ave\ per\ frame}$):

$$k_{ave\ per\ frame} = \frac{k_{video}}{f} \quad (12)$$

where f is the number of frames.

Iterations needed to reconstruct a frame averaged across the entire database ($k_{per\ frame(database)}$):

$$k_{per\ frame(database)} = \frac{\sum_{1}^{v} k_{ave\ per\ frame}}{v} \quad (13)$$

Where v is the number of videos.

The results are analysed for measurements m=4s per block, where s is the sparsity of each block which is assumed to be 10. The graphs in the figure display the Relative error, PSNR and iterations needed for reconstruction using OMP, ROMP-10% and ROMP-20% for each of the fourteen sensing matrices across the entire database. The average algorithm reconstruction time per frame for the entire database averaged across all the sensing matrices for 8x8 block reconstruction of a 256x256 image is given in fig. 15.

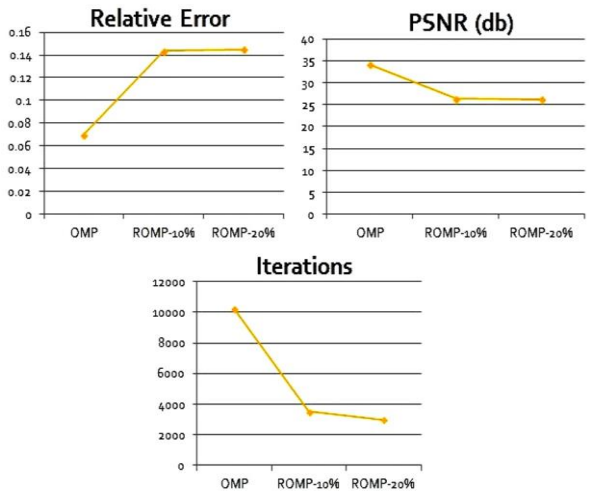

Fig. 1. Two-band QMF bank

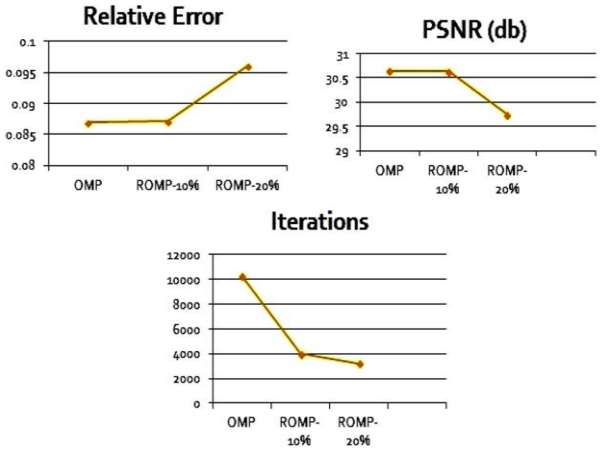

Fig. 2. Comparison of greedy algorithms for Gaussian (Orthogonal) Sensing matrix

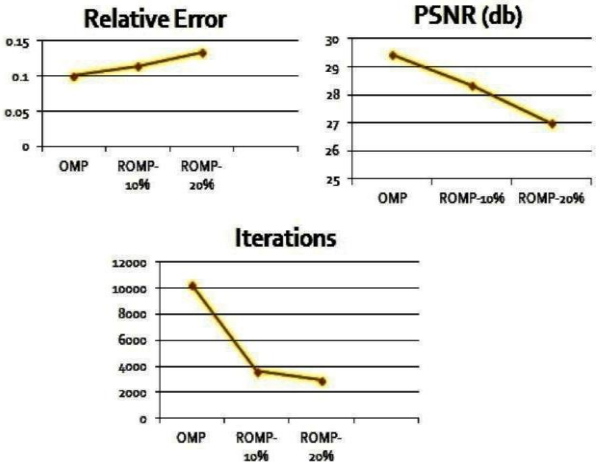

Fig. 3. Comparison of greedy algorithms for Bernoulli random (+/- 1 entries) Sensing matrix

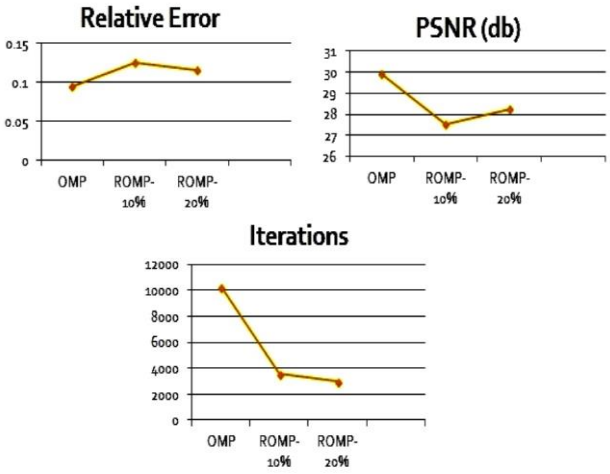

Fig. 4. Comparison of greedy algorithms for Bernoulli random (1/0 entries) Sensing matrix

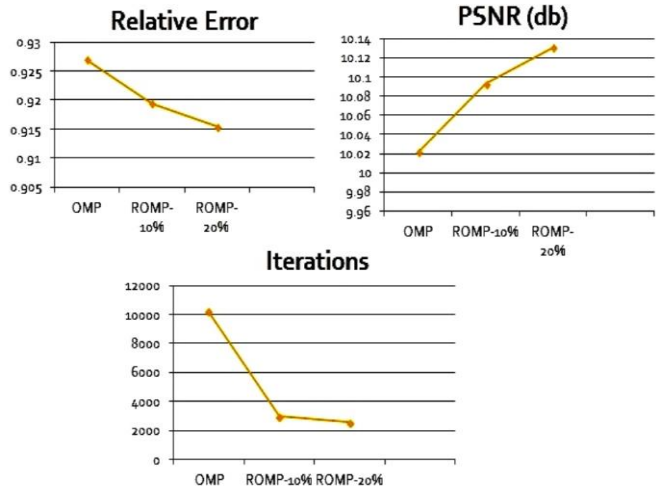

Fig. 5. Comparison of greedy algorithms for Fourier Sensing matrix

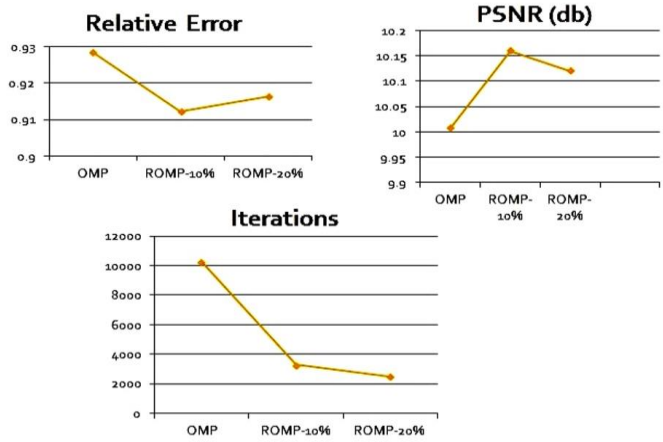

Fig. 6. Comparison of greedy algorithms for Fourier (without dc) Sensing matrix

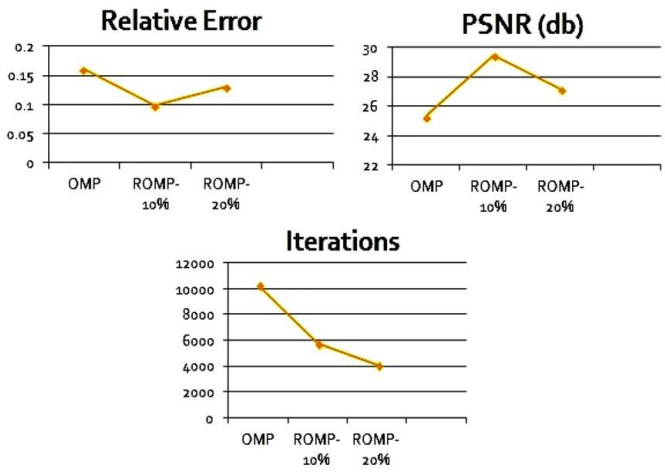

Fig. 7. Comparison of greedy algorithms for Toeplitz (gaussian random) Sensing matrix

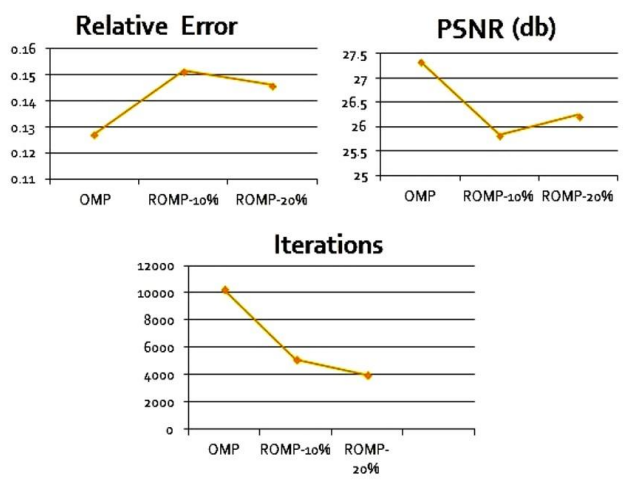

Fig. 8. Comparison of greedy algorithms for Toeplitz (bernoulli) Sensing matrix

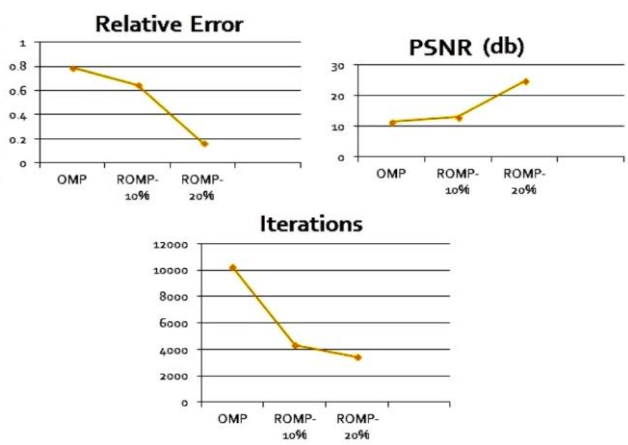

Fig. 9. Comparison of greedy algorithms for Circular (gaussian) Sensing matrix

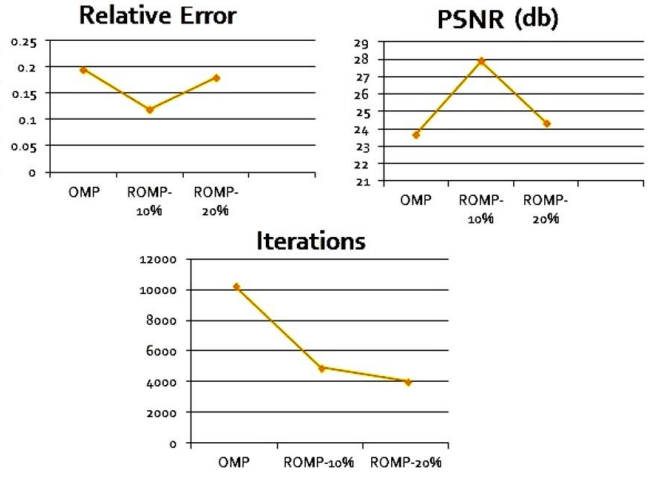

Fig. 10. Comparison of greedy algorithms for Circular (bernoulli) Sensing matrix

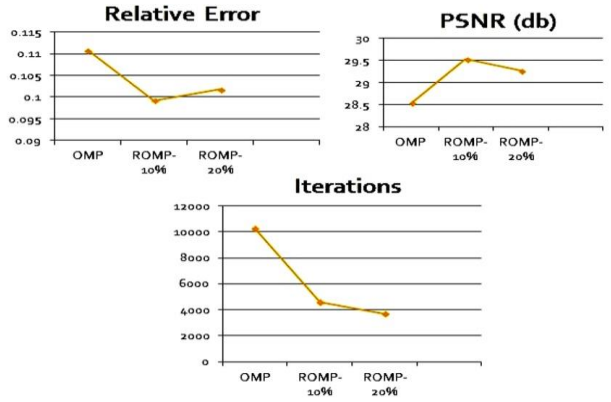

Fig. 11. Comparison of greedy algorithms for Hadamard Sensing matrix

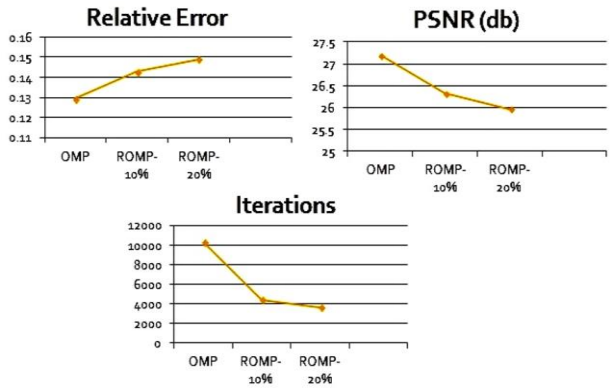

Fig. 12. Comparison of greedy algorithms for Hadamard (Normalised) Sensing matrix

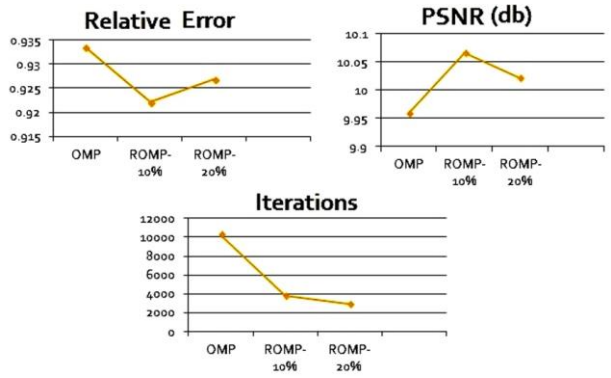

Fig. 13. Comparison of greedy algorithms for Hadamard (without dc) Sensing matrix

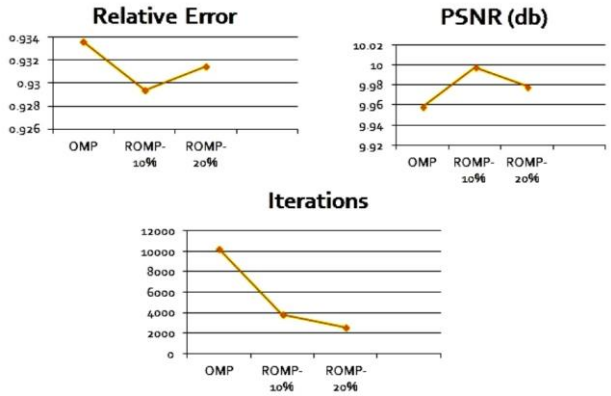

Fig. 14. Comparison of greedy algorithms for Hadamard without dc (Normalised) Sensing matrix

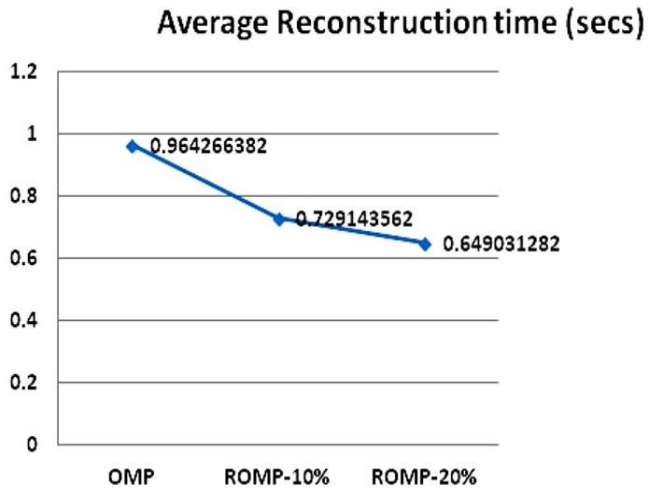

Fig. 15. Average reconstruction time for Greedy algorithms

VI. Conclusion

Among the Reconstruction algorithms, OMP is the best followed by ROMP 10% and finally ROMP 20% in terms of error and PSNR. In terms of iterations ROMP 20% is the best followed by ROMP 10% and OMP. As the threshold for ROMP increases (10%, 20% and above), the iterations required for reconstruction reduces but at the cost of increased error and lower PSNR. In general, an iteration of ROMP would consume more time than an iteration of OMP. Hence efficient implementation of ROMP would be the key for faster reconstruction.

The 8x8 block based reconstruction analysis done in this paper, can be extended to 16x16, 32x32 and 64x64 block based frame reconstruction by selecting sparsity and measurements 4, 16 and 64 times that used for 8x8 block, maintaining the same image reconstruction quality.

Gaussian, Bernoulli and Hadamard are the best sensing matrices for reconstruction using greedy algorithms. Hadamard has the added advantage of being deterministic. Bernoulli (+/-1 entries and 1/0 entries) and Hadamard are much more friendly for hardware implementation compared to Gaussian.

Fourier sensing matrix gives a lot of distortion possibly due to high coherence between the sensing matrix and DCT basis. Toeplitz and Circulant matrices were random in their behaviour and need more testing to validate its use as a reliable sensing matrix.

Object specific reconstruction is possible by using Fourier and Hadamard sensing matrices without dc component because it removes the background information and preserves the object thus easing the process of object detection. But this technique does not provide high saving in terms of measurements.

As compared to [3] which reconstructs an image of 64x64 size with much larger time consumption which is further sensitive to the kind of sensing matrix selected, this paper reconstructs a large frame of size 256x256 in 0.964, 0.729 and 0.649 seconds using OMP, ROMP 10% and ROMP 20% respectively.

REFERENCES

[1] Mark A. Davenport, et al, "Introduction to compressed sensing", in *Compressed Sensing: Theory and Applications*, Cambridge University Press, 2012. [Amazon.com].

[2] Donoho D L., "Compressed sensing", *IEEE Transactions on Information Theory*, vol. 52(4), pp. 1289-1306, 200.

[3] Dias, Usham, and Milind E. Rane. "Comparative Analysis of Sensing Matrices for Compressed Sensed Thermal Images", *IEEE International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s)*, pp. 265 – 270, 2013.

[4] Usham Dias, Milind Rane, S. R. Bandewar, "Survey of Compressive Sensing*", International Journal of Scientific & Engineering Research*, Volume 3, Issue 2, February-2012, ISSN 2229-5518.

[5] Abhijit Mahalanobis, Robert Muise, "Object Specific Image Reconstruction using a Compressive Sensing Architecture for Application in Surveillance Systems", *IEEE transactions on aerospace and electronic systems*, vol. 45, no. 3 July 2009.

[6] Wotao Yina, Simon Morganb, Junfeng Yangc, Yin Zhanga; "Practical Compressive Sensing with Toeplitz and Circulant Matrices", Dept. CAAM, Rice University.

[7] Shunliao Yang, et al, "The Compressive Sensing based on Biorthogonal wavelet Basis", *IEEE International Symposium on Intelligence Information Processing and Trusted Computing*, pp. 479-482, 2010.

[8] Fan Yang; Shengqian Wang; Chengzhi Deng; "Compressive Sensing of Image Reconstruction Using Multi-wavelet Transforms", *IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, vol. 1, pp. 702 – 705, 2010.

[9] Gesen Zhang et al, "Compressed Sensing and Reconstruction with Bernoulli matrices", *IEEE International Conference on Information and Automation*, pp. 455-460, 2010.

[10] Deanna Needell, "Topics in Compressed Sensing," Dissertation for Doctor of philosophy in Mathematics, University of California, Davis, 2009.

[11] Philip Breen, "Algorithms for Sparse Approximation", School of Mathematics, University of Edinburgh, Year 4 Project, 2009.

[12] Ming-Jun Lai, "On Sparse Solutions of Underdetermined Linear Systems," Department of Mathematics, the University of Georgia, Athens, GA 30602, January 17, 2009.

[13] IEEE OTCBVS WS Series Bench. [Online]. Available: http://www.cse.ohio-state.edu/otcbvs-bench/.

[14] Radu Berinde, Piotr Indyk; "*Sparse recovery using sparse random matrices*",MIT, April 26, 2008.

[15] Xingxiu Li, Zhihui Wei, Liang Xiao, Yubao Sun, Jian Yang, "Compressed sensing image reconstruction based on morphological component analysis", *IEEE* 2009.

[16] Wu, J.; Liu, F.; Jiao, L. C.; Wang, X.; Hou, B., "Multivariate Compressive Sensing for Image Reconstruction in the Wavelet Domain: Using Scale Mixture Models", *IEEE Transactions on Image Processing*, Vol.20, Issue 12 , pp. 3483 – 3494, 2011.

[17] E. Candes and T. Tao. "Decoding by linear programming", *IEEE Trans. Inform. Theory*, vol. 51(12), pp. 4203-4215, 2005.

[18] Jean-Luc Starck, Fionn Murtagh, Jalal M. Fadili, *"Sparse image and signal processing: Wavelets, Curvelets, Morphological Diversity"*,Cambridge university press, 2010.

[19] Bubacarr Bah, "Restricted Isometry Property (RIP)," Graduate School of Mathematics, University of Edinburgh, First-Year Report, September 2009.

Authors' Profiles

Usham V. Dias has received his Bachelors Degree in Electronics and Telecommunication from Padre Conceicao College of Engineering, Goa in 2009 and Masters Degree in Electronics and Telecommunication with specialization in Signal Processing from Vishwakarma Institute of Technology, Pune in 2013. His research interests include Compressive Sensing, Signal & Image processing.

Milind E. Rane received his BE degree in Electronics Engineering from University of Pune and M Tech in Digital Electronics from Visvesvaraya Technological University, Belgaum, in 1999 and 2001 respectively. His research interest includes image processing, pattern recognition and BiometricsRecognition.

How to cite this paper: Usham V. Dias, Milind E. Rane,"Block-Based Compressive Sensed Thermal Image Reconstruction using Greedy Algorithms", IJIGSP, vol.6, no.10, pp.36-42, 2014.DOI: 10.5815/ijigsp.2014.10.05

