Compressed Sensing in the Image Domain
Mid-point Progress Report

David Fridovich-Keil and Grace Kuo

I. Introduction

Over the last decade or so, there has been a lot of progress in the field of compressed sensing. Indeed, the entire field is only about ten years old. The term "compressed sensing" essentially refers to the idea that some signals – specifically those that are sparse in some basis – can be reconstructed from fewer samples than the amount dictated by the Nyquist sampling theorem. This has been demonstrated with some success in the audio and MRI domains, e.g. in [1] and [2] respectively.

However, there is actually more to the notion of "compressed sensing" than compression in post-processing. The idea actually extends to sensing the signal in the first place, where, given a sparse signal it is hypothesized that one can sample a sufficient number of random basis elements directly and be guaranteed within some probabilistic bound that it is possible to reconstruct the signal [3].

This leads us to consider images, where there are many obvious applications. For example, one could imagine a simple compressive approach that might be able to shrink image file sizes smaller than, say, JPEG. Alternatively, compressed sensing could be used in a new CMOS sensor to improve the band rate and reduce power consumption by directly sensing the image in compressed form and transmitting only the compressed version off-chip.

Unfortunately, there has not been very much work in the image domain, and consequently the best results are not very encouraging. Preliminary results have limited scalability, low SNR, and low compression ratios compared with other compressive sensing applications [4]. In addition, most of the research in this area has been on designing cameras to implement compressive sensing schemes [4], [5]. In contrast, we focus on techniques for 2D image compression and reconstruction.

Compressed sensing is very closely related to convex optimization. One natural formulation is to minimize the L_2 norm of reconstruction error, penalizing some measure of the sparsity of the representation (e.g. it's L_1 norm). In the following sections, we discuss our formulation of the problem and several incremental approaches to solving it, with preliminary results.

II. Problem Formulation

We formulate compressed sensing problem as follows. Suppose that there exists some basis Ψ (represented as a matrix) such that the image y (as a column vector) is sparse in Ψ, i.e. $\exists x$ s.t. x is k-sparse (x has at most k non-zero elements) and the following equation holds:

$$y = \Psi x, ||x||_0 \leq k$$

In general, however, what we "measure" is not the true image y, but actually some transformed version which we denote m. We assume a linear measurement model, which we represent with the matrix A. We write:

$$m = Ay = A\Psi x$$

In general, this model places no restrictions on the matrix A. However, in the case of compressed sensing, we are interested in *random* matrices A, which we refer to as "mixing matrices." By using a random matrix, we represent y with respect to a basis whose vectors are random linear combinations of the original basis, Ψ. Moreover, we allow A to be of less than full rank. This is useful for compressed sensing because, *a priori*, we do not know the basis Ψ – the best we can do is make an educated guess. Also, allowing low-rank mixing matrices lets us model the *sensing* process by reducing the image to an arbitrary number of linear samples.

Now, we wish to find x given m and some guess of a sparse basis, $\hat{\Psi}$. In particular, we wish to minimize the L_2 distance between the reconstructed measurements and the actual measurements, while penalizing the number of non-zero entries in x by some non-negative parameter α. We cast this as the following optimization problem:

$$x^* = \arg\min_x \|A\hat{\Psi}x - m\|_2^2 + \alpha\|x\|_0 \qquad (1)$$

Of course, this problem is not generally tractable for arbitrary bases $\hat{\Psi}$. In the following sections, we consider variations on this problem in which there exist tractable (or even closed-form) solutions or approximations.

III. Preliminary Results

So far, we have implemented several compression schemes in an open-source Python repository, hosted at: https://github.com/dfridovi/compressed_sensing. These include:

1) *Pure Fourier compression* – Here, we take an image and compute its Fourier transform, then zero out all but the largest k coefficients, and reconstruct.
2) *Pure DCT compression* – As above, but here we replace the Fourier transform with the discrete cosine transform.
3) *Sketching in the image domain* – Here, we suppose that the image is sparse in the image domain, sketch

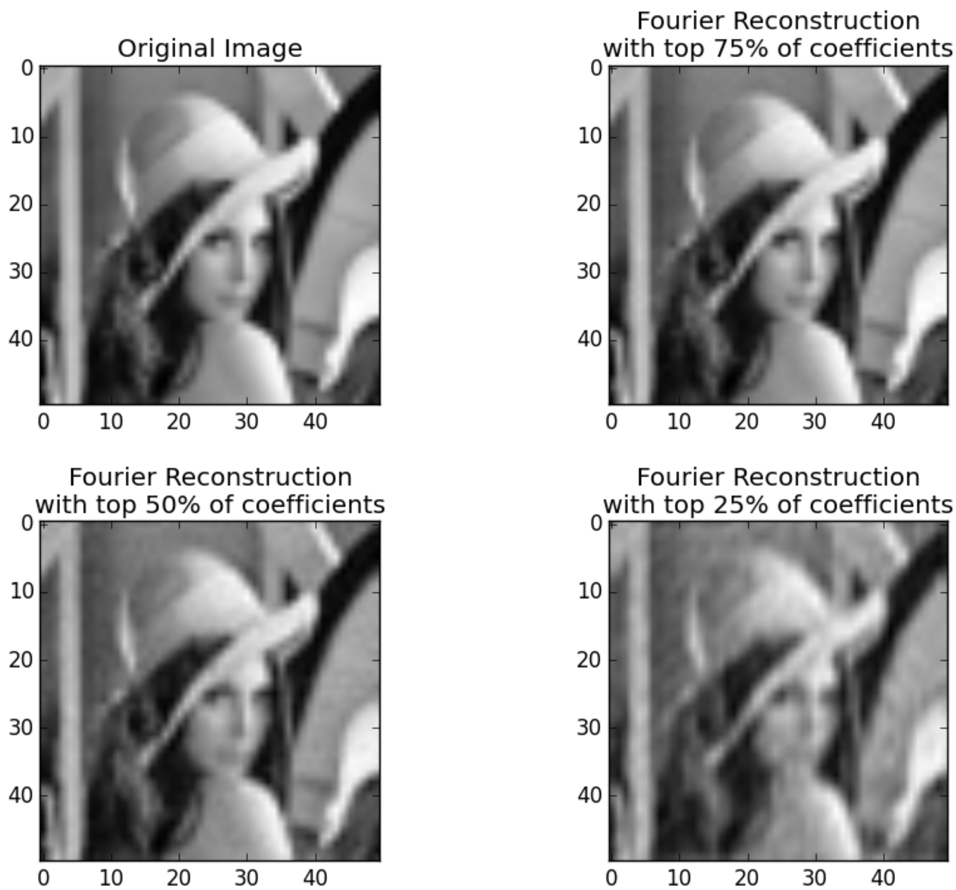

Fig. 1. Reconstruction results for pure Fourier compression. From the top left to the bottom right we iteratively reduce the number of non-zero coefficients in the Fourier domain. Predictably, the reconstruction fidelity degrades once we remove too many coefficients.

the standard basis, and compute an L_1-sparse reconstruction.

4) *Sketching in the DCT domain* – As above, but here we suppose that the image is sparse in the DCT domain and sketch the DCT basis before computing an an L_1-sparse reconstruction.

The following subsections describe each of these approaches in greater detail.

A. Pure Fourier Compression

We reformulate equation (1) as follows:

$$x^*_{Fourier}(k) = \arg\min_x \|Fx - m\|_2^2 : \|x\|_0 \leq k \qquad (2)$$

Here, we have set A to the identity, replaced $\hat{\Psi}$ with the Fourier basis, F, and replaced the cardinality penalization with a constraint that the cardinality of x must not exceed an arbitrary k, i.e. that x be k-sparse. Although the L_0 "norm" is not truly a convex function – meaning that neither (1) nor (2) is a convex problem – we can still see that there is some connection between the parameters α and k. In particular, for sufficiently high α, the solution to (1) is the zero vector, and for $k = 0$ the solution is also the zero vector. In general, increasing α forces more elements of x^* to zero, which corresponds to a lower k.

Now that we have eliminated the mixing matrix, reformulated the cardinality penalization as a constraint, and assumed that basis $\hat{\Psi}$ is Fourier (and in particular that it is orthogonal), we can solve this problem in closed form.

The intuition is simple: at optimum, we must have exactly k non-zero elements in x (otherwise the constraint is not active). Since the basis F is orthogonal, we can simply project m onto F (i.e. take the Fourier transform, $F^T m$) and zero out all but the largest k coefficients in the result. The result is precisely $x^*_{Fourier}(k)$.

More formally, we can prove the result by recognizing that the problem is identical to:

$$\arg\min_x \|x - F^T m\|_2^2 : \|x\|_0 \leq k$$

where $\hat{x} = F^T x$ is the Fourier transform of x and $F^T m$ is the Fourier transform of m. We are able to rewrite the objective function in the Fourier domain because Parseval's identity ensures the invariance of the squared L_2 norm under

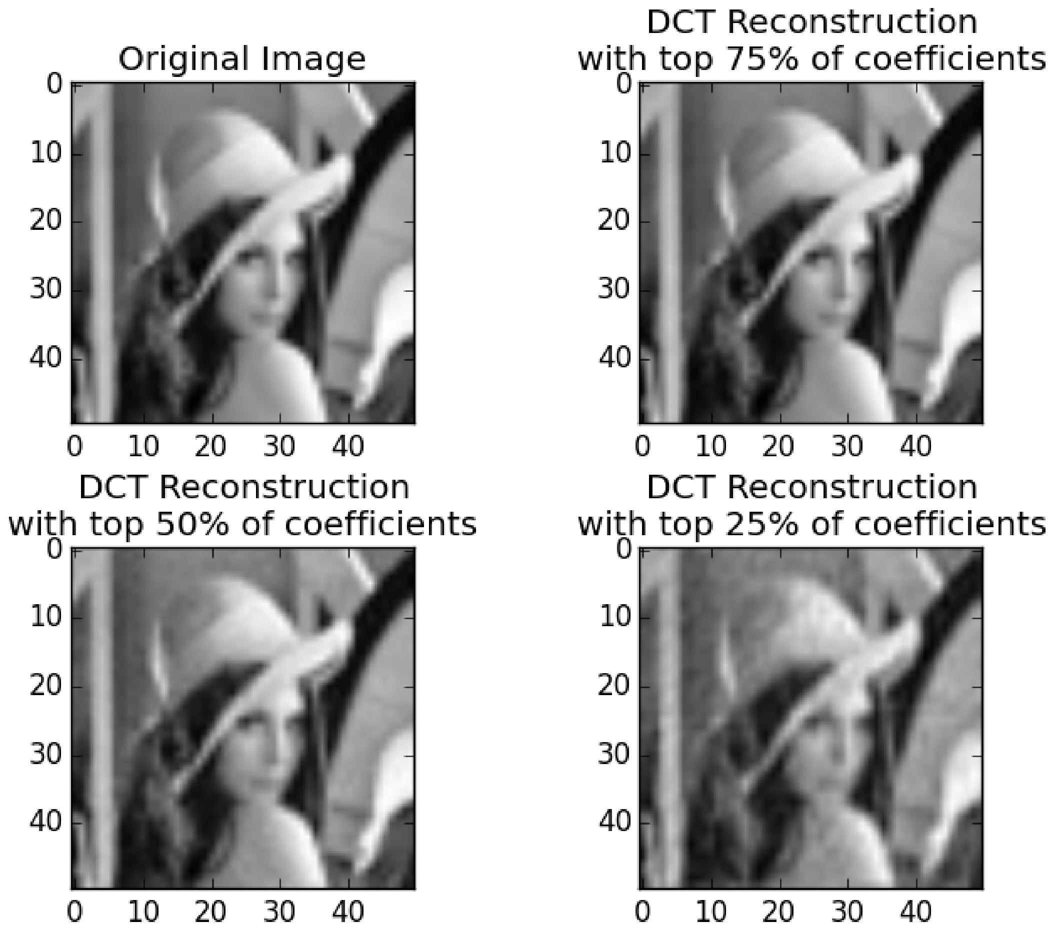

Fig. 2. Reconstruction results for pure DCT compression. From the top left to the bottom right we iteratively reduce the number of non-zero coefficients in the DCT domain. Predictably, the reconstruction fidelity degrades once we remove too many coefficients.

the Fourier transform. More generally, this holds for any unitary transformation $U : U^*U = I$ because $\|Uv\|_2^2 = v^*U^*Uv = \|v\|_2^2$.

In this new form, it is clear that the optimal x must share the same largest k entries as F^Tm, but have the rest of its elements equal to zero.

We have implemented this simple algorithm on the Lenna test image at a resolution of 50x50 pixels for various k. The reconstruction results are shown in Fig. 1.

B. Pure DCT Compression

Eventually, we will relax some of our rather strong assumptions in the previous subsection. In particular, we will eventually want to solve (1) as nearly as possible, only relaxing the non-convex cardinality penalty to a convex function (like the L_1 norm). In order to solve the problem using traditional tools like CVX, we will be forced to use only real numbers. Unfortunately, using the Fourier transform forces all of our vectors and matrices to be complex-valued.

In order to circumvent this rather artificial dilemma, we replace the Fourier transform with the discrete cosine transform, or DCT. The DCT basis is still orthogonal, so we can solve (2) in exactly the same way if we replace the Fourier basis with the DCT basis. Specifically, the same algorithm is optimal: take the DCT of m, then zero out all but the largest k components.

The results are shown in Fig. 2 for the same values of k, again using the 50x50 pixel Lenna test image.

C. Sketching in the image domain

Now, we consider a different sort of simplification of (1): namely, we set $\hat{\Psi} = I$. That is, we assume that we do not know of any basis in which the image is more likely to be sparser than it is in the standard basis. We also relax the cardinality penalty to an L_1 cost. This gives us:

$$x^* = \arg\min_x \|Ax - m\|_2^2 + \alpha\|x\|_1 \tag{3}$$

The problem above is convex and real-valued, and moreover it is in standard LASSO form, which means that it can be solved using an off-the-shelf QP solver. We use the Python package CVXPY.

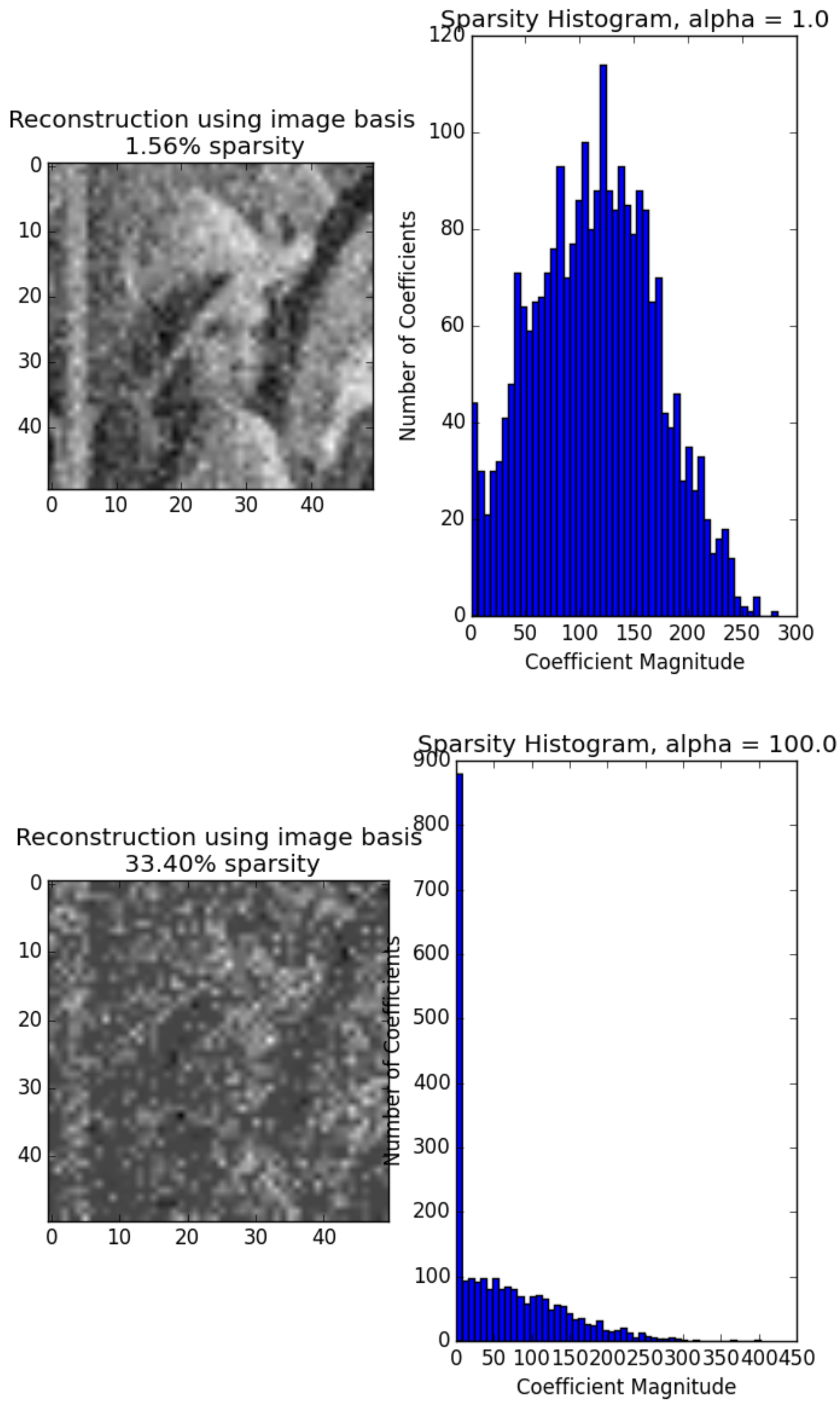

Fig. 3. Reconstruction results for sketching in the image domain. Starting from $\alpha = 1.0$ (which yields an inaccurate and non-sparse reconstruction), we increase α until we see significant sparsity. At $\alpha = 100.0$, we do see a reasonable compression ratio of just over 30%, but the reconstruction is utterly unrecognizable. The histograms confirm that the coefficients are not very well clustered around zero for low α.

Figure 3 shows the reconstruction results for varying α. In each case, A is composed of independent, identically distributed (IID) standard Gaussian random variables. Note that we calculate sparsity by taking the fraction of optimal coefficients that are less than 1% of the maximum coefficient in absolute value. Since we are using the L_1 norm, none of the optimal coefficients are ever *identically* zero, so this is a good measure of when a coefficient is "small."

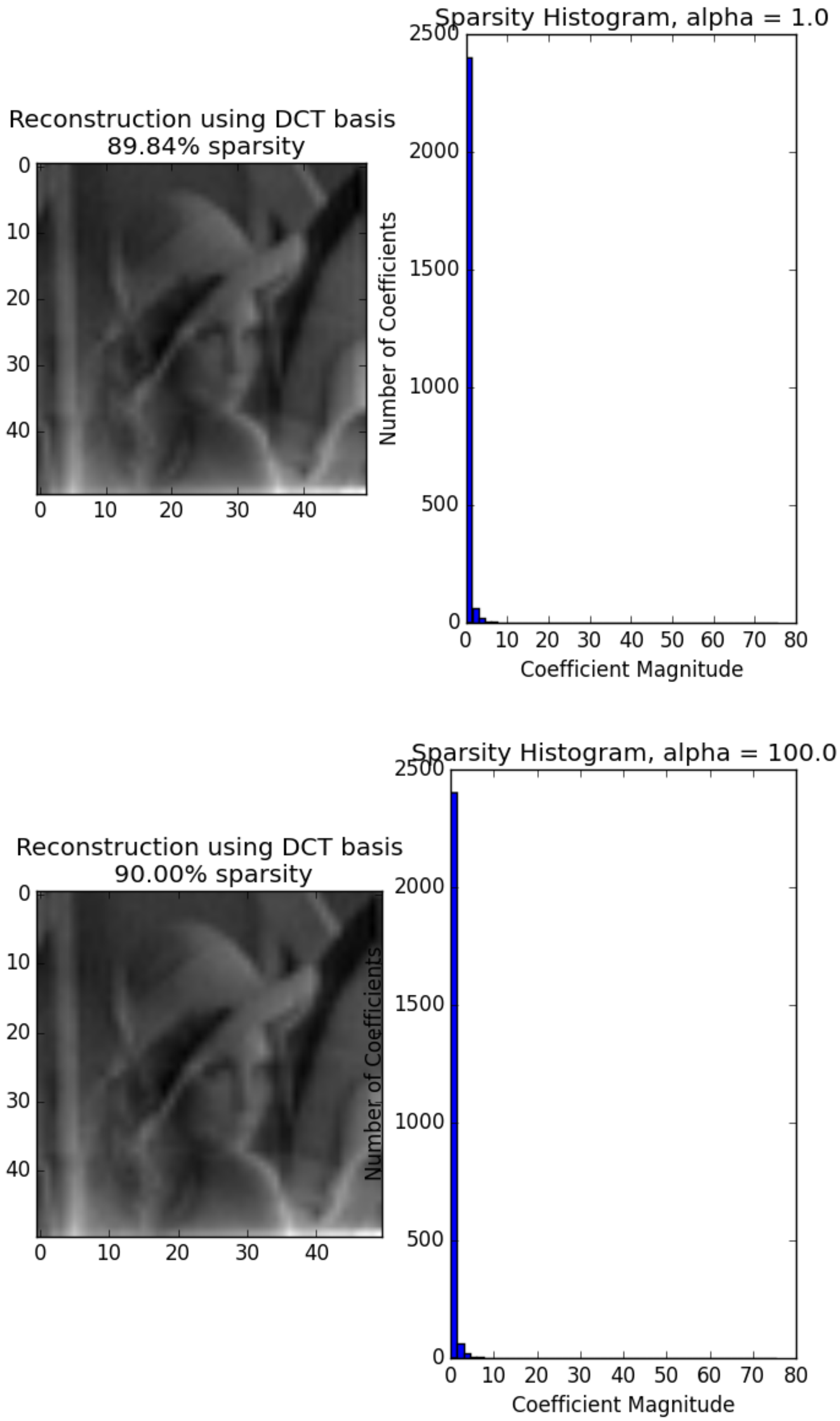

Fig. 4. Reconstruction results for sketching in the DCT domain. From top to bottom, we increase α, resulting in only marginally increased sparsity for $\alpha = 100.0$. Both reconstructions are completely recognizable at 90% sparsity – a significant improvement over sketching in the image domain.

D. Sketching in the DCT domain

Finally, we consider re-introducing a non-identity basis $\hat{\Psi}$ in which we suppose the image to be more sparse than the standard basis. In particular, we set $\hat{\Psi} = C$, where C is the DCT basis. The optimization problem becomes:

$$x^* = \arg\min_x \|ACx - m\|_2^2 + \alpha\|x\|_1 \tag{4}$$

Figure 4 shows reconstruction results for exactly the same values of α as in Figure 3. Additionally, we show the

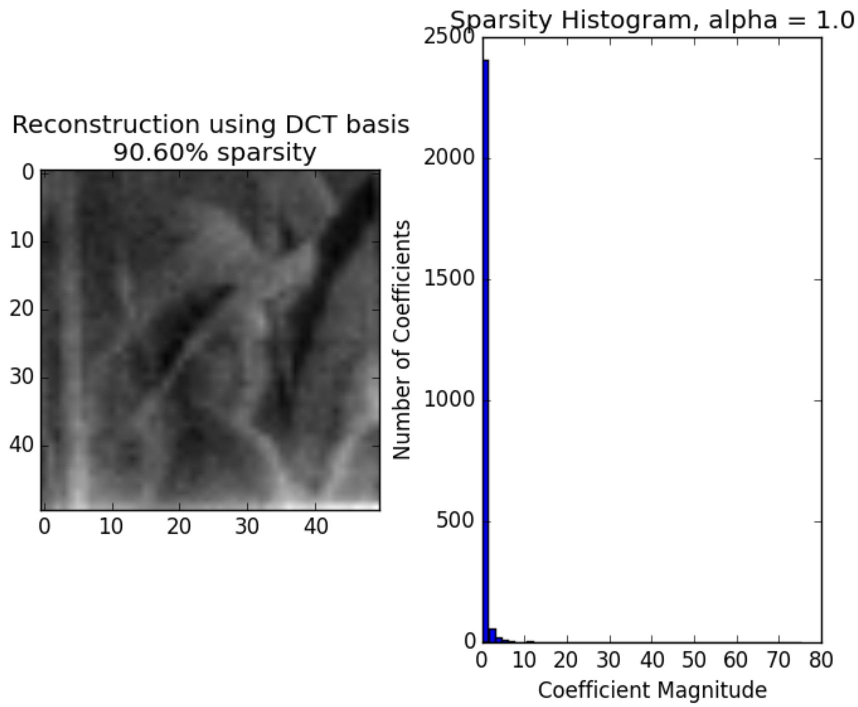

Fig. 5. Reconstruction results for sketching in the DCT domain. Here, we force A to have only 70% as many rows as it has columns (i.e. we simulate under-sampling the image). When A does not have full rank we do not observe a dramatic decrease in fidelity or increase in sparsity (unlike sketching in the image domain), but we also do not observe a significant change in sparsity.

reconstruction results when we decrease $\mathrm{rank}(A)$ such that A has only 70% as many rows as it has columns, in order to simulate undersampling (i.e. *compressed* sensing). These results are very encouraging – we can achieve a relatively good reconstruction at a compression ratio of just over 90%.

IV. Shortcomings

At this point, the most noticeable shortcoming in our approach is that we are forced to use tiny images. The reason is simple: when we solve either of equations (3) or (4) we must store the matrix A (and in the case of (4), C) in memory. For a 50x50 pixel image, a full-rank basis A has $50^2 = 2500$ rows and columns, which means it has $2500^2 = 6,250,000$ entries. Each entry is double-precision, which means the entire matrix occupies roughly $6,250,000 \cdot 8 = 50,000,000$ bytes, or 50 MB. Memory usage is thus quartic in image size, which means that a typical high definition 1920x1080 pixel image would require more than 34 TB of memory. Clearly, this is not feasible.

Using small images has drawbacks, though. Since there are fewer pixels in these downsized images, in general they are much more difficult to compress (i.e. they simply are not sparse).

As mentioned below, we plan to circumvent this memory issue by partitioning the image into small blocks and compressing each block independently. This way we will not need to artificially resize the image before compression. We expect to be able to achieve much greater compression ratios using this blocking scheme.

V. Future work

We plan to implement the following functionality in the coming weeks.

1) Try LASSO on vanilla DCT (e.g. replace “top k” with sparse LASSO solution – we expect this to give a worse result)
2) Re-write sketching stuff to be blockwise on the entire image (expect better compression ratios)
3) Build in oversampling with a random measurement matrix A. Apply this in image domain and DCT domain.
4) Replace the L_1 penalization with the reversed-Huber penalization.

References

[1] Anthony Griffin and Panagiotis Tsakalides, “Compressed Sensing of Audio Signals UsingMultipleSensors,” in Proc. 16th European Signal Processing Conference, Lausanne, Switzerland, pp. 1-5, Aug. 2008.
[2] Michael Lustig, David Donoho, and John M. Pauly, “Sparse MRI: The Application of Compressed Sensing for Rapid MR Imaging,” in Magnetic Resonance in Medicine, vol. 58, pp. 1182-1195, 2007.
[3] Emmanuel Candes and Michael Wakin, “An Introduction to Compressive Sampling,” in IEEE Sig. Proc. Mag., pp. 21-30, Mar. 2008.
[4] Yusuke Oike and Abbas El Gamal, “CMOS Image Sensor With Per-Column $\Sigma\Delta$ ADC and Programmable Compressed Sensing,” in IEEE Journal of Solid-State Circuits vol. 48, no. 1, pp. 318-328, Jan. 2013.
[5] S. Derin Babacan, Reto Ansorge, Martin Luessi, Rafael Molina, and Aggelos K. Katsaggelos, “Compressive Sensing of Light Fields,” in Proc. 16th IEEE Int. Conf. on Image Processing (ICIP), pp. 2337-2340, Nov. 2009.