Single Pixel Camera Written Report
ECEN 5012 Final Project

Parker Boyce* Mohammed Adib Oumer*
* University of Colorado, Boulder

***Abstract*—For this project we researched the models that make single pixel cameras (SPC) possible and wrote an implementation of a SPC in Matlab. The software implementation uses images taken with a digital camera to calculate measurements which a single pixel camera would make for the same scene and perform reconstruction, either by raster scan, basis scan or compressive sensing. We simulated images acquired through all these techniques. For compressive sensing we experimented further using different probability distribution functions for random sensing matrices, different compressive bases and varying numbers of samples.**

I. Introduction

Unlike the million pixel arrays used in our everyday electronics such as our phones, a single pixel camera uses a single pixel or sensor to produce images. Usually, we have a light source shone on an object and the light reflecting off the object into the pixel arrays of our camera. Based on the pattern and intensity of light reflected into the pixels, we can reconstruct the picture and apply sampling and compression after obtaining measurements. In contrast a single pixel camera collects the total light intensity reflected into its sensor and produces one singular measurement. Different mechanisms can be used to gather measurements with the spatial information necessary for image reconstruction. The three most common mechanisms are raster scan, basis scan and compressive sensing/sampling. Compressive sensing edges out the rest based on practical advantages it offers (discussed later). The single pixel camera is ordinarily constructed using a DMD (digital micromirror device) or an LCD matrix as a mask and a photosensor. A sample construction is shown in Figure 1 [7]. A single-pixel camera produces images by exploring the object with a "series of spatially resolved light field patterns made by a spatial light modulator (mask) and measuring the correlating light intensity on the detector without any spatial resolution" [7].

Some of the advantages of a single pixel camera are listed below:

(a) Video applications such as shutterless video imaging are possible by continuously sampling, and taking advantage of the spatial and temporal redundancies in compressing while imaging [5].

(b) We can detect a larger range of the electromagnetic spectrum (including UV and IR). This is attributed to two reasons: the DMD mirrors can reflect UV and IR wavelengths while silicon used in pixel sensors works only in visible light spectrum, and because we only use a single photosensor (and not millions of pixels), we can easily use specialized sensors made to work in UV and/or IR regions. [2].

(c) The photosensors are able to sense $\frac{N}{2}$ times more pixels than an average pixel sensor, meaning there is less distortion from dark noise [5].

In this project, we simulate a single pixel camera in Matlab, apply several scanning methods and discuss our results. Finally, we suggest further topics in this domain we wish we had time to explore.

II. Theory of operation

We simulate samples from a single pixel camera using images taken by a digital camera and already produced images. The DMD can be simulated by applying a mask to an image, dropping those pixels which are masked and then taking a global sum or root mean square of the remaining pixels. Even though this method of measuring light intensity is imperfect since the camera will make adjustments to prevent overexposing images, this global sum can be used as a stand-in for light intensity which would ordinarily be measured by a photo-detector. The mask is simulated using a vector ϕ_m. Image reconstruction can then be thought of as a linear program; each of the

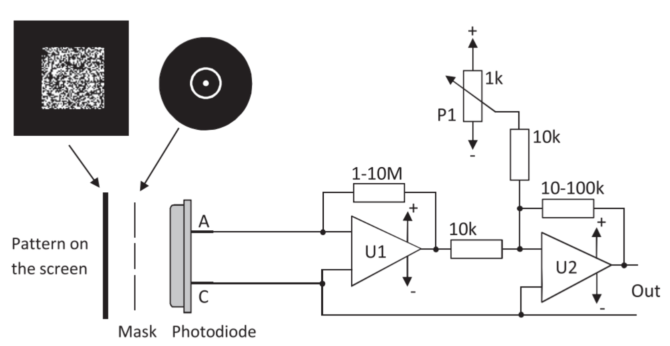

Figure 1. Sample SPC construction using LCD matrix and photodiode

masks used to take measurements with the SPC can be thought of as equations in a linear system, the solution to which is the original image. A raster scan is a trivial example of how this could work.

By putting several of these masks together, we get a sensing matrix Φ that will be applied to our image to obtains our measurements y. In order to do so, we vectorize the $\sqrt{N}\times\sqrt{N}$ (N pixels) image matrix into an $N\times 1$ vector x in $\mathbb{R}^{\mathbb{N}}$. One of the benefits of this representation is that any such vector can be represented in terms of a basis of N orthonormal $N\times 1$ vectors that when put together form an $N\times N$ basis matrix Ψ. What this means is that we can represent our vectorized image x as a product of our basis matrix Ψ and some coefficient vector α, i.e. $x = \Psi\alpha$. A nice property of Ψ is that $\Psi^T = \Psi^{-1}$ (and thus $\alpha = \Psi^T x$, inner product between each basis vector and our image. This is important in easing computational complexity). Also, because of the vectorization, our measurements can be expressed as $y = \Phi x = \Phi\Psi\alpha = A\alpha$. From this discussion, one must note that the only things we have access to in a single pixel camera image capture and reconstruction are the sensing matrix Φ, basis matrix Ψ and our measurements y, and NOT the coefficient vector α or equivalently our original image x. Below, we investigate applying several scan methods that apply different Ψ and Φ and the results from them.

A. Raster Scan

One could perform a raster scan where only one "pixel" of the mask is sampled at a time until all pixels are captured and then trivially reconstruct the image. Thus sampling and reconstruction are limited by the resolution of the mask, so it's useful to keep in mind how many samples are required for a complete raster scan as that is the upper limit on the number of samples required to produce an image. In simple terms, the photosensor takes N light measurements sequentially from each of the N pixels over the capture time. Because of this pixel-wise operation, raster scan can be intensive in real life testing and prone to noise. However, in simulation, it is simplified by choosing a $\Phi = \mathbb{I}$, the identity matrix. Here Φ will have dimension N and thus, our measurement $y = \Phi x = \mathbb{I}x = x$, is directly obtained from the vectorized image. Thus our reconstructed image $\hat{x} = y$. We can then reshape the vector into a $\sqrt{N}\times\sqrt{N}$ image to get our reconstructed image in 2D-form. Obviously, this is a very trivial process to simulate and gives the exact result as our original image. We can try to emulate real life conditions by adding noise after reconstructing the image but the variables in terms of what type of noise and what parameters should be used are subjective. We chose to leave our final image as is and not add noise, thereby not really being able to compare our result with results from compressive sensing in a realistic way (this choice was justified since we could try adding a lot of noise to make the raster scan look worse than compressive sensing, and vice versa). Note that, in this scanning method, our basis matrix Ψ is also $\mathbb{I}$, meaning our coefficient vector α is essentially our image as well.

B. Basis Scan

Here, the photosensor takes N light measurements sequentially from different combinations of the N pixels as determined by test functions ϕ_m that are not delta functions (as in raster scan) but from some more general basis [5]. Unlike raster scan, in this method, we want to be able to compress our image but still keep the same dimension of the image in our measurements. This is done by keeping $\Psi = \mathbb{I}$ but switch Φ to a basis that allows α (equivalently x) vector to result in compressible (lots of small coefficients and some large coefficients) or sparse (lots of zeros, some large coefficients) measurements. Two dominant practical examples of such bases are the DCT (discrete cosine transform) basis (we use DCT-II based on [7]) and the Hadamard basis. The DCT-II basis is used in JPEG, which shows that it indeed provides compressible measurements y. The Hadamard basis is a generalization of the common sparsifying transform, the DFT basis [1]. These two bases are popular because they can be represented in purely real form (DCT-II is real, Hadamard basis has a real form used in Matlab) and do not use complex form like the DFT-basis. They are thus more practical in a single pixel camera and require much less storage (for example, DFT generally requires twice the amount of storage used in DCT). Here we have $\Phi^T = \Phi^{-1}$. Given $y = \Phi x$, our reconstruction will then be $\hat{x} = \Phi^T y$. We do implement a sample of this discussed later.

C. Compressive Sensing

The most popular method of image reconstruction used in single pixel camera is the most recent development, compressive sensing. Here, the photosensor takes $M \ll N$ light measurements sequentially from different combinations of the N pixels [4], [8]. The popularity of compressive sensing stems from this. Practically for example, if you have a 10 MP (megapixel) image, you would have to do 10 million measurements to reconstruct your image in raster or basis scan. However, you are able to achieve a comparable result using much fewer measurements by applying compressive sensing. In our simulation, we make Ψ either the DCT-II basis or Hadamard basis (only to be used during reconstruction). Then, our Φ will be an $M\times N$ matrix. As discussed in class, we want Φ and Ψ to have less overlap with each other so that we can capture as much information about α as possible (this property is called "restricted isometry property"). Given that we are choosing Ψ to be a given basis, one effective choice for Φ that was shown both empirically and theoretically to be effective is to take a random matrix. This is achieved in a practical single pixel camera by programming the

LCD or DMD matrix to randomly reorient so that only M measurements are received by the photosensor [7]. Thus, we have $y = \Phi x$ where y is an $M \times 1$ vector. This results in an underdetermined system and we cannot reconstruct our image using the simple method ($\hat{x} = \Phi^T y$). One advantage of having an underdetermined system is that it is less prone to noise (which in a way adds to the denoising effect). Moreover, one requirement we have is that our coefficient vector α has to be K-sparse, meaning it should have K nonzero elements, where $K \leq M$ (therefore, being able to represent our image with as few samples as possible). Such an underdetermined problem with sparsity requirement is solved using a convex optimization algorithm. The general formulation is given by Equation 1 as an unconstrained optimization problem.

$$\min_{\alpha} \|A\alpha - y\|^2 + \lambda \|\alpha\|_p \tag{1}$$

where $A = \Phi\Psi$, and λ controls the trade-off between the data misfit error $A\alpha - y$ and the penalty in terms of the p-norm of the parameters [7]. The choice of p is important. p = 0 will lead to ℓ_0-minimization which leads to a non-convex problem so it is avoided. p = 1 leads to ℓ_1-minimization which leads to sparse solutions and therefore, is the value of choice. p = 2 leads to ℓ_2-minimization which is easily solved as a least squares problem but does not lead to sparse solutions so it is generally avoided [8]. By choosing ℓ_1-minimization, we formulate our problem as Basis Pursuit (BP) equality constrained optimization problem given in Equation 2. There are several solvers to solve BP problems. We used the SPGL1 library for Matlab. The optimization provides us with optimum α, and the reconstructed image is thus $\hat{x} = \Psi\alpha$. This image is the result we plot in the next section.

$$\min_{\alpha} \|\alpha\|_1 \text{ subject to } y = A\alpha \tag{2}$$

A major drawback of compressed sensing is solving a convex optimization problem which is computationally demanding, making compressed sensing a slow process. An image might take up to 5.25s to be captured and reconstructed [3], as opposed to a click on our phone cameras or however long it takes to take $N - M$ measurements in raster or basis scan (generally less than the time it takes to perform compressive sensing).

III. Results

(PS: Upon experimentation, we encountered that the computations were heavy on our laptops so we took several measures to get decent outputs in a reasonable amount of time. We mostly used the images provided in Assignment 4)

We used Matlab to simulate the process of reconstructing images via compressive sensing, using M samples taken with a single pixel camera. Since the reconstruction is not perfect unless $M = N$ the resulting images are approximations of the originals. We experimented with generating Φ via the standard normal distribution and the Bernoulli $+1$, -1 probability distribution (with parameter 0.5) and Ψ was either the DCT-II basis or the Hadamard basis. All our test images were resized to 64x64 pixels, except for a self portrait taken with a web camera which was resized to 100x100 pixels. Unfortunately we were unable to experiment with larger images because Matlab could not complete basis pursuit without error or in reasonable amount of time. There is still enough detail in the images to reveal differences in the approximations produced by the two choices of Φ and Ψ.

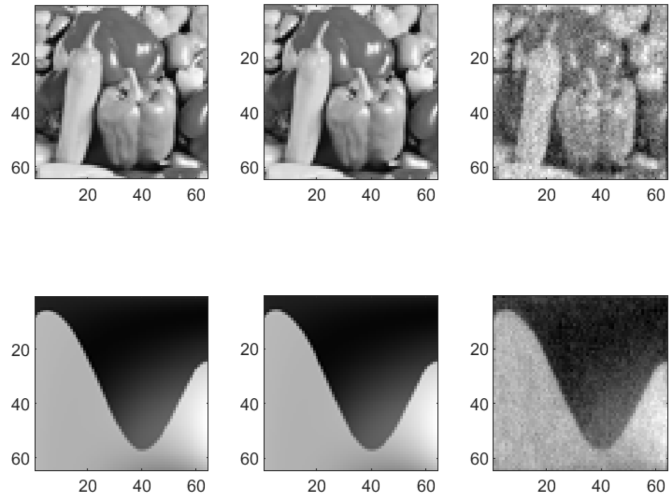

Figure 2. Raster scan (left), Basis scan (middle) and Compressive sensing (right) results. "peppers" top row, "c2c2" bottom row.

Figure 2 shows the resulting images on "peppers" and "c2c2" from raster scan (which is essentially the original image as described in the section above), basis scan using the DCT-II basis and compressive sensing using Φ as normal distribution and Ψ as DCT-II basis. $M = 0.5N$ where $N = 4096$ for compressive sensing. For comparison sake, the running times were 0.1-0.2s, 0.25-0.35s and 15-25s, the MSE errors of reconstruction were 0 (as expected), 1.7×10^{-15} and 0.58 (for "peppers") , 0.05 (for "c2c2"), and the PSNRs were ∞, 147dB and 2.4dB (for "peppers") , 13dB (for "c2c2") from raster scan, basis scan and compressive sensing respectively. Based on the qualitative results, we can only conclude that compressive sensing indeed results in a decent reconstruction, considering we only used half as many measurements. We assume the PSNR is relatively low for "peppers" because it is nuanced and has more features.

The test images we chose provide differing amounts of detail, for instance "smoothIm" is the simplest test image and can be produced by plotting a bivariate normal distribution. Unsurprisingly it has the highest level of sparsity and requires the smallest M to produce a legible approximation. When using the Gaussian sensing

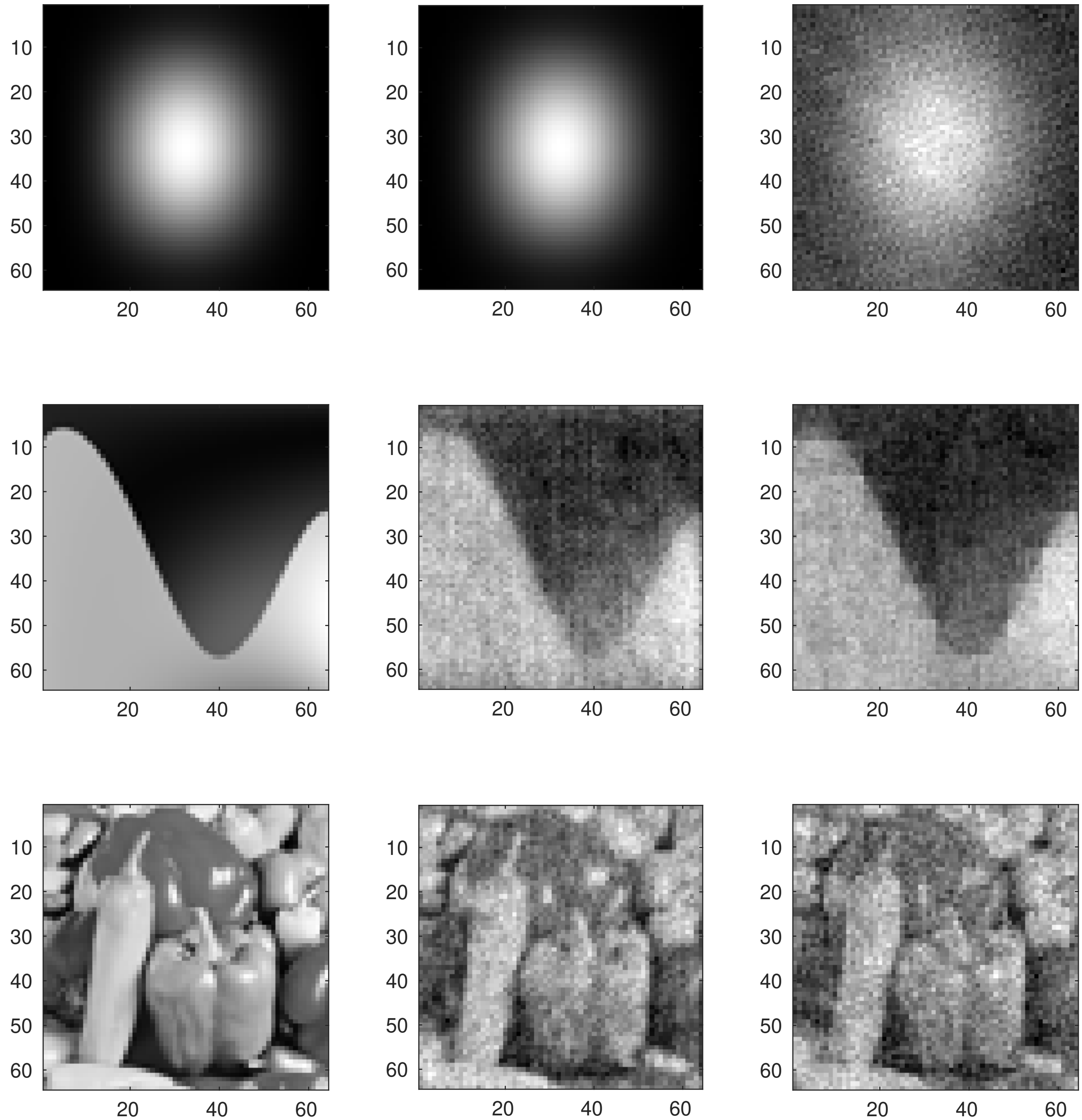

Figure 3. Images reconstructed using compressive sensing with a simulated single pixel camera. The left column shows the original images. The top row is "smoothIm", the middle row is "c2c2", and the bottom row is "peppers". $M = 200$ for smoothIm, $M = 1000$ for c2c2 and $M = 2000$ for peppers. Φ is Gaussian for the top middle image and Bernoulli for the top right image. Ψ is the Hadamard matrix for the middle right and bottom right images. For all other images Ψ is the DCT-II matrix.

matrix for Φ it's difficult to see any noise introduced by compressive sensing. The noise introduced by compressive sensing follows the same probability distribution as the elements of Φ (given that the elements of Φ are independent and identically distributed). Since the sum of normal distributions is also normal this can explain the apparent lack of noise in the reconstruction of smoothIm. When using the Bernoulli distribution for Φ the image reconstruction is noisier. These results are apparent when comparing the middle and right images in the top row of figure 3. A self portrait was taken with a web camera to provide a more complex test image for reference and experimentation. The reconstruction adds noise and removes some detail but the scene is still remarkably legible considering that $M = \frac{N}{2}$

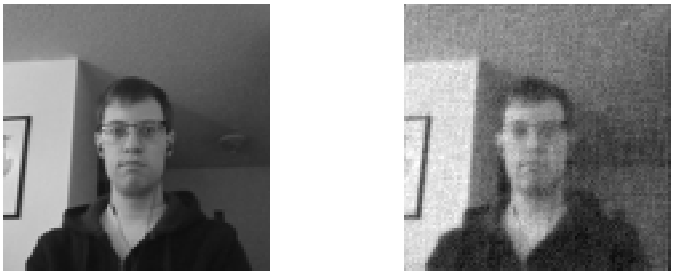

Figure 4. 100x100 pixel self portrait taken with a web cam. The original is on the left and a reconstruction performed with compressive sensing is on the right. $N = 10000$, $M = 5000$, Φ is Gaussian random and Ψ is the DCT-II basis.

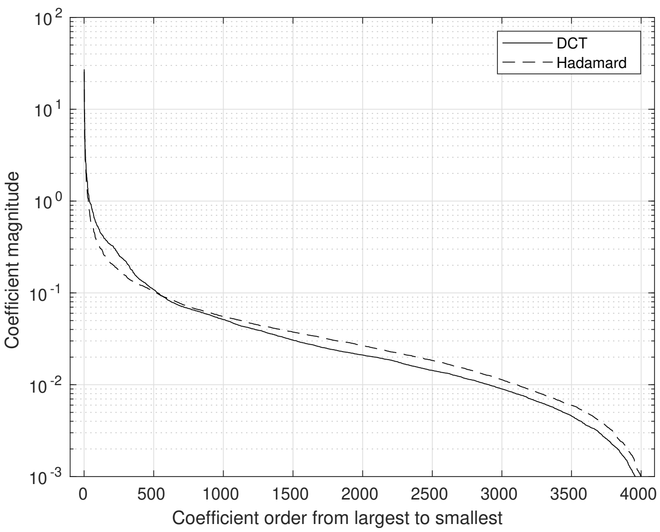

Figure 5. Log scale plot of magnitude of coefficients of c2c2 in the DCT-II and Hadamard bases, sorted from largest to smallest.

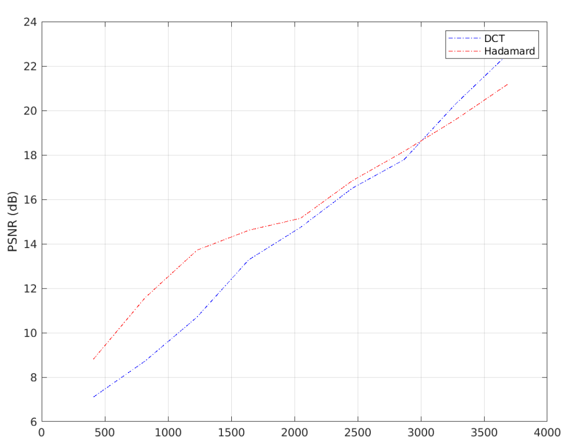

Figure 6. Plot of PSNR for reconstruction of c2c2 test image using Hadamard basis and DCT-II basis.

in figure 4.

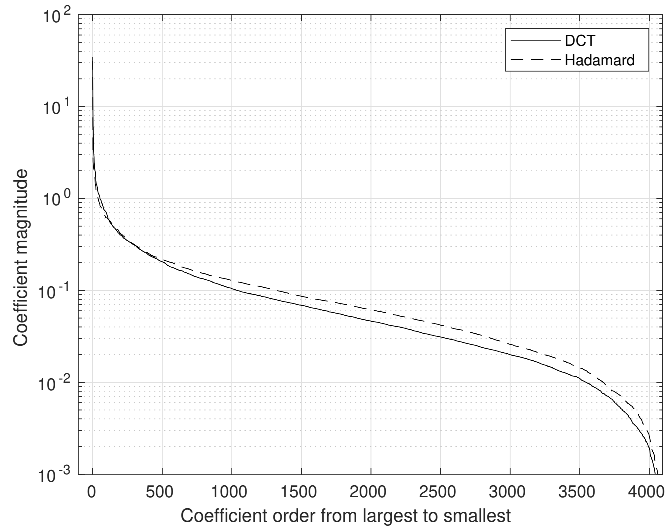

Figure 7. Log scale plot of magnitude of coefficients of "peppers" test image in the DCT-II and Hadamard bases, sorted from largest to smallest.

c2c2 was chosen to illustrate the visual differences in reconstruction when using the DCT-II matrix vs. the Hadamard matrix for Ψ. The reconstruction with the Hadamard basis produces an approximation with blocky, pixelated artifacts. Figure 5 shows the sparsity of c2c2 in the DCT-II basis and the Hadamard basis. If we take our tolerance to be about 0.1, then we get a sparse coefficient vector with $K = 500$. The number of large coefficients appears to be the same for both bases, although the rate at which the coefficients decrease in size is larger for the DCT-II coefficients than the Hadamard coefficients. The reconstruction process was performed for various values of M with both the Hadamard and DCT-II bases to compare their performance, the results of which are shown in figure 6. In this test the Hadamard basis performs slightly better than the DCT-II basis for values of M less than 2000, otherwise the two bases perform similarly.

The "peppers" test image shows how the legibility of the image is impacted during reconstruction. With $M = 2000$ only the peppers in the foreground clearly appear to be peppers, the background could be confused for rocks or a textured wall. The reconstruction with the Hadamard basis appears to be more grainy in this instance, and specular highlights are also not as visible as they are in the reconstruction with the DCT-II basis. Figure 7 shows the sparsity of peppers in the DCT-II basis and the Hadamard basis. If we take our tolerance to be about 0.1, then we get a sparse coefficient vector with $K = 1000$.

IV. Future Work

There are other noise-aware variants of BP such as Basis Pursuit Denoising (BPDN), LASSO and Quadratically Constrained Basis Pursuit (QCBP). They all give more

or less similar results, however we were not able to test these variants and would have to be included in future work. Another scanning method we did not consider was Computational Ghost Imaging (CGI) discussed in [6], [7], [8]. It looks interesting to explore more.

Most of the work we read focused on grayscale images as they are easier to test out and simulate. In principle a single pixel camera could take color pictures with the help of filters or the use of multiple sensors, so future work could be done on RGB images as well.

Another interesting exploration would be to test out how a single pixel camera will work with video imaging. There are theoretical papers that discussed this such as [5] but one can test it out in experiment as well.

More work could be done in exploring different bases for Ψ and comparing which bases provide empirically better results.

(One thing to note is that we are able to reconstruct our signal under several assumptions using sub-Nyquist sampling but this does not necessarily imply that we are contradicting Nyquist-Shannon sampling theorem. The theorem only guarantees a condition for perfect reconstruction but does not inhibit reconstruction attempts from sub-Nyquist sampling.)

References

[1] "Hadamard matrices and orthonormal bases," https://orion.math.iastate.edu/esweber/cbms2018/bootcamp/1-hadamard.pdf, (Accessed on 12/15/2021).

[2] "Products - inview technology corporation," https://web.archive.org/web/20130331045434/http://inviewcorp.com/products/, (Accessed on 12/15/2021).

[3] "The purpose of a single pixel camera," https://www.vocal.com/compressed-sensing/single-pixel-camera/, (Accessed on 12/15/2021).

[4] R. G. Baraniuk, "Compressive sensing [lecture notes]," *IEEE signal processing magazine*, vol. 24, no. 4, pp. 118–121, 2007.

[5] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 2008.

[6] G. M. Gibson, S. D. Johnson, and M. J. Padgett, "Single-pixel imaging 12 years on: a review," *Opt. Express*, vol. 28, no. 19, pp. 28 190–28 208, Sep 2020. [Online]. Available: http://www.osapublishing.org/oe/abstract.cfm?URI=oe-28-19-28190

[7] T. A. Kuusela, "Single-pixel camera," *American Journal of Physics*, vol. 87, no. 10, pp. 846–850, 2019. [Online]. Available: https://doi.org/10.1119/1.5122745

[8] D. Mackenzie, "Compressed sensing makes every pixel count," 2009.