Compressive Deconvolution Algorithms for a Computational Lightfield Display for Correcting Visual Aberrations

Anmol Parande

Electrical Engineering and Computer Sciences
University of California, Berkeley

May 13, 2022

Acknowledgement

Compressive Deconvolution Algorithms for a Computational Lightfield Display for Correcting Visual Aberrations

by Anmol Parande

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Brian Barsky
Research Advisor

13 May 2022

(Date)

* * * * * * *

Professor Michael Lustig
Second Reader

05/09/22

(Date)

Compressive Deconvolution Algorithms for a Computational Lightfield Display for
Correcting Visual Aberrations

Abstract

Compressive Deconvolution Algorithms for a Computational Lightfield Display for
Correcting Visual Aberrations

by

Anmol Parande

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Brian Barsky, Chair

Millions of people in the world are afflicted by visual aberrations. Vision correcting displays
help accomodate those with visual aberrations by determining a new image to present the
user such that they will see it in focus. This work builds upon previous vision correcting
display research by developing and implementing theory that bridges the tradeoff between
accuracy and speed that past implementations have faced. In particular, we explore how
ideas from compressed sensing can be used to solve the vision correction problem by lever-
aging sparsity and nonlinear reconstruction techniques. We first model the vision correction
problem as a compressive deconvolution problem. We then provide a proof-of-concept imple-
mentation which validates the efficacy of the theory for a variety of blur strengths. Next, we
propose new techniques for modeling the relationship between the eye and the user's display.
Finally, we suggest future directions which can take the proof-of-concept developed in this
work and turn it into a practical application for people to use.

To my parents, Rajendra and Sangita Parande, my sister, Anjali Parande, and my girlfriend, Shubha Jagannatha

Thank you all for supporting me.

Contents

List of Figures

List of Tables

Acknowledgments

Chapter 1

Introduction

It is estimated that 61% of people in the United States need some sort of vision correction [1]. Without correction, they suffer from blurry vision. Typically, corrective lenses like eyeglasses or contact lenses are the go-to solution, but they can be a hassle to put on and remove. Moreover, now that people are increasingly interfacing with digital media, if the devices at which they are looking could take into account their eye condition, then that would eliminate the need to wear corrective lenses while looking at computers, phones, and tablets. These "Vision Correcting Displays"(VCD) would make it possible for people to see in focus even if they don't have access to corrective lenses. This work explores how we can write efficient algorithms for building such displays.

1.1   Background

The human eye is a complex optical system (shown in fig. 1.1[2]). Light first hits the cornea, which is a high powered lens with a fixed focal length. The cornea refracts light through the pupil of the eye. Light then travels through the lens, which adaptively changes its focal length in order to refract the light onto the retina, where rod and cone cells convert the incident light into electrical signals which the brain then perceives as an image. The **near point distance** is the closest point at which the eye can focus, while the **far point distance** is the furthest point at which the eye can focus. The planes at each of these distances are sometimes referred to as the focal plane depending on the context (fig. 1.2). Unfortunately, many people have aberrations in their eye which causes light to not focus on the retina. The most common conditions are myopia, presbyopia, and hyperopia where the focal length of the cornea and lens does not match the length of the eye. In myopia (near-sightedness), the far-point, which is typically considered to be at ∞, is instead finite. In hyperopia and presbyopia (farsightedness), the near-point is further from the eye than it would be for someone with normal vision. These types of "lower-order" aberrations are

---

[1]https://www.mesvision.com/includes/pdf_Broker/MESVision%20Facts%20and%20Statistics.pdf
[2]https://www.specialtyeyeinstitute.com/services/retina-care/eye-anatomy/
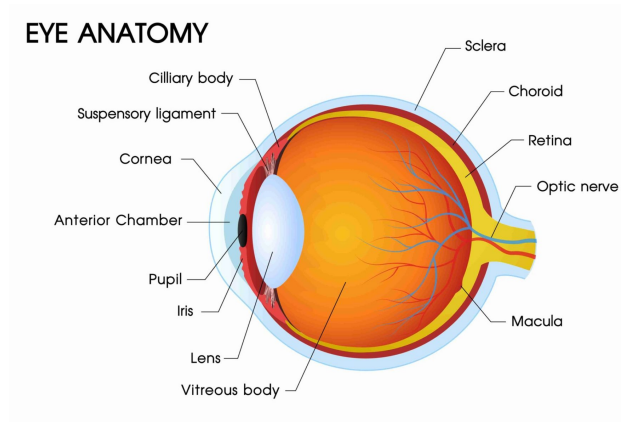
Figure 1.1: Diagram of the Eye
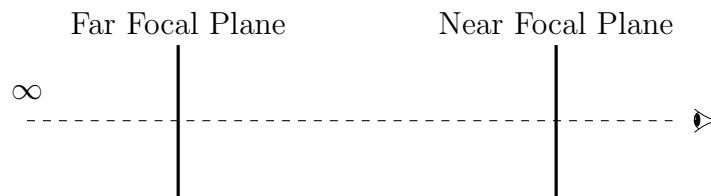
Figure 1.2: Near point and Far Point

currently fixed using corrective lenses. However, there are also "higher-order" aberrations which cannot be corrected by lenses that are caused by structural aberrations in the lens and cornea. For both high and low order aberrations, a vision correcting display is one which can account for a user's visual aberrations and display an focused image without the aid of a user's corrective lenses.

1.2 Problem Description

The primary problem addressed in this work can be summarized as follows. *Given measurements of an individual's visual aberrations, can we design a display which presents in-focus images to the individual*? To view such a display, the individual would not need any corrective lenses such as eyeglasses or contact lenses. The problem itself is an "inverse" problem where we must determine the input **x** which produces a desired output **y** when put through an optical system: the person's eye. Solving the problem will require modifications to the physical display as well as algorithms which can efficiently solve the inverse problem. This work demonstrates how a pinhole mask placed on top of a user's device in conjunction with a

compressive deconvolution algorithm can effectively solve the inverse problem and present an in-focus image. We will focus on lower-order aberrations, specifically presbyopya/hyperopia, since farsighted individuals have a difficult time looking at their mobile phones, making this a very common use case.

Chapter 2

Related Work

In [11], Huang attempts to solve the vision correction problem using classical image processing techniques without any display modification. In particular, he demonstrates results using space-domain and frequency-domain deconvolution techniques, including Richardson-Lucy and the Wiener filter. These approaches ultimately caused large ringing artifacts in the recovered images due to zeros in the magnitude transfer function (MTF) of the eye. To get around this issue, Huang proposed changing the display from a conventional display to a *lightfield display*.

2.1 Lightfield Displays

In a conventional display, each pixel can only emit a single color, no matter the direction it is viewed at. In contrast, a lightfield display is one where each pixel emits a different color depending on the angle at which it is viewed [12]. In practice, these are built by placing either a pinhole array or a microlens array over a high resolution screen. To avoid confusion, we will use "display" to refer to the lightfield display and "screen" to refer to the pixels on the high resolution screen, which could be a smartphone or laptop. Each pixel on the screen has pitch p, which is its width. In the case of the pinhole mask, the pinholes are separated by a distance Δx, and the width of each pinhole is identical to the pitch of the display pixels. The pinhole mask is placed a depth f_l away from the display. Huang et. al showed in [12] that a lightfield display setup could be used to effectively solve the vision correction problem.

2.2 Previous Algorithms for Vision Correction

Lightfield Projection

One of the earliest attempts to create a computational light field display was in 2014 from Fu Chung Huang et. al [12]. Huang et. al. set up and solved an optimization problem which computed the lightfield that needed to be produced by the display. They then generated a

prefiltered image by compositing the lightfield. Formally, Huang et. al created a matrix P, known as the lightfield projection matrix, where if $\mathbf{i}$ is a vector representing the discretized image on the retina and $\mathbf{l_d}$ is a vector representing the discretized lightfield on the display, then

$$\mathbf{i} = P\mathbf{l_d}. \tag{2.1}$$

Using P from eq. (2.1), Huang et. al computed the display lightfield that will correct the user's vision by letting $\mathbf{i}$ be the image and solving the optimization problem

$$\mathbf{l_d} = \underset{\mathbf{l}}{\operatorname{argmin}} \|P\mathbf{l} - \mathbf{i}\|_2^2 + \lambda\|\mathbf{l}\|_2^2. \tag{2.2}$$

The method of constructing P is described further in appendix A.

Forward Optimization

In 2016, the Forward Optimization algorithm was introduced by [17]. It followed the same general idea as Huang's original algorithm in [12] in that it attempted to solve the problem by constructing a matrix which modeled the relationship between the display and the retina. However, the major change made by the Forward Optimization algorithm is that it worked directly with the pixels on the display rather than the lightfield produced by the display-pinhole combination. More specifically, [17] constructed a matrix P such that $\mathbf{y} = P\mathbf{x}$ where $\mathbf{x}$ is the image on the display and $\mathbf{y}$ is the image on the retina. Given the target image $\mathbf{y}$, the prefiltered image $\mathbf{x}$ can be computed by solving the optimization problem

$$\mathbf{x} = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{y} - P\mathbf{x}\|_2. \tag{2.3}$$

Equation (2.3) has closed form solution

$$\mathbf{x} = (P^T P)^{-1} P^T \mathbf{y},$$

but in practice, it is iteratively solved using L-BFGS-B due to the large size of P as well as its ill-conditioning. The matrix P is constructed by tracing rays from the display to the retina and marking an entry in the row and column of P which correspond to the sensor pixel and display pixel hit by the ray respectively. The full algorithm for constructing the matrix is presented in appendix B. One important assumption that is required to construct this matrix is the **One-to-One Assumption**, which says that each pixel is only visible through a single pinhole. [18] presents the conditions underwhich this assumption is valid.

Ray-Tracing Algorithms

After the Lightfield Projection and Forward Optimization algorithms, the next broad class of algorithms used to solve the vision correction problem trace rays between the screen pixels and retinal pixels. They use these rays to determine how to assign values to each screen

	Lightfield Projection	Forward Optimization	Point to Point	Area to Area
Time	$168\,\mathrm{s}$	$17.639\,\mathrm{s}$	$0.024\,\mathrm{s}$	$0.131\,\mathrm{s}$
PSNR	$43.9\,\mathrm{dB}$	$52.0\,\mathrm{dB}$	$27.9\,\mathrm{dB}$	$20.3\,\mathrm{dB}$

Table 2.1: Comparison of Previous Work

pixel. These algorithms perform significantly worse than the Forward Optimization and Lightfield Projection methods, but they are significantly faster. Table 2.1 shows results for a 128×128 pixel image at $200\,\mathrm{cm}$ from a presbyopic eye with near point distance $500\,\mathrm{cm}$ and focal length $20\,\mathrm{cm}$.

Point-to-Point Algorithm (a.k.a Forward Algorithm)

In the Point-to-Point algorithm, introduced by Shichao Yue, rays are traced from screen pixels to the retina. Like in the Forward Optimization algorithm, one ray is sampled for each screen pixel. The ray is first drawn from the screen pixel to the closest pinhole. It is then traced to the retina using the same ray-tracing equations as the forward optimization matrix (see eq. (B.2) in appendix B). The color assigned to the screen is a bilinear interpolation of the neigbors of the retinal pixels that the ray hits. Like the Forward Optimization matrix, this approach makes the one-to-one assumption.

Area-to-Area Algorithm

In the Area-to-Area algorithm, introduced in 2019 by Yirong Zhen and Sibo Dong [18], both screen pixels and retinal pixels are considered as areas instead of points. For a given screen pixel, rays are traced from each corner of the screen pixel to each corner of the pinhole. These rays are then traced to the retina, where they form an area. The color of the screen pixel is the result of averaging the colors of all retinal pixels that fall within the covered area. Like the Forward Optimization and Point-To-Point methods, this approach makes the one-to-one assumption.

Chapter 3

Light Transport Analysis

One way to model the blur induced by the eye on an image is to consider how it changes the lightfield emanating from the display. A lightfield models the radiance of light rays emanating from each spatio-angular location. In particular, consider a "reference" plane X and an angular plane U which is parallel to, and one unit away from, the reference plane X. The radiance of a lightray emanating from spatial location $\mathbf{x}$ on the reference plane and hitting $\mathbf{u}$ on the angular plane is given by the lightfield $L(\mathbf{x}, \mathbf{u})$. This is known as the "two-plane" parameterization of lightfields [13] and is shown in fig. 3.1. As Liang et.al showed in

Two-Plane
Parameterization

In-Camera
Parameterization
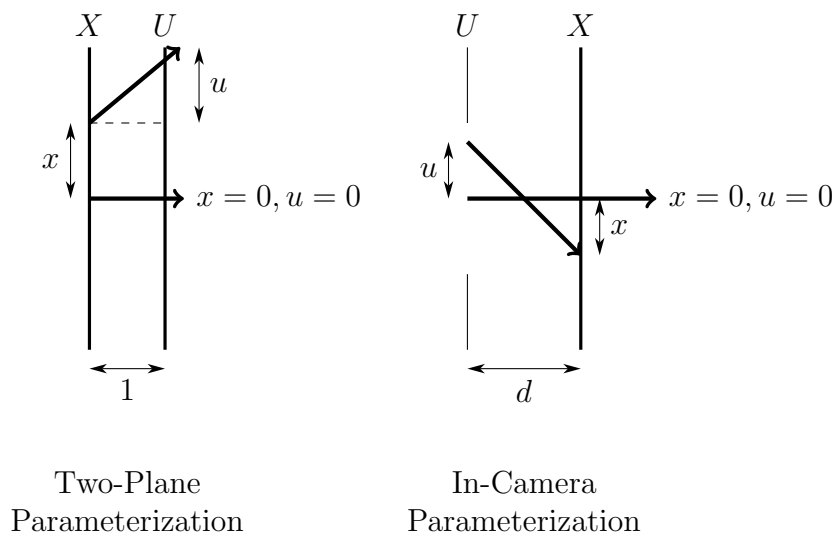
Figure 3.1: Lightfield Parameterizations

[13], this parameterization makes it easy to express how the lightfield changes under common optical operations. This helps use lightfields to create a unified imaging framework, which Liang et. al call *Light Transport Analysis* in [13]. Let $\mathbf{x}$ be a vector representing the location

the light ray originates from on the reference plane, $\mathbf{u}$ represent the location on the angular plane that the light rays intersects, and $\mathbf{z} = \begin{bmatrix} \mathbf{x} & \mathbf{u} \end{bmatrix}^\top$ be a concatenation of the two. Then under this framework, there are four lightfield operations.

1. **Propagation:** The lightfield L_2 at a plane a distance d from the original reference plane is a sheared version of the original lightfield L_1 where $L_2(\mathbf{z}) = L_1\left(T(d)^{-1}\mathbf{z}\right)$.

2. **Lens Refraction:** The lightfield L_2 after being refracted by a lens with focal length f is a sheared version of the original lightfield L_1 where $L_2(\mathbf{z}) = L_1\left(R(f)^{-1}\mathbf{z}\right)$.

3. **Reparameterization:** Once light passes the aperture of the imaging system, it can be convenient to put the angular plane on the aperture and the reference plane on the sensor which is a distance d away from the aperture. This is known as the "in-camera representation" in [13] and is depicted in fig. 3.1. The reparameterization is achieved by shearing the original lightfield L_1 where $L_2(\mathbf{z}) = L_1\left(Q(d)^{-1}\mathbf{z}\right)$.

4. **Occlusion:** In an imaging system, there can be occluders such as an aperture. These are modeled by a function $A(\mathbf{z})$, and they change the lightfield through multiplication where the new lightfield L_2 is related to the old lightfield by the relation $L_2(\mathbf{z}) = L_1(\mathbf{z})A(\mathbf{z})$.

The matrices for propagation, lens refraction, and reparameterization in the 1D setting are given in table 3.1. These same matrices can be adapted for a 2D setting by treating each axis (horizontal and vertical) as independent.

Propagation	Refraction	Re-parameterization
$T(d) = \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$	$R(f) = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix}$	$Q(d) = \begin{bmatrix} 1 & 0 \\ 1 & -d \end{bmatrix}$

Table 3.1: Light Transport Matrices in 1D

In the vision correction problem, light moves in three stages as depicted in fig. 3.2.

1. Light propagates a distance d_o from the display to the eye.

2. The light at the eye is refracted by a lens with focal length f and occluded by a circular aperture with radius r.

3. The light at the lens travels a distance d_i to the retina.

It is important to note that this is a slightly simplified eye model. In reality, the eye has two elements which refract light: the cornea and the lens. The aperture, which is the pupil,
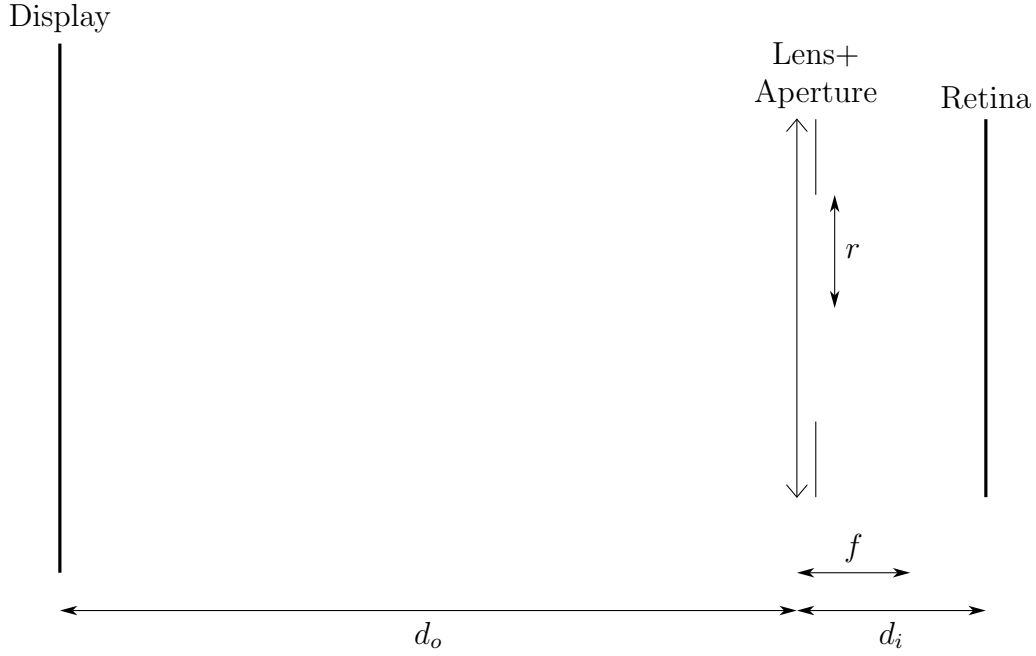
Figure 3.2: Imaging System Model

is in between the lens and cornea. We choose to model the cornea, lens, and pupil as a single lens with an aperture because the distances between all three elements are small, so modeling them as a single element does not lose much information. Under this model, the lightfield at the retina $L_r(\mathbf{z})$ is related to the lightfield produced by the display through a coordinate transformation M such that $L_r(\mathbf{z}) = L_d(M^{-1}\mathbf{z})$ when we neglect the aperture. In one dimension, the coordinate transformation M is given by

$$M = Q(d_i)T(d_i)R(f)T(d_0) = \begin{bmatrix} 1 & 0 \\ 1 & -d_i \end{bmatrix} \begin{bmatrix} 1 & d_i \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \begin{bmatrix} 1 & d_o \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & d_i \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & d_o \\ -\frac{1}{f} & 1 - \frac{d_o}{f} \end{bmatrix}$$

$$= \begin{bmatrix} 1 - \frac{d_i}{f} & d_o d_i \Delta \\ 1 & d_o \end{bmatrix}$$

where $\Delta = \frac{1}{d_i} + \frac{1}{d_o} - \frac{1}{f}$. Δ is a measure of how much defocus there is in the system. If d_o, d_i, and f satisfy the gaussian geometric ray tracing equation

$$\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f}, \tag{3.1}$$

then $\Delta = 0$ and the lightfield will be sheared only in the angular direction. We can extend this analysis to 2D by assuming that the $x - u$ and $y - v$ spatio-angular directions are

separable and can be treated independently. This breaks down in cases where the user has aberrations which correlate the two directions, but works for defocus (myopia, presbyopia, hyperopia) and astigmatism, the most common forms of aberration.

3.1 Shape of the Continuous Lightfield from the VCD

On a vision correcting display, the lightfield is generated by placing a pinhole mask over the pixels on the screen. Define the macropixel as the area on the pinhole mask which corresponds to a single pinhole. If a screen pixel has width p, then a macropixel has width $p_d = ap$ where a is the number of screen pixels underneath each macropixel. Figure 3.3
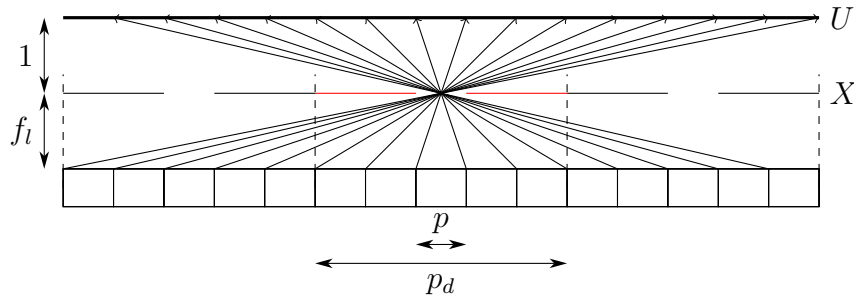
Figure 3.3: Angular Boundaries of Central Macropixel on Lightfield Display

shows the light rays emanating from the edges of each screen pixel for a target macropixel (highlighted in red). The intersection of these rays with the U plane defines the angles of the macropixel each screen pixel occupies. Since the radiance is constant over a screen pixel, it follows that the radiance will be constant over each angular region defined by the angular boundaries. Mathematically, the lightfield on the display is piecewise-continuous in space and angle. We will assume that the lightfield is nonzero for all spatial locations across a macropixel. In reality, the lightfield is only non-zero over the pinhole width because our pinhole mask setup only approximates an "ideal" light field display, so this assumption is technically not true. However, making this assumption is akin to assuming we have an ideal lightfield display, so it can be thought of as an "engineering approximation".

Consider a single piecewise segment of the continuous display lightfield. For ease, we'll look at the 1D case. The four corners of the segment are the coordinates $a = (0,0), b = (0, a_d), c = (p_d, 0), d = (p_d, a_d)$ where p_d is the width of a macropixel and a_d is the width of

an angular segment. When transported to the retina, these points will be transformed to

$$a' = M \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \qquad b' = M \begin{bmatrix} 0 \\ a_d \end{bmatrix} = \begin{bmatrix} d_0 d_i \Delta \\ d_o \end{bmatrix} a_d,$$

$$c' = M \begin{bmatrix} p_d \\ 0 \end{bmatrix} = \begin{bmatrix} 1 - \frac{d_i}{f} \\ 1 \end{bmatrix} p_d, \qquad d' = M \begin{bmatrix} p_d \\ a_d \end{bmatrix} = \begin{bmatrix} \left(1 - \frac{d_i}{f}\right) p_d + d_o d_i \Delta a_d \\ p_d + d_o a_d \end{bmatrix}.$$

Figure 3.4 shows the results of this transformation graphically. Note that in both the blurred
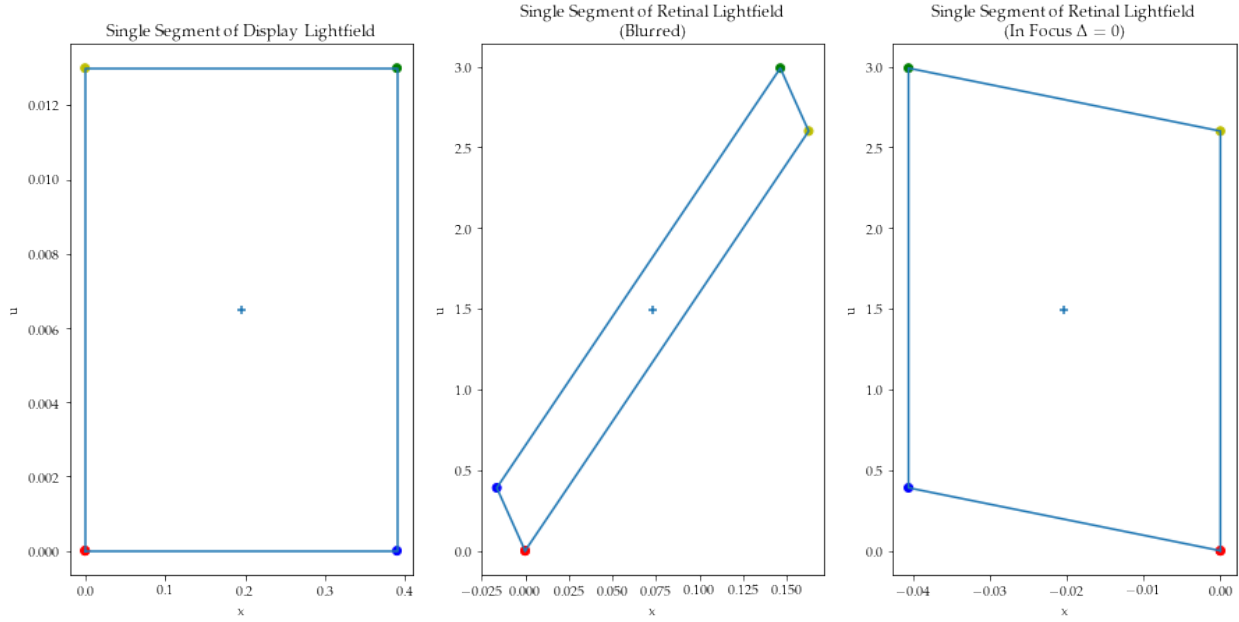
Figure 3.4: Segment of Piecewise Lightfield Under Different Levels of Blur

and unblurred, case, the lightfield is sheared in the angular direction by the same amount.

3.2   Image Formation

The lightfield at the retina is closely related to the image on the retina, which is what make lightfields a useful model for vision correction. Suppose the image on the retina is given by the function $I(\mathbf{x})$. This image is a projection of the 4D lightfield on the retina [13].

$$I(\mathbf{x}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} L_r(\mathbf{z}) du dv \tag{3.2}$$

Since $L_r(\mathbf{z}) = L_d(M^{-1}\mathbf{z})A(\mathbf{u})$, eq. (3.2) becomes

$$I(\mathbf{x}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} L_d(M^{-1}\mathbf{z})A(\mathbf{u}) du dv. \tag{3.3}$$

When $\Delta \neq 0$ and there is defocus blur, the lightfield at the retina is sheared, so integrating over the angles will create blur in the image. The intuition for how this works is best understood in one dimension and is depicted in fig. 3.5. When we place a gaussian which
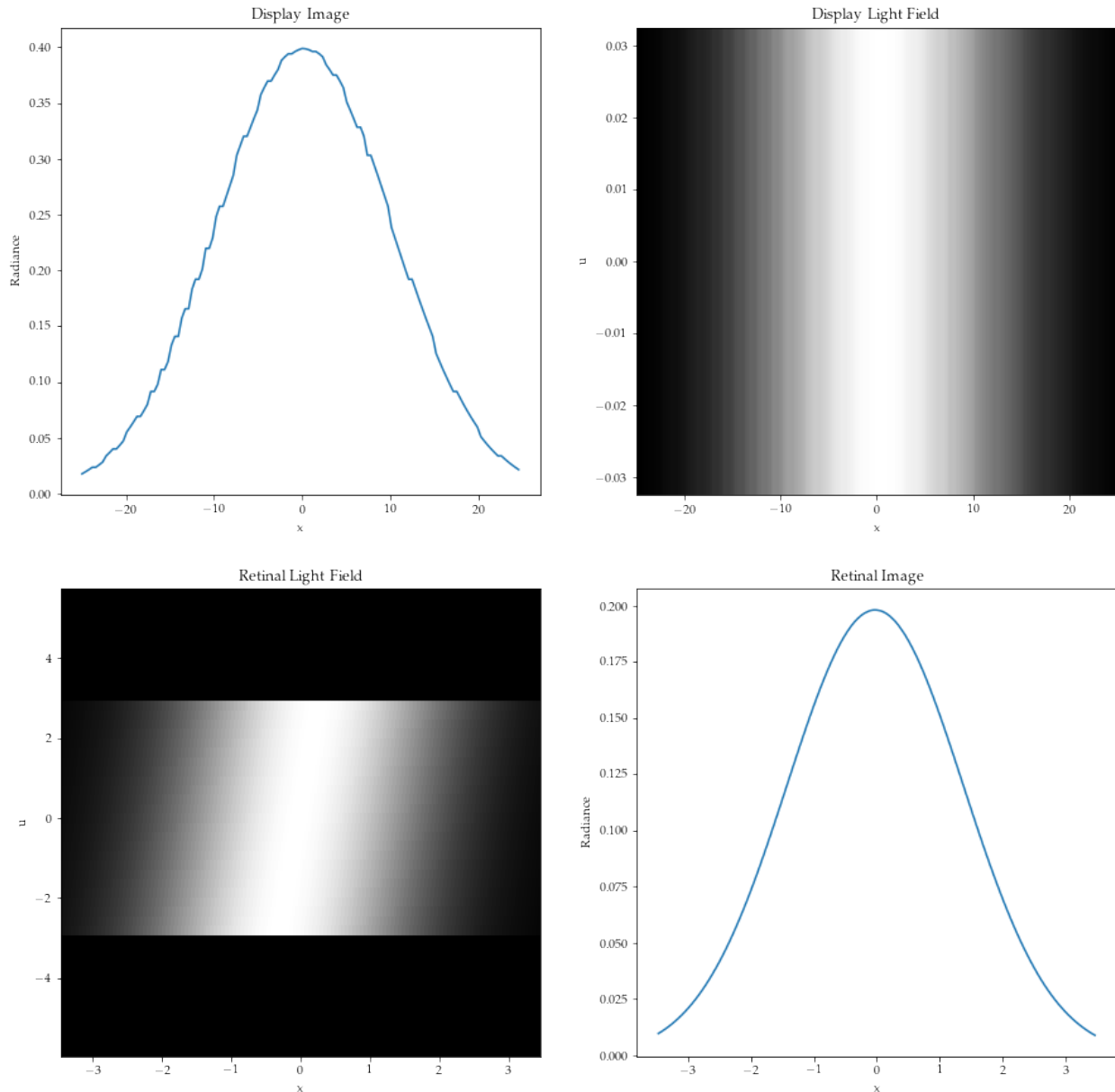
Figure 3.5: Image formation in 1D

is made to be piecewise constant on the display, its display lightfield is uniform across all angles. On the retina, this lightfield becomes sheared by the defocus in the system and

partially occluded by the aperture. Integrating over the angles results in a gaussian which is no longer has jagged edges but is instead entirely smooth, demonstrating the existence of blur. Therefore, the job of a vision correcting algorithm would be to create a display lightfield that, when sheared, the lightfield will integrate to reconstruct the original image.

Chapter 4

Compressed Sensing

The primary contribution of this work is to apply compressed sensing techniques to the vision correction problem. Compressed sensing, also known as compressive sampling, leverages sparsity, non-uniform sampling, and non-linear reconstruction to reconstruct signals with fewer samples than the Nyquist rate.

4.1   The Sensing Problem

Sampling, also called sensing, is the process of taking a continuous signal and storing enough information about it that the signal can be accurately reconstructed from the samples. Typically, this is done with *uniform sampling*, where one measurement is taken every T time-units where T is a constant known as the sampling rate. Mathematically, if we have a one-dimensional continuous-time signal, then our sampled signal $y(t)$ is given by

$$y(t) = x(t) \sum_{k=-\infty}^{\infty} \delta(t - kT) \tag{4.1}$$

where $\delta(t)$ is the Dirac delta. Since $y(t)$ is zero everywhere except at kT, we can easily represent it as a disrete-time signal $y[n] = y(nT)$. If $x(t)$ is bandlimited in the frequency domain by B, then the Nyquist Criterion says that we can perfectly reconstruct $x(t)$ from the measurements using a simple linear reconstruction scheme so long as $T < \frac{B}{\pi}$.

However, the uniform sampling scheme does not capture the full generality of sensing. Suppose $\mathbf{x}$ is a signal which belongs to some vector space. One way to think of a measurement would be to project $\mathbf{x}$ onto another vector ϕ_i. In this sensing scheme, each measurement y_i is the inner-product of $\mathbf{x}$ with a vector from a measurement basis Φ,

$$y_i = \langle \phi_i, \mathbf{x} \rangle. \tag{4.2}$$

Under this general sensing framework, uniform sampling is where each measurement vector is a delta function $\phi_i = \delta(t - iT)$ shifted by the the sampling rate T.

4.2 Reconstruction from Compressed Measurements

Once m measurements are taken according to eq. (4.2), producing $\mathbf{y} \in \mathbb{C}^m$, we need a method to accurately reconstruct the signal. To do this, we need some assumptions on Φ and the signal $\mathbf{x}$. As described by Candes and Wakin in [8], suppose $\mathbf{x} \in \mathbb{R}^n$ and $m \ll n$. Further assume that $\mathbf{x}$ is $S-$sparse in the basis Ψ. "$S-$sparse" means that in the coordinates of Ψ, the signal $\mathbf{x}$ has $n - S$ zero coordinates. Under these assumptions, $\mathbf{x}$ can be reconstructed from $\mathbf{y}$ using a convex optimization problem over $\tilde{\mathbf{x}} = \Psi^{-1}\mathbf{x}$ and $\mathbf{y}$, a vector of m randomly selected measurements taken according to Φ:

$$\begin{aligned} \text{argmin} \quad & \|\tilde{\mathbf{x}}\|_1 \\ \text{s.t } & y_i = \langle \phi_i, \Psi\tilde{\mathbf{x}}, \rangle \quad \forall i. \end{aligned} \tag{4.3}$$

As shown in [6], the solution to eq. (4.3) is exact so long as

$$m \geq C\mu^2(\Phi, \Psi)S \log n \tag{4.4}$$

where C is a positive constant and $\mu(\Phi, \Psi)$ is the coherence of the Φ and Ψ bases. The coherence of two bases is simply the maximal inner product of each pair of basis vectors

$$\mu(\Phi, \Psi) = \sqrt{n} \max_{1 \leq k, j \leq n} |\langle \phi_k, \psi_j \rangle|. \tag{4.5}$$

It measures how much Φ "smears" the signals information in the Ψ basis. When μ is small, then m can be quadratically smaller. Moreover, eq. (4.4) implies that one only needs to take $O(S \log n)$ measurements to perfectly reconstuct an $S-$sparse finite signal. Since one needs logarithmically fewer measurements, compressed sensing schemes speed up signal acquisition. For example, Lustig et al. demonstrate in [14] how Compressed Sensing could improve MRI data collection speed without sacrificing image quality.

However, few real-world signals are perfectly $S-$sparse. Instead, they are approximately $S-$sparse, meaning in the Ψ basis, there are S large coordinates and $n-S$ coordinates which are approximately zero. Fortunately, eq. (4.3) still works well so long as the measurements taken approximately preserve the norms of $S-$sparse signals. Mathematically, given a matrix A, the isometry constant δ_S as defined in [8] is given by

$$\begin{aligned} \delta_S = \min_{\delta} \quad & \delta \\ \text{s.t } & (1 - \delta)\|\mathbf{x}\|_2^2 \leq \|A\mathbf{x}\|_2^2 \leq (1 + \delta)\|\mathbf{x}\|_2^2 \quad \forall \mathbf{x} \in \mathbb{R}^n, \|\mathbf{x}\|_0 = S. \end{aligned} \tag{4.6}$$

δ_S is a measure of how well A preserves the norms of vectors $\mathbf{x}$ with exactly S non-zero entries. In compressed sensing, the matrix A is the $m \times n$ matrix which represents the constraint $\mathbf{y} = A\tilde{\mathbf{x}}$ in eq. (4.3). Candes and Tao demonstrated in [7] that if the sensing matrix A has a restricted isometry of order $2S$, meaning $\delta_{2S} < \sqrt{2} - 1$, then the solution to eq. (4.3) will extract the S largest coefficients of $\tilde{\mathbf{x}}$, giving a near perfect reconstruction. In this work, we will call this property the *Restricted Isometry Property* (RIP). Sensing

matrices which satisfy the RIP can be used to reconstruct approximately $S-$sparse signals using eq. (4.3). As stated in [8], if the matrix A is a random subset of rows from $\Phi\Psi$ where Φ and Ψ are orthogonal bases, then it will satisfy the RIP with high probability if

$$m \geq CS(\log n)^4 \tag{4.7}$$

where C is a positive constant.

4.3 Designing Sensing Matrices

In order for compressed sensing to be used in practice, we need two properties:

1. The signal $\mathbf{x}$ should be approximately $S-$sparse in some basis Ψ. This is typically not a problem because common signals such as natural images are sparse in a number of Wavelet bases [16].

2. The measurement matrix Φ should satisfy the RIP. Note that previously, we used Φ to represent the measurement basis, but going forward, we will use Φ to mean the sensing matrix itself, which is a random subset of measurement basis vectors.

There exist universal sensing matrices Φ which satisfy the RIP no matter which basis $\mathbf{x}$ is sparse in. These matrices include gaussian random matrices and matrices where each entry is drawn from a symmetric Bernoulli distribution [8]. However, these matrices are difficult to work with for large $\mathbf{x}$ because they are dense and take up lots of memory. To get around this issue, [10] proposed the *Structurally Random Matrix* (SRM), another universal sensing matrix. The SRM is a product of three matrices

$$\Phi = \sqrt{\frac{N}{M}}DFR. \tag{4.8}$$

- The matrix R is a randomization. It can either be a uniform random permutation matrix or a matrix which flips the sign of each coordinate of $\mathbf{x}$ in an i.i.d Bernoulli fashion.

- F is an orthonormal fast transformation such as the FFT, the DCT, or the Walsh-Hadamard Transform. The fast transformation spreads the energy of the signal over all the measurements.

- D is a downsampling matrix which randomly selects m rows from FR.

Each of these operations is linear and can be represented by a matrix, and they also have fast implementations that require no matrix computations at all. Moreover, the transpose of each matrix is easy and fast to compute. This can be important when implementing compressed sensing using iterative techniques which require computing Φ^T.

- $R^T = R^{-1}$ since R is always an orthonormal matrix. A uniform permutation matrix is a permutation of the columns of the identity matrix, so it is orthonormal, and a diagonal matrix with ± 1 entries on the diagonal is also orthonormal.

- $F^T = F^{-1}$ since F is an orthonormal transform. Thus if F is a fast transform that also has a fast inverse, F^T is easy to compute.

- D^T expands the input, placing a 0 at nonsampled coordinates. This is easiest to see by example.

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad D\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} a \\ c \end{bmatrix} \quad D^T\begin{bmatrix} a \\ c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} a \\ c \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ c \\ 0 \end{bmatrix}$$

Since the SRM is a universal sensing matrix, it means we do not have to know in which basis Ψ the signal $\mathbf{x}$ we want to measure is sparse.

4.4 Compressive Deconvolution

In the traditional deconvolution problem, we have access to a signal $H\mathbf{x}$ where $\mathbf{x}$ is an unblurred signal and H represents a convolution with some point spread function. In general, the deconvolution problem is ill-conditioned, which is why multiplying by H^{-1} is not a viable solution. However, the ideas from Compressed Sensing can be applied to approximate a solution to the problem. Since there is a blur H in the acquisition system, the measurements become

$$\mathbf{y_i} = \langle \phi_\mathbf{i}, H\mathbf{x} \rangle \qquad \mathbf{y} = \Phi H \mathbf{x}. \tag{4.9}$$

This makes our effective sensing matrix ΦH, and we can now attempt to reconstruct $\mathbf{x}$ using eq. (4.3). In [3], Bahmani and Romberg showed that if the blur H is known, then eq. (4.3) can be modified to accurately reconstruct $\mathbf{x}$ despite the blur if the measurements Φ are a random mask. Similarly, in [9], Chen et. al. demonstrated that they could deblur ultrasound images when they used the Structurally Random Matrix from [10] as their measurement matrix. Both [3] and [9] demonstrate that Compressive Deconvolution could be another valid approach to tackle the VCD problem.

Chapter 5

Compressive Deconvolution For Vision Correction

5.1 Motivation

In chapter 2, we discussed prior approaches to solving the vision correction problem. Broadly speaking, Lightfield Projection and Forward Optimization can be described as optimization-based approaches, while the Point-to-Point and Point-to-Area algorithms are ray-tracing algorithms. The optimization-based approaches first construct a matrix which characterizes the relationship between the display and the retina. Then they set up and solve a least squares problem to determine the image placed on the display. In contrast, the ray-tracing approches trace rays between the screen and the retina and assign colors to screen pixels using heuristics like uniform integration or bilinear interpolation. The ray-tracing algorithms are significantly faster than the optimization-based algorithms. However, they are also less accurate because they leverage heuristics instead of finding the optimum of a convex optimization problem. In order to be used practically, vision correction algorithms must run in real-time, so there is a great need to create an algorithm which has the speed of the ray-tracing algorithms but the accuracy of the optimization-based algorithms.

One reason the optimization algorithms are so slow is because they involve large matrices which scale poorly with the size of the display image. If the image is $n \times n$ pixels, then the lightfield projection and forward optimization matrices each have $O(n^4)$ entries since they map an $O(n^2)$ input to an $O(n^2)$ output. Constructing a matrix with fewer entries would necessarily reduce the time needed to compute the matrix and solve an optimization problem involving that matrix, but it would need to be done in a manner which does not significantly impact the fidelity of the solution to the optimization. Since both lightfields [2] and natural images are sparse in a Wavelet domain, compressed sensing provides a natural solution. With compressed sensing, if properties such as the restricted isometry property (RIP) are satisfied, then we only need to take $O(\log n)$ measurements, yielding a matrix with $O(n^2 \log n)$ entries. For large n, this is significantly fewer entries to compute and has the potential to yield a

large speedup without sacrificing accuracy.

5.2   Problem Setup

Let $\mathbf{y}$ be the image sensed by the retina, and let H be a matrix which captures the relationship between the display and the retina. This relationship can either be between the display lightfield and the retinal image as in the Lightfield Projection algorithm, or it can be directly between the screen pixels and the retinal image as in the Forward Optimization algorithm. Regardless of which matrix H is used, let the target variable be $\mathbf{x}$. If H is the lightfield projection matrix, then $\mathbf{x}$ represents the display lightfield, and if H is the forward optimization matrix, then $\mathbf{x}$ represents the screen image. For both approaches,

$$\mathbf{y} = H\mathbf{x}. \tag{5.1}$$

The image placed on the display, which is determined based on $\mathbf{x}$, is known as the *prefilter*. Compressed sensing is typically used when reconstructing images obtained from sensor data. The sensor takes measurements of the target $\mathbf{x}$ according to a measurement matrix Φ. If ϕ_i denotes the ith row of Φ, then the ith measurement y_i is given by $y_i = \langle \phi_i, \mathbf{x} \rangle$. For example, the Φ for the retina is the identity matrix since the retina senses in the space domain. However, the VCD problem is not like traditional compressed sensing problems for three key reasons.

1. We have access to the measurements if the eye is unaberrated.

2. We have no control over the sensor. The retina's measurement matrix will always be the identity.

3. We cannot measure $\mathbf{x}$ directly, we can only measure $H\mathbf{x}$.

However, we can still utilize compressed sensing by conceptualizing an imaginary sensing setup. Pretend that instead of an eye, we have a camera which is viewing the vision correcting display. The camera has a lens with the same blurring characteristics as the viewer's eye (i.e it has the same matrix H). Rather than acting like a true camera where $\Phi = I$, this special camera takes measurements according to a desired measurement matrix Φ. Thus the measurements captured by the camera will be

$$\hat{\mathbf{y}} = \Phi H\mathbf{x}. \tag{5.2}$$

We want the user to see the unblurred image $\mathbf{y}^*$, which means we want the camera to see $\hat{\mathbf{y}}^* = \Phi\mathbf{y}^*$. Under this setup, prefiltering an image for the vision correcting display becomes a question of finding the correct $\mathbf{x}$ such that $\hat{\mathbf{y}}^* \approx \Phi H\mathbf{x}$. This setup is identical to the Compressive Deconvolution setup described in [9]. If we assume that $H\mathbf{x}$ is sparse in some

Wavelet basis Ψ, then based on [9], we can recover the target $\mathbf{x}$ with an appropriately designed Φ by solving the optimization problem

$$\min_{\mathbf{x}} \frac{1}{2\mu}\|\hat{\mathbf{y}}^* - \Phi H\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_p^p + \gamma\|\Psi^{-1}H\mathbf{x}\|_1, \quad p \in [1,2]. \tag{5.3}$$

Each component of eq. (5.3) has an interpretable reason for existing in the optimization.

- $\frac{1}{2\mu}\|\hat{\mathbf{y}}^* - \Phi H\mathbf{x}\|_2^2$ is a penalty on the reconstruction error where μ is a hyper-parameter. This term encourages the solution $\mathbf{x}$ to produce the desired measurements when viewed by the camera. Smaller μ means this term will be emphasized more, while larger μ will relax how well $\mathbf{x}$ has to match the measurements.

- $\lambda\|\mathbf{x}\|_p^p$ is a regularization term which regulates against noise in the signal $\mathbf{x}$. $p = 2$ implicitly assumes $\mathbf{x}$ is corrupted by Gaussian noise, whereas $p = 1$ implicitly assumes $\mathbf{x}$ is corrupted by Laplacian noise. λ is related to the variance of the noise distribution. The primary source of noise in our setup, if any at all, is from discretization error.

- $\gamma\|\Psi^{-1}H\mathbf{x}\|_1$ is the sparsity penalty. It helps guarantee that $H\mathbf{x}$ is sparse in a wavelet basis Ψ. The size of γ determines how sparse we believe $H\mathbf{x}$ to be, with larger γ imposing a higher sparsity requirement. In traditional compressed sensing, the sparsity penalty is on the reconstructed vector $\mathbf{x}$, but as per [9], because there are no guarantees that ΦH satisfies the RIP, it is safer to penalize the sparsity of the image seen by the retina.

For additional simplicity, in eq. (5.3), we set $\lambda = 0$ so there is no regularization on $\mathbf{x}$, giving

$$\min_{\mathbf{x}} \frac{1}{2\mu}\|\hat{\mathbf{y}}^* - \Phi H\mathbf{x}\|_2^2 + \gamma\|\Psi^{-1}H\mathbf{x}\|_1, \quad p \in [1,2]. \tag{5.4}$$

5.3 Implementation

In implementing and solving eq. (5.4), we have the freedom to choose both the sensing matrix Φ as well as the algorithm used to solve eq. (5.4). Since the purpose of this work is to investigate whether or not compressed sensing is a viable approach to solving the VCD problem, we chose to use the Structurally Random Matrix (SRM) introduced in [10] as the sensing matrix Φ. The SRM has the benefit of being incoherent with any Wavelet basis, and it is computationally fast to implement. Our implementation of the SRM used the orthonormal Discrete Cosine Transform (DCT) as the fast transformation and a uniform random permutation matrix as the pre-randomizer. However, it should be noted that in a practical implementation of Compressive Deconvolution for VCD, the SRM is an illogical choice because computing ΦH requires computing the full H matrix, which needs to be avoided in order for Compressive Deconvolution to bring any speedup. Nevertheless, the fact that the SRM is incoherent with all Wavelet bases makes it ideal for the proof of concept

we aim to provide in this work. To solve eq. (5.4), we chose to use the Alternating Method of Direct Multipliers (ADMM) algorithm, introduced in [4] and used in [9] for compressive deconvolution. ADMM solves the optimization problem

$$
\begin{aligned}
\min_{\mathbf{u},\mathbf{v}} \quad & f(\mathbf{u}) + g(\mathbf{v}) \\
\text{s.t} \quad & B\mathbf{u} + C\mathbf{v} = \mathbf{b},
\end{aligned}
\tag{5.5}
$$

by breaking it into smaller subproblems which iteratively optimize the lagrangian

$$
\mathcal{L}(\mathbf{u}, \mathbf{v}, \lambda) = f(\mathbf{u}) + g(\mathbf{v}) - \lambda^\top (B\mathbf{u} + C\mathbf{v} - b) + \frac{\beta}{2}\|B\mathbf{u} + C\mathbf{v} - \mathbf{b}\|_2^2, \quad \beta > 0.
\tag{5.6}
$$

The subproblems are

$$
\begin{aligned}
\mathbf{u}_{k+1} &= \operatorname*{argmin}_{\mathbf{u}} \mathcal{L}(\mathbf{u}, \mathbf{v}_k, \lambda_k) \\
\mathbf{v}_{k+1} &= \operatorname*{argmin}_{\mathbf{v}} \mathcal{L}(\mathbf{u}_k, \mathbf{v}, \lambda_k) \\
\lambda_{k+1} &= \lambda_k - \beta(B\mathbf{u}^{k+1} + C\mathbf{v}^{k+1} - \mathbf{b})
\end{aligned}
\tag{5.7}
$$

To apply ADMM to *eq.* (5.4), we can set

$$
\mathbf{u} = \mathbf{x}, \qquad \mathbf{v} = \Psi^{-1}H\mathbf{x}, \qquad \Phi H\mathbf{u} - \Phi\Psi\mathbf{v} = 0.
\tag{5.8}
$$

This yields the subproblems

$$
\begin{aligned}
\mathbf{u}_{k+1} &= \operatorname*{argmin}_{\mathbf{u}} \frac{1}{2\mu}\|\hat{\mathbf{y}}^* - \Phi H\mathbf{u}\|_2^2 - \lambda_k^\top \Phi H\mathbf{u} + \frac{\beta}{2}\|\Phi H\mathbf{u} - \Phi\Psi\mathbf{v}_k\|_2^2 \\
\mathbf{v}_{k+1} &= \operatorname*{argmin}_{\mathbf{v}} \gamma\|\mathbf{v}\|_1 - \lambda_k^\top \Phi\Psi\mathbf{v} + \frac{\beta}{2}\|\Phi H\mathbf{u}_k - \Phi\Psi\mathbf{v}\|_2^2 \\
\lambda_{k+1} &= \lambda_k - \beta(\Phi H\mathbf{u}_{k+1} - \Phi\Psi\mathbf{v}_{k+1})
\end{aligned}
\tag{5.9}
$$

$$
\tag{5.10}
$$

Each of the subproblems in eq. (5.9) can be solved using existing solvers. We use L-BFGS-B [5] to compute $\mathbf{u}_{k+1}$ since it allows us to force $0 \le \mathbf{u} \le 1$. For $\mathbf{v}_{k+1}$, we can use OWL-QN [1], a variant of L-BFGS-B that allows for l_1-norm optimization. Note that each subproblem in eq. (5.9) only depends on ΦH, not H alone, meaning we can compute the smaller ΦH and not the full H matrix. ΦH has significantly fewer entries than H, and this can bring the speedup desired over other methods which require computing H. This makes ADMM a practical choice for solving the VCD problem with compressive deconvolution.

Parameter	Value
μ	10^{-2}
γ	10^{-3}
β	10^0
Focal Length	$20\,\text{mm}$
Aperture Radius	$1.5\,\text{mm}$
Near Point Distance	$500\,\text{mm}$
Screen Pixel Pitch	$0.078\,\text{mm}$
Pinhole Mask Separation Distance	$6.0\,\text{mm}$

Table 5.1: Parameters for Compressive Deconvolution Experiments.

5.4 Evaluation

We evaluate the proposed Compressive Deconvolution framework using two different H matrices. The first matrix H is the Forward Optimization matrix introduced by [17]. As described in chapter 2, this matrix models the relationship between screen pixels and the retinal image, and is constructed by tracing rays according to eq. (B.2). The second matrix we evaluate our framework with is the Lightfield Projection matrix introduced in [12]. This matrix models the relationship between the lightfield at the display and the image at the retina. It is constructed using the light transport matrices from [13], and it discretizes eq. (3.3). The hyperparameters used in all the experiments are provided in table 5.1. The hyper-parameters to the optimization μ, γ, β where chosen based on emprical performance. The remaining parameters were chosen to model a 326 PPI screen and a presbyopic eye under normal lighting conditions.

Figure 5.1 shows the corrected images at various compressed sensing ratios. These are the images that the user would see if the prefilter from the forward optimization matrix and lightfield projection matrix were placed on the display. It is immediately clear that the generated prefilters, when viewed by the user, appear in focus. However, depending on the compressed sensing ratio, there is grain in the reconstructed image. The amount of grain is inversely related to the compressed sensing ratios (i.e higher ratios mean lower grain). This indicates that grain arises because of information lost in the measurement taking process. Despite the grain, the PSNR of the reconstructed image from the lightfield projection matrix is better than the uncorrected image with just a 25% compressed sensing ratio. In contrast, the reconstructed image from the forward optimization matrix only yields an improvement in PSNR over the uncorrected image starting at a compressed sensing ratio around 85%. This indicates that for the compressive deconvolution problem, modeling the problem using lightfields performs better than modeling the problem by tracing rays from the screen to the retina. Figure 5.2 shows the PSNRs achieved by both methods across all of the compressed sensing ratios tested.
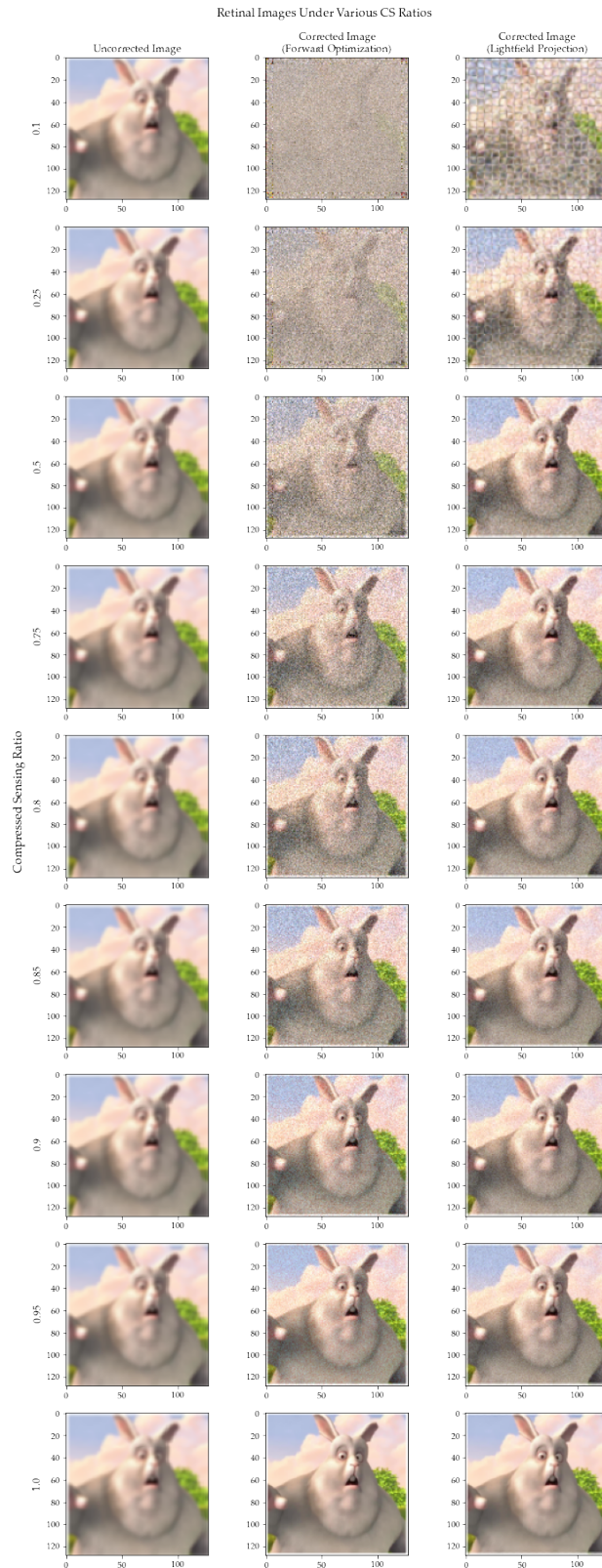
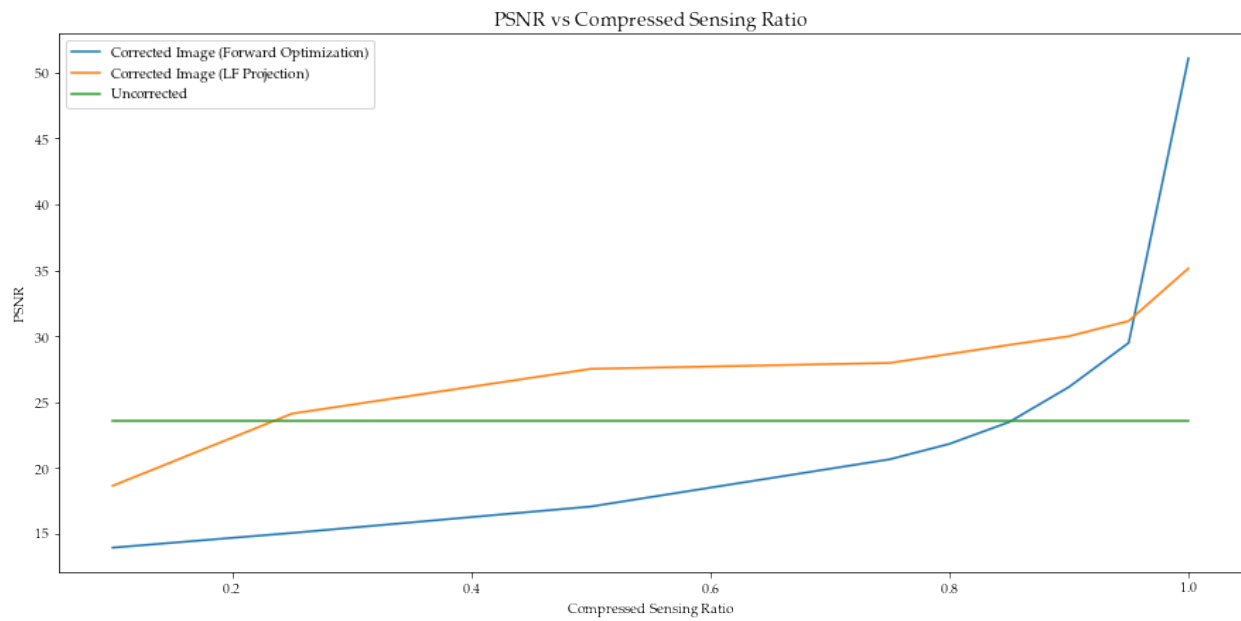Figure 5.1: Retinal images at various compressed-sensing ratios.

Figure 5.2: PSNR of the retinal image at various compressed-sensing ratios.

In addition to testing reconstruction accuracy across compressed sensing ratios, we also checked that the compressive deconvolution approach worked across different levels of blur in fig. 5.3. These images were generated by holding the compressed ratio fixed at 0.8 and varying the viewing distance. Though grain is present, the reconstructed retinal image is still in focus. When compared to the reconstructed retinal image from prefilters generated using least squares, the grain is the only downside to compressive deconvolution in terms of the quality of the prefilter. Interestingly, the grain decreases as the display moves closer to the focal plane when using the lightfield projection matrix, whereas the grain is constant for all distances when using the forward optimization matrix. The plot of the PSNR of the reconstructed retinal image for the different prefilters also demonstrates the visual improvement over the uncorrected images (fig. 5.4). The forward optimization compressive deconvolution prefilter yields nearly the same performance across the viewing range. Since the uncorrected image improves in quality as the display nears the focal plane, this means that the forward optimization matrix is un-ideal for small defocus. In contrast, the compressive prefilter from the lightfield projection matrix consistently provides an improvement over the uncorrected image.

Analysis of Grain

With both the lightfield projection matrix and the forward optimization matrix, solving eq. (5.4) produces a prefilter which, when projected onto the retina, produces a grainy retinal image. This phenomenon occurs regardless of the Wavelet transform used, indicating that it is not caused by a lack of sparsity in the unblurred image. Instead, the grain is caused by imperfect reconstructions in the high frequency sub-bands of the images. For example, consider the prefilter generated with a compressed sensing ratio of 0.5 for the lightfield projection matrix and the prefilter with a compressed sensing ratio of 0.85 for the forward optimization matrix, both with a viewing distance of 200 mm. If we replace the coefficients in sub-bands 8 - 16 of the retinal image which differ from the true wavelet coefficients by more than 0.05, the new retinal images have no grain (depicted in fig. 5.5). For lightfield projection, 18% of all coeffients satisfy this criterion, and for the forward optimization matrix, 49.5% of all coefficients satisfy this criterion. What this demonstrates is that the grain is caused by incorrectly reconstructing the high frequency wavelet coefficients in the retinal image. Since the grain disappears with higher sensing ratios, the number of measurements is tightly connected to how much grain exists. Moreover, because the lightfield projection matrix produces much less grain that the forward optimization matrix, how we model the vision correction problem is also important (solving for a lightfield vs solving for screen pixel values directly). The dependence on the matrix likely has to do with properties of ΦH such as whether or not it satisfies the RIP.
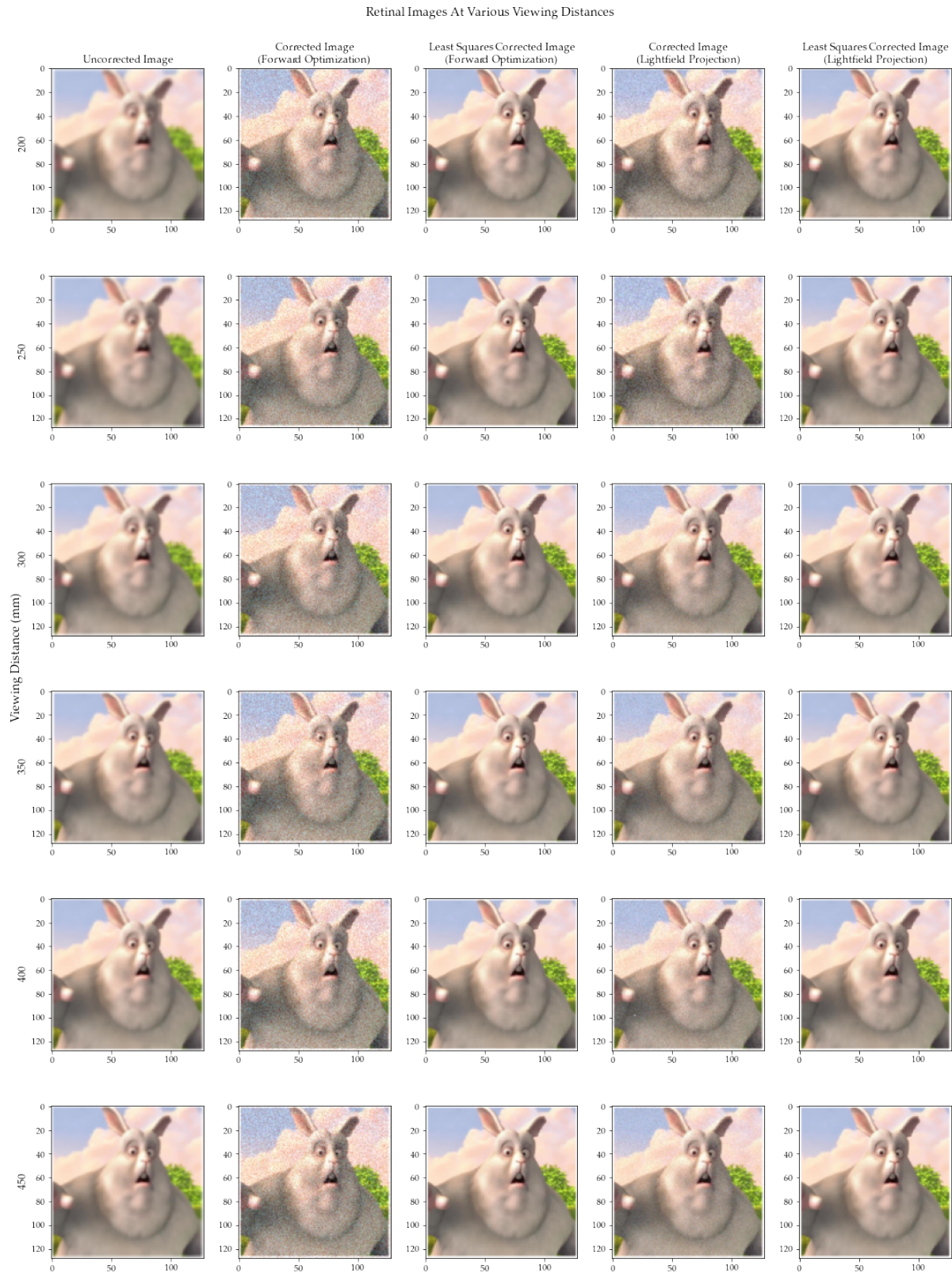
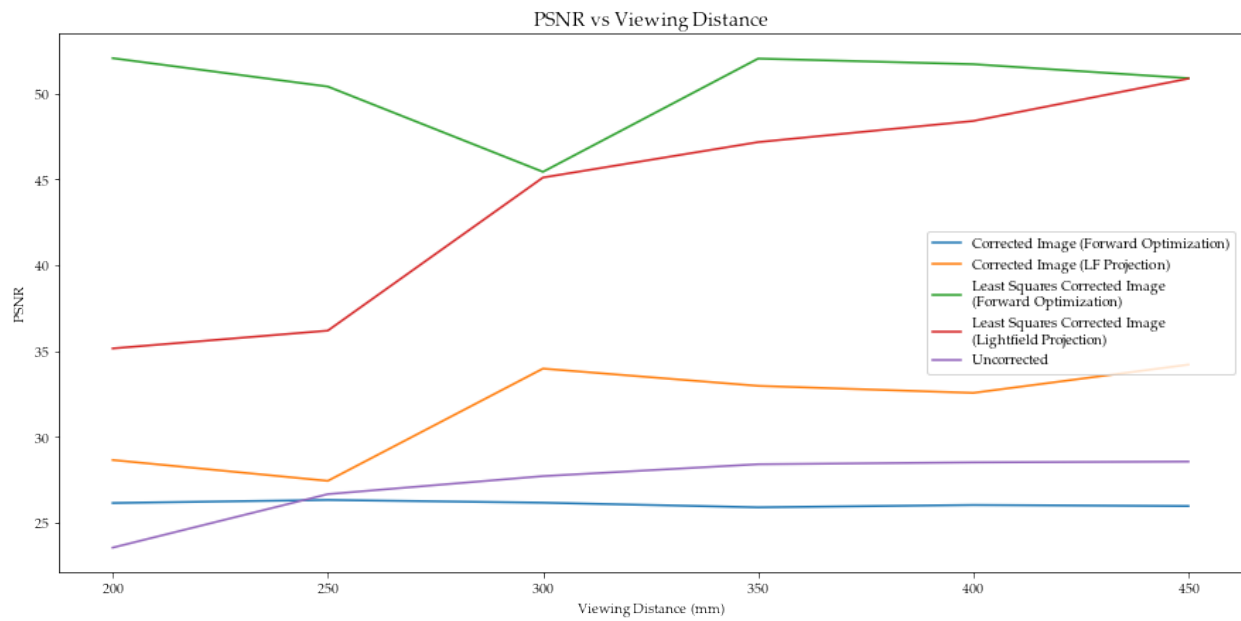Figure 5.3: Retinal images at various viewing distances.

Figure 5.4: PSNR of the retinal image at various viewing distances.
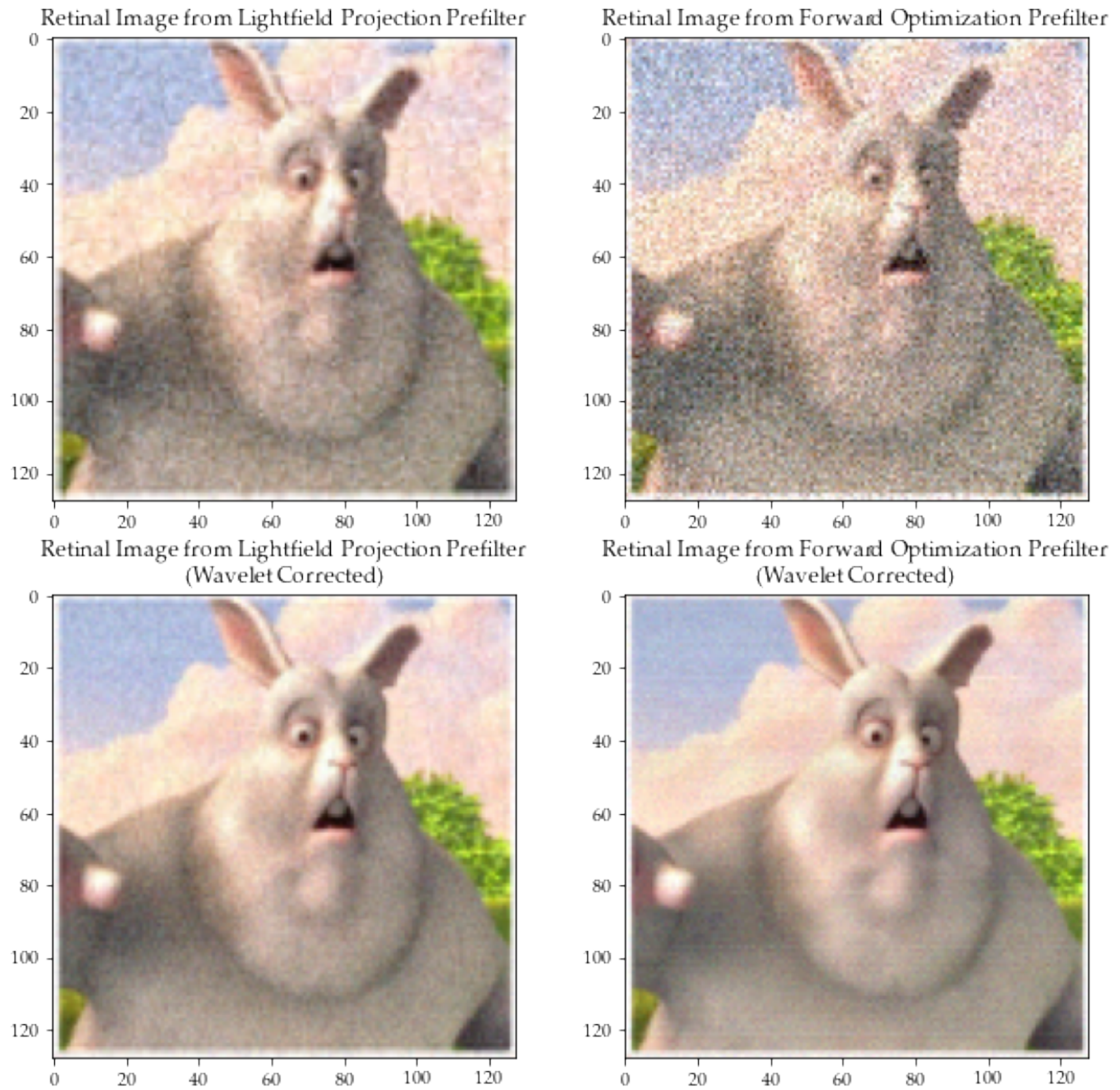
Figure 5.5: De-grained retinal images from compressive deconvolution prefilters

Chapter 6

Lightfield Blur

6.1 Motivation

As demonstrated in chapter 5, modeling the relationship between the display lightfield and
the retinal image (lightfield projection) is better for compressive deconvolution than modeling
the relationship between the display image and the retinal image (forward optimization).
However, in modeling the relationship between the display lightfield and the retinal image,
we are restricting the type of measurements that we can take of the display lightfield. In
particular, the lightfield projection matrix H, which models this relationship, is really the
product of two matrices: an "integration" matrix P and a "blur" matrix B,

$$H = PB. \tag{6.1}$$

The matrix P discretizes the integral in eq. (3.3), while the matrix B discretizes the rela-
tionship

$$L_r(\mathbf{z}) = L_d(M^{-1}\mathbf{z})A(\mathbf{u})$$

where L_r is the retinal lightfield, L_d is the display lightfield, M is the light transport matrix,
and A is the aperture function. Since the matrix P is integrating out the angular views for
every pixel in the image, it is restraining the types of measurements that we can take in
the compressive deconvolution setting. In particular, the set of measurements that we can
take is restricted to linear combinations of retinal pixel values. In other words, we lose the
freedom to take measurements in the angular dimension. Thus, if we use the matrix B in
the compressive deconvolution framework rather than the matrix H, the problem changes
to sensing the retinal lightfield rather than the retinal image. This would enable us to use
sensing techniques that are specific to lightfield reconstruction such as coded apertures [2].
These techniques may yield increased performance, both in PSNR and speed, when used in
the compressive deconvolution framework.

6.2   Matrix Construction

The lightfield blur matrix is constructed in a manner which is almost identical to the the lightfield projection matrix.

1. Sample rays from each sensor pixel to the aperture.

2. Filter out the rays which fall outside of the aperture.

3. Use the light transport matrix M to determine the spatial location and angular direction of the rays on the lightfield display.

4. Filter rays which fall outside of the spatial extent of the display.

5. Assign the remaining rays to a retinal spatio-angular bucket i and a display spatio-angular bucket j.

6. Construct a square matrix where the $ij-$th entry is the number of rays which map to the tuple (i, j).

7. Divide each row of the matrix by the number of times a ray hits the same retinal spatial bucket that the row represents.

When the lightfield blur matrix B is constructed with the above procedure, the projection matrix P is simply a matrix which sums over the indices which correspond to each spatial bucket. For a more detailed account of how to construct the lightfield blur matrix, see appendix C.

6.3   Least Squares Optimization

To test whether the the lightfield blur matrix B can be used for vision correction, we first formulate a least squares optimization because if the least square formulation produces poor results, then the compressive deconvolution formulation will also produce poor results. In the least squares formulation, we solve the optimization problem

$$
\begin{aligned}
\min_{\mathbf{x}} \quad & \|\mathbf{y}_{LF} - B\mathbf{x}\|_2^2 \\
\text{s.t} \quad & \mathbf{y} = P\mathbf{y}_{LF}.
\end{aligned}
\tag{6.2}
$$

The target variable $\mathbf{y}_{LF}$ is the desired retinal lightfield. Since we want the user to see an unblurred image, we need $\mathbf{y}$ to be equal to the projection of $\mathbf{y}_{LF}$, which is why the new constraint is included in eq. (6.2). The vector $\mathbf{y}$ represents a discrete time signal $y[i, j]$ which is the unblurred image. The vector $\mathbf{y}_{LF}$ is a discrete-time lightfield $y_{LF}[i, j, u, v]$ where i, j are

spatial indexes and u, v are angular indexes. With this notation, the constraint in eq. (6.2) can be written as

$$y[i, j] = \sum_u \sum_v y_{LF}[i, j, u, v]. \tag{6.3}$$

Notice that eq. (6.3) is the discrete time version of the lightfield projection relationship given by eq. (3.2). Let $\mathbf{b}_{i,j,u,v}^\top$ be the row of the lightfield blur matrix B that corresponds to $y_{LF}[i, j, u, v]$. Then due to the manner in which the matrix was normalized,

$$\sum_u \sum_v \|\mathbf{b}_{i,j,u,v}\|_1 = 1 \tag{6.4}$$

Thus, if we let

$$y_{LF}[i, j, u, v] = \frac{1}{\|\mathbf{b}_{i,j,u,v}\|_1} y[i, j], \tag{6.5}$$

then eq. (6.3) holds. This is one possible solution to eq. (6.3), and we choose to use it because it stems from the intuition that when the image is unblurred, its lightfield should be constant for all angular direction (i.e it is Lambertian).

Results

The results of eq. (6.2) when setting $\mathbf{y}_{LF}$ accordding to eq. (6.5), are summarized in figs. 6.1 and 6.2. The experimental parameters are identical to those in table 5.1. Just as in chapter 5, we use L-BFGS-B to solve the optimization problem. The lightfield blur matrix does a good
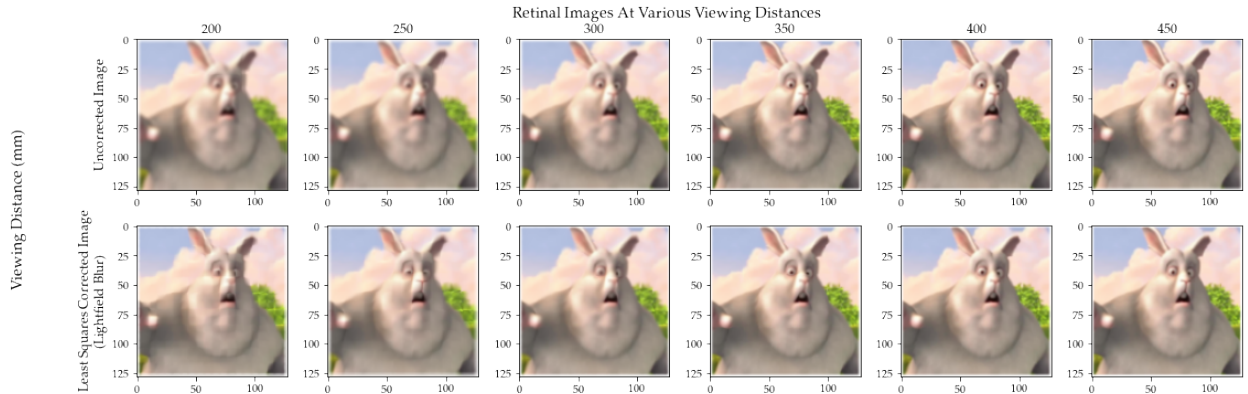
Figure 6.1: Retinal images of the Least Square Lightfield Blur prefilter at various viewing distance.

job of correcting for defocus blur, providing a PSNR improvement for all viewing distances tested. However, for strong level of defocus, the lightfield blur matrix appears to introduce blocking artifacts in the recovered retinal image which were not present in the recovered retinal images for the lightfield projection matrix at the same viewing distances.
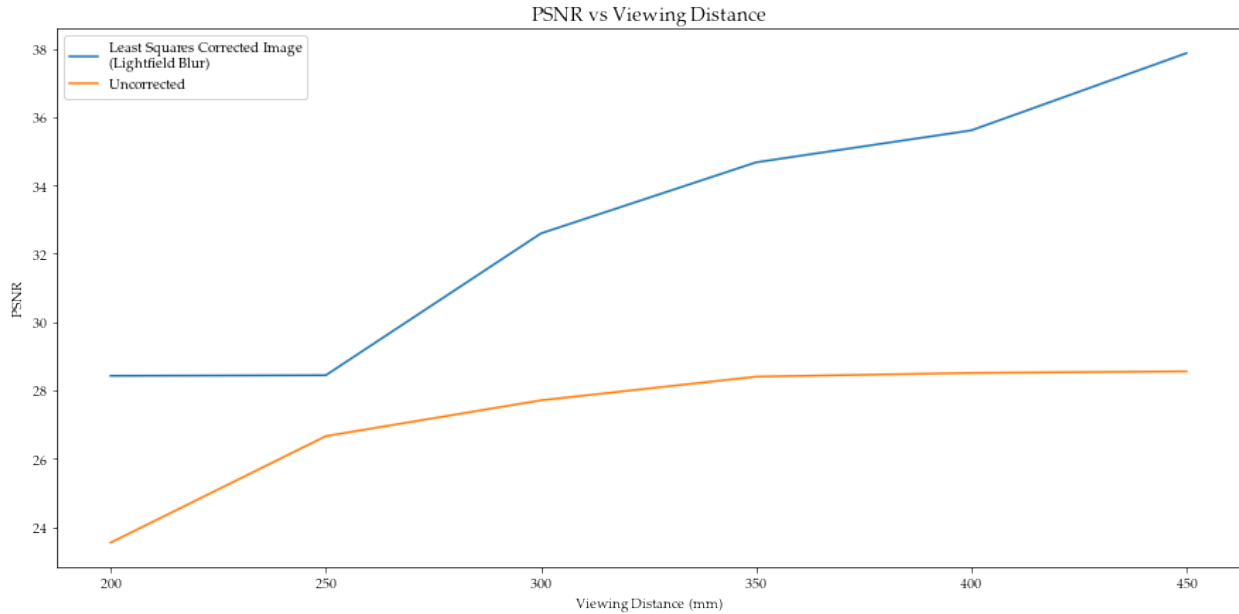
Figure 6.2: PSNR of the Least Squares Lightfield Blur prefilter at various viewing distances.

Analysis of Blocking

The blocking artifact is caused by the fact that for large amounts of defocus, it is impossible to recreate a Lambertian retinal lightfield. In other words, the $\mathbf{y}_{LF}$ from eq. (6.5) gets further and further away from the column space of B with higher amounts of defocus, contributing to larger reconstruction error in the form of blocking artifacts. To solve this issue, we need to consider other possible values of $\mathbf{y}_{LF}$. More specifically, each value of y_{LF} will produce a different prefilter $\mathbf{x}$. Setting $\mathbf{y}_{LF}$ according to eq. (6.5) is just one valid solution to the constraint. Figure 6.3 shows another possible $\mathbf{y}_{LF}$. This $\mathbf{y}_{LF}$ is what we would get if we took the prefilter generated by solving the least squares optimization problem with the lightfield projection matrix at a viewing distance of 200 mm. This lightfield produces an in-focus image when integrated, so it demonstrates that it is possible to have an unblurred retinal image without having a Lambertian retinal lightfield. Figure 6.4 confirms that when using the lightfield blur matrix, the choice of $\mathbf{y}_{LF}$ matters. The retinal image on the left in fig. 6.4 is what the user would see if we used the prefilter that is computed for a display 200 mm away from the user using the $\mathbf{y}_{LF}$ from eq. (6.5). If we instead set $\mathbf{y}_{LF}$ to the retinal lightfield shown in fig. 6.3, then the user sees the image on the right. The left image has blocking artifacts while the right image does not, indicating that the choice of $\mathbf{y}_{LF}$ made a significant difference in our ability to produce an unblurred retinal image. This issue does not appear when using the lightfield projection matrix because in that optimization problem, the retinal lightfield is left unconstrained.
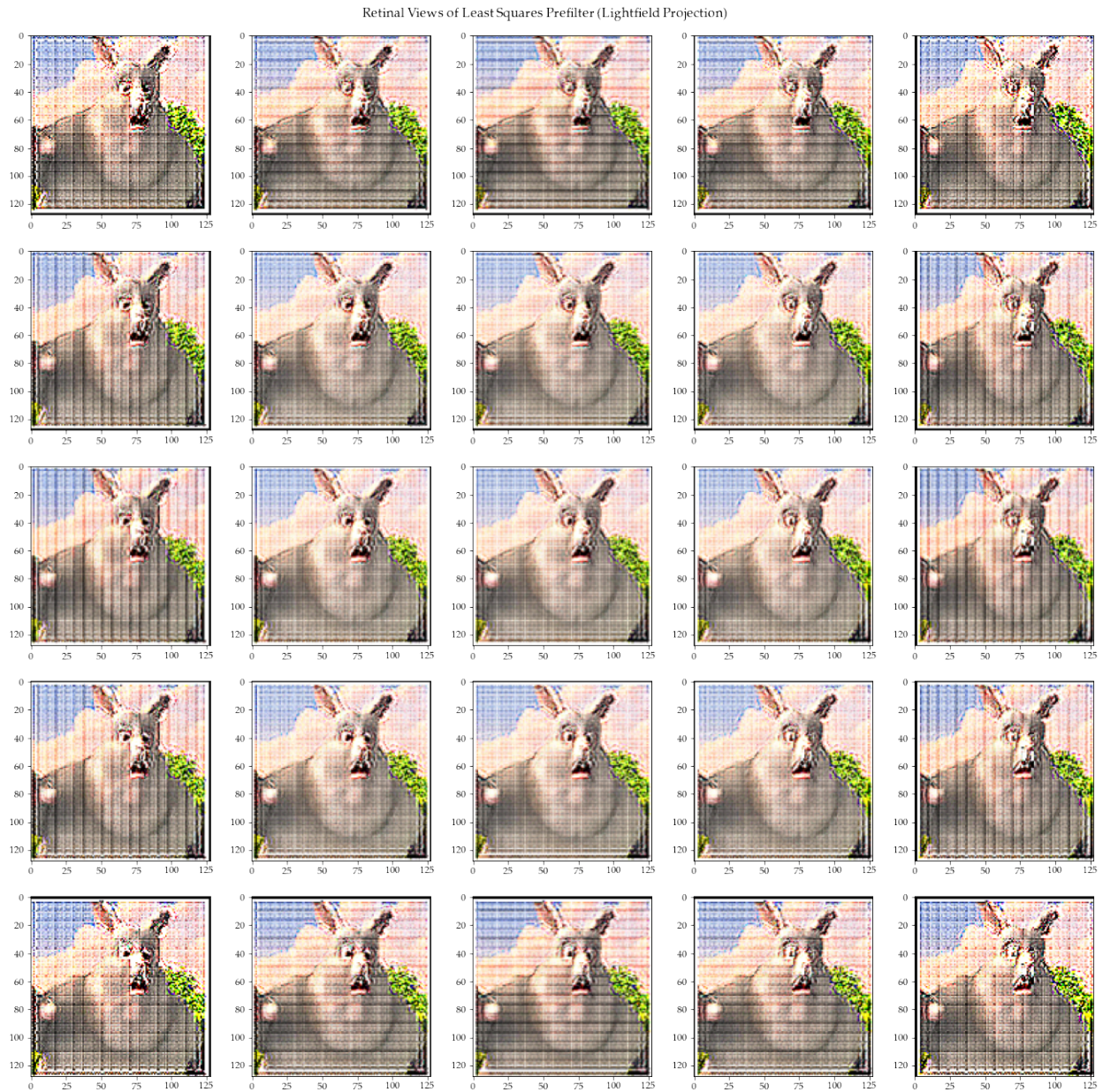
Figure 6.3: Retinal lightfield produced by the Least Squares Lightfield Projection prefilter.

Therefore, in order to achieve the best possible prefilter with the lightfield blur matrix, eq. (6.2) should really be an optimization problem over the target retinal lightfield as well.

Retinal Image of Least Squares Prefilter With Different Target Lightfields (Lightfield Blur)
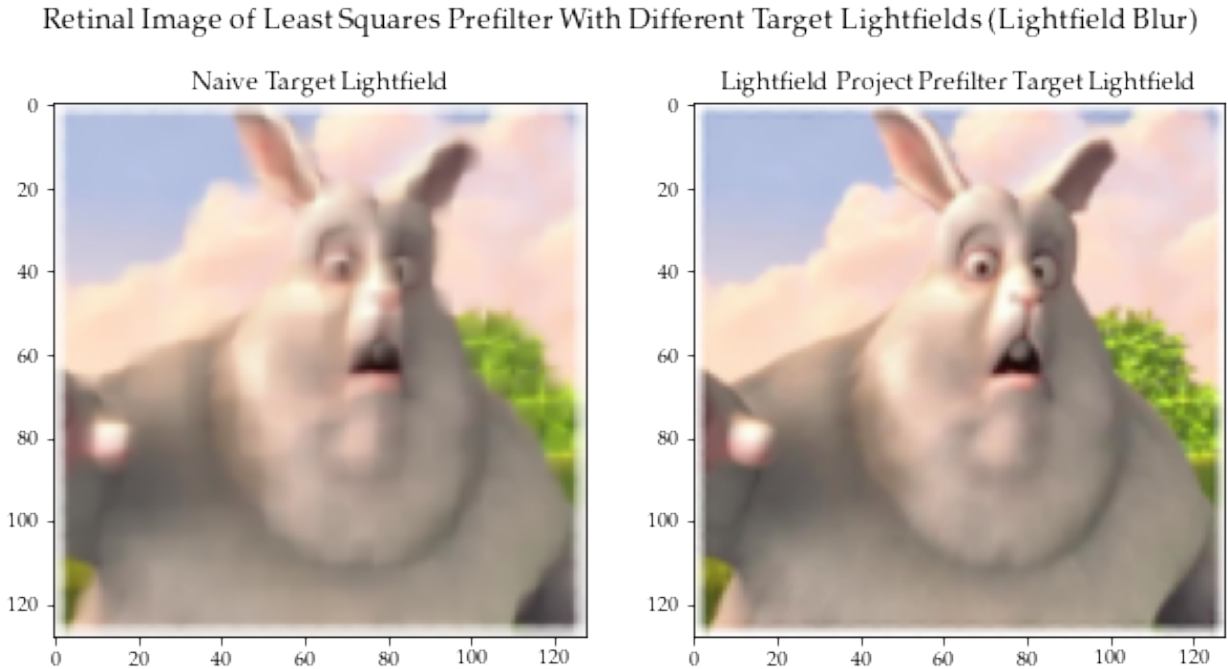
Figure 6.4: Retinal image produced by different target lightfields in Lightfield Blur Least Squares Optimization

Mathematically, this means we should solve the optimization problem

$$\min_{\mathbf{x},\mathbf{y}_{LF}} \quad \|\mathbf{y}_{LF} - B\mathbf{x}\|_2^2$$
$$\text{s.t} \quad \mathbf{y} = P\mathbf{y}_{LF}. \tag{6.6}$$

However, when attempting to solve eq. (6.6) using L-BFGS-B, the solution would not converge in a meaningful amount of time. Thus, in order to use a least squares technique with the lightfield blur matrix, we need to use a different optimization routine.

6.4   Compressive Deconvolution Formulation

The compressive deconvolution formulation of the vision correction problem using the lightfield blur matrix is similar to the formulation presented in chapter 5. The goal is to find the optimal display lightfield $\mathbf{x}$. As before, we pretend that we have an imaginary camera. However, rather than capturing images, this camera captures lightfields. The camera has a lens aberration described by the lightfield blur matrix B. We then design a sensing matrix Φ such that our hypothetical camera captures the measurements of the retinal lightfield

$$\hat{\mathbf{y}}_{LF} = \Phi B\mathbf{x}. \tag{6.7}$$

We want the image to appear unblurred on the retina, so if $\mathbf{y}^*_{LF}$ is a lightfield that satisfies $\mathbf{y} = P\mathbf{y}^*_{LF}$, then our objective is to find $\mathbf{x}$ such that $\hat{\mathbf{y}}^*_{LF} = \Phi\mathbf{y}^*_{LF} \approx \Phi B\mathbf{x}$. If we assume that $B\mathbf{x}$ is sparse in some Wavelet basis Ψ, then we can recover $\mathbf{x}$ with an appropriately designed Φ by solving the optimization problem

$$
\min_{\mathbf{x}, \mathbf{y}_{LF}} \frac{1}{2\mu} \quad \|\hat{\mathbf{y}}_{LF} - \Phi B\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_p^p + \gamma\|\Psi^{-1}B\mathbf{x}\|_1
$$
$$
\text{s.t} \quad \mathbf{y} = P\mathbf{y}_{LF}
$$
$$
\hat{\mathbf{y}}_{LF} = \Phi\mathbf{y}_{LF} \tag{6.8}
$$
$$
p \in [1, 2].
$$

There are a few important differences between eq. (6.8) and eq. (5.3).

1. In eq. (5.3), the target $\hat{\mathbf{y}}^*$ is easily found by taking the unblurred image $\mathbf{y}$ and multiplying it by Φ. However, in eq. (6.8), we don't know $\mathbf{y}^*_{LF}$ *a priori*, so this adds two constraints to the optimization problem.

2. Whereas in eq. (5.3) we were penalizing the sparsity on the blurred retinal image, in eq. (6.8), we penalize the sparsity of the blurred retinal lightfield $B\mathbf{x}$.

Solving this optimization problem will yield the desired $\mathbf{x}$ which will present an unblurred retinal image to the user when placed on a vision correcting display.

Chapter 7

Future Work

Overall, this work provided a successful proof-of-concept application of compressed sensing to the vision correction problem. In particular, we demonstrated how to model vision correction in the compressive deconvolution framework and verified that the method could work in practice. However, in order for compressive deconvolution to be useful, it must work in real-time and be able to correct many types of aberrations.

7.1   Speed

In practice, users of a vision correcting display would not be viewing a static image. Instead, they would be scrolling or interacting with their device, causing the image shown on the screen to be constantly changing. Videos are typically played at rates of 24 fps or 30 fps, so the compressive prefiltering algorithm needs to take anywhere between 0.03 s to 0.04 s to run on a standard display. This is orders of magnitude faster than the current implementation. There are two main bottlenecks in the algorithm: constructing the matrix H and solving the optimization problem. Constructing H can largely be parallelized, so this would be the first step towards reducing the computation time. The second step would be to devise and implement optimization routines which are faster than ADMM. For example, Mattingley and Boyd in [15] showed that they could solve convex optimization problems at rates of over 100 Hz. Another approach would be to apply a closed-form update to the prefiltered image that utilizes the difference between the new frame and the current frame. If this is possible, then we could limit the number of times we need to compute the full optimization problem.

7.2   Modeling

In this work, we tried three different models of the sensing problem.

1. Using the forward optimization matrix

2. Using the lightfield projection matrix

3. Using the lightfield blur matrix

While the usage of the forward optimization and lightfield projection matrices is well-explored, the use of the lightfield blur matrix is not. In particular, the optimization problem for lightfield blur compressed sensing still needs to be solved. Moreover, further investigation is required to determine if the optimal target lightfield $\mathbf{y}_{LF}$ can be determined solely from the target image $\mathbf{y}$ and the user's visual aberrations.

Outside of deciding how to model the relationship between the display and retina, this work also chose to use the SRM as the sensing matrix. This choice was ideal for a proof-of-concept, but impractical since it requires computing the full H matrix, neglecting the reason for performing compressed sensing in the first place. Therefore, practical implementations will require the sensing matrix Φ to be one such that ΦH is easy to compute. If H is the Lightfield Blur matrix, then techniques from coded aperture imaging, which were used in [2], could be helpful since it only requires applying a random mask over the aperture. Further investigation is also needed in choosing a Φ which minimizes the grain present in the reconstructed retinal image.

7.3 Higher Order Aberrations

The final shortcoming of the current implementation is that it only works for defocus blur. Although defocus blur is what afflicts most people with ocular aberrations, there are other aberrations which need to be accounted for in order to have a complete vision correcting display. Accounting for these aberrations will require modifying the matrix H which models the relationship between the display and the retina. Modifying this matrix would also requiring revisiting the light transport equations presented in chapter 3.

Chapter 8

Conclusions

Previous approaches to the vision correction problem demonstrate a tradeoff between speed and accuracy. Unfortunately, practical implementations require strong performance along both of these dimensions. This work attempts to balance the tradeoff by looking at the vision correction problem through the lens of compressed sensing. Compressed sensing has theoretical guarantees for perfectly reconstructing signals from very few measurements by leveraging sparsity and nonlinear recovery techniques. Others have used ideas from compressed sensing to solve deconvolution problems of a similar nature to vision correction. Taking inspiration from these compressive deconvolution approaches in other domains, this work models the vision correction problem in the compressive deconvolution framework. We also provide a proof-of-concept implementation which demonstrates the efficacy of the theory. In demonstrating the efficacy of the theory, we compare two different modeling approaches — mapping the display pixels to the retinal pixels (forward optimization), and mapping the display lightfield to the retinal pixels (lightfield projection) — and propose a third modeling approach: mapping the display lightfield to the retinal lightfield (lightfield blur). The lightfield-based modeling approaches produce the most accurate results, though more work needs to be done to determine how to effectively use the lightfield blur matrix for high levels of blur. Despite the successes of this work, in order to be used practically, the algorithms we present need to be made faster, which can be done through existing optimization routines. Therefore, with some improvement, the theory presented in this work is well-primed to help correct the vision of millions of people with visual aberrations.

Bibliography

[1] Galen Andrew and Jianfeng Gao. "Scalable Training of L1-Regularized Log-Linear Models". In: *Proceedings of the 24th International Conference on Machine Learning*. ICML '07. Corvalis, Oregon, USA: Association for Computing Machinery, 2007, pp. 33–40. ISBN: 9781595937933. DOI: 10.1145/1273496.1273501. URL: https://doi.org/10.1145/1273496.1273501.

[2] S. Derin Babacan et al. "Compressive Light Field Sensing". In: *IEEE Transactions on Image Processing* 21.12 (2012), pp. 4746–4757. DOI: 10.1109/TIP.2012.2210237.

[3] Sohail Bahmani and Justin Romberg. "Compressive Deconvolution in Random Mask Imaging". In: *IEEE Transactions on Computational Imaging* 1.4 (2015), pp. 236–246. DOI: 10.1109/TCI.2015.2485941.

[4] Stephen Boyd et al. "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Foundations and Trends® in Machine Learning* 3.1 (2011), pp. 1–122. ISSN: 1935-8237. DOI: 10.1561/2200000016. URL: http://dx.doi.org/10.1561/2200000016.

[5] Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. "A Limited-Memory Algorithm for Bound Constrained Optimization". In: *SIAM Journal on Scientific Computing* (1994).

[6] Emmanuel Candè s and Justin Romberg. "Sparsity and incoherence in compressive sampling". In: *Inverse Problems* 23.3 (Apr. 2007), pp. 969–985. DOI: 10.1088/0266-5611/23/3/008. URL: https://doi.org/10.1088%2F0266-5611%2F23%2F3%2F008.

[7] E.J. Candes and T. Tao. "Decoding by linear programming". In: *IEEE Transactions on Information Theory* 51.12 (2005), pp. 4203–4215. DOI: 10.1109/TIT.2005.858979.

[8] Emmanuel J. Candes and Michael B. Wakin. "An Introduction To Compressive Sampling". In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 21–30. DOI: 10.1109/MSP.2007.914731.

[9] Zhouye Chen, Adrian Basarab, and Denis Kouamé. "Compressive Deconvolution in Medical Ultrasound Imaging". In: *IEEE Transactions on Medical Imaging* 35.3 (2016), pp. 728–737. DOI: 10.1109/TMI.2015.2493241.

[10] Thong T. Do et al. "Fast and Efficient Compressive Sensing Using Structurally Random Matrices". In: *IEEE Transactions on Signal Processing* 60.1 (2012), pp. 139–154. DOI: 10.1109/TSP.2011.2170977.

[11] Fu-Chung Huang. "A Computational Light Field Display for Correcting Visual Aberrations". PhD thesis. EECS Department, University of California, Berkeley, Dec. 2013. URL: `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-206.html`.

[12] F. Huang et al. "Eyeglasses-free Display: Towards Correcting Visual Aberrations with Computational Light Field Displays". In: *ACM Trans. Graph. (Proc. SIGGRAPH)* 33.4 (2014), pp. 1–12.

[13] Chia-Kai Liang, Yi-Chang Shih, and Homer H. Chen. "Light Field Analysis for Modeling Image Formation". In: *IEEE Transactions on Image Processing* 20.2 (2011), pp. 446–460. DOI: `10.1109/TIP.2010.2063036`.

[14] Michael Lustig et al. "Compressed Sensing MRI". In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 72–82. DOI: `10.1109/MSP.2007.914728`.

[15] John Mattingley and Stephen Boyd. "Real-Time Convex Optimization in Signal Processing". In: *IEEE Signal Processing Magazine* 27.3 (2010), pp. 50–61. DOI: `10.1109/MSP.2010.936020`.

[16] Justin Romberg. "Imaging via Compressive Sampling". In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 14–20. DOI: `10.1109/MSP.2007.914729`.

[17] Zehao Wu. "Investigating Computational Approaches and Proposing Hardware Improvement to the Vision Correcting Display". MA thesis. EECS Department, University of California, Berkeley, May 2016. URL: `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-67.html`.

[18] Yirong Zhen. "New Algorithms for the Vision Correcting Display". MA thesis. EECS Department, University of California, Berkeley, May 2019.

Appendix A

Lightfield Projection Matrix Construction

The lightfield projection matrix was introduced by Fu Chung Huang et. al in [12]. It models the relationship between the display lightfield and the retinal image by discretizing the lightfield projection equation (eq. (3.3)). First, light rays are uniformly sampled from the aperture to the retina. These rays are parameterized using the "in-camera" parameterization from [13]. Then, these rays are transported back to the display using the light transport matrix M. Rays which fall outside of the display are discarded. Each light ray is assigned a retinal index i which corresponds to the retinal pixel from where it began. Each ray is also assigned a display index j which corresponds to the display location and angular view it originated from. In other words, each j maps to a unique x, y, u, v coordinate in the two-plane parameterization of the display lightfield. Suppose the lightray lands corresponds to x, y, u, v on the display. Algorithm 1 produces a pixel index (x_d, y_d) and angular view index (u_d, v_d) which can be mapped to an index j. Note that a refers to the number of pixels per macropixel, and p is the screen pixel pitch. These formulas can be derived geometrically by considering how display pixels correspond to each angular view, as shown in fig. A.1. Once
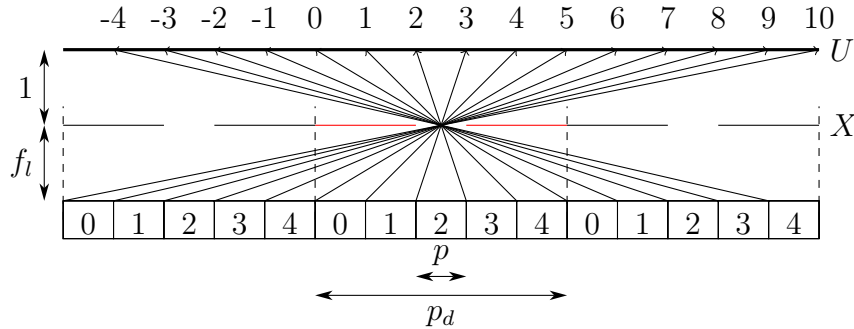

Figure A.1: Angular Boundaries of Central Macropixel on Lightfield Display

---

Algorithm 1: Assign Display Pixels

Data: $x, y, u, v \in \mathbb{R}$

Data: $x_d, y_d, u_d, v_d \in \mathbb{Z}$

$x_d \leftarrow \lfloor \frac{x}{ap} \rfloor$;

$y_d \leftarrow \lfloor \frac{y}{ap} \rfloor$;

$u_d \leftarrow \text{nearest\_angular\_boundary}(u)$;

$v_d \leftarrow \text{nearest\_angular\_boundary}(v)$;

if $u_d \geq 0$ **then**

$\quad \mid \quad x_d \leftarrow x_d - \lfloor \frac{u_d}{a} \rfloor$;

else

$\quad \lfloor \quad x_d \leftarrow x_d - \lfloor \frac{-u_d}{a} \rfloor$;

if $v_d \geq 0$ **then**

$\quad \mid \quad y_d \leftarrow y_d - \lfloor \frac{v_d}{a} \rfloor$;

else

$\quad \lfloor \quad y_d \leftarrow y_d - \lfloor \frac{-v_d}{a} \rfloor$;

$u_d \leftarrow (a - 1) - ((u_d - 1) \mod a)$;

$v_d \leftarrow (a - 1) - ((v_d - 1) \mod a)$;

---

each ray is mapped to an ij pair, the ijth entry in the matrix is the number of rays which map to that ij pair. The ith row of the matrix is normalized by the number of rays hitting the ith retinal pixel.

Appendix B

Forward Optimization Matrix Construction

The forward optimization matrix was introduced in [17]. It models the relationship between the display pixels and retinal pixels. The parameters to the algorithm are given in table B.1. The first step of the algorithm is to sample rays from each pixel to the pinhole mask. The

Parameter	Description
d_0	Distance from the display to the eye
f_l	Distance between the screen and the pinhole mask
a	Number of pixels per pinhole (angular views)
p	Screen Pixel Pitch
n	Near point distance of the eye
f	Focal length of the eye
d_i	Length of the eye

Table B.1: Parameters for Forward Optimization

forward optimization matrix assumes that each pixel is only visible through a single pinhole, so it only samples rays to the nearest pinhole. Suppose the ray starts at position x_d. Then the position of the center of the nearest pinhole u_d is found by tracing a ray to the center of the lens and determining which macropixel that lightray passes through.

$$u_d = \left(\left\lfloor \frac{x_d * \left(1 - \frac{f_l}{d_o} \right)}{ap} \right\rfloor + 0.5 \right) * ap \tag{B.1}$$

This is illustrated by fig. B.1. From the pinhole mask, the light ray then travels to the lens and hits it at point x_l. The lens, which has focal length f, refracts the ray onto the
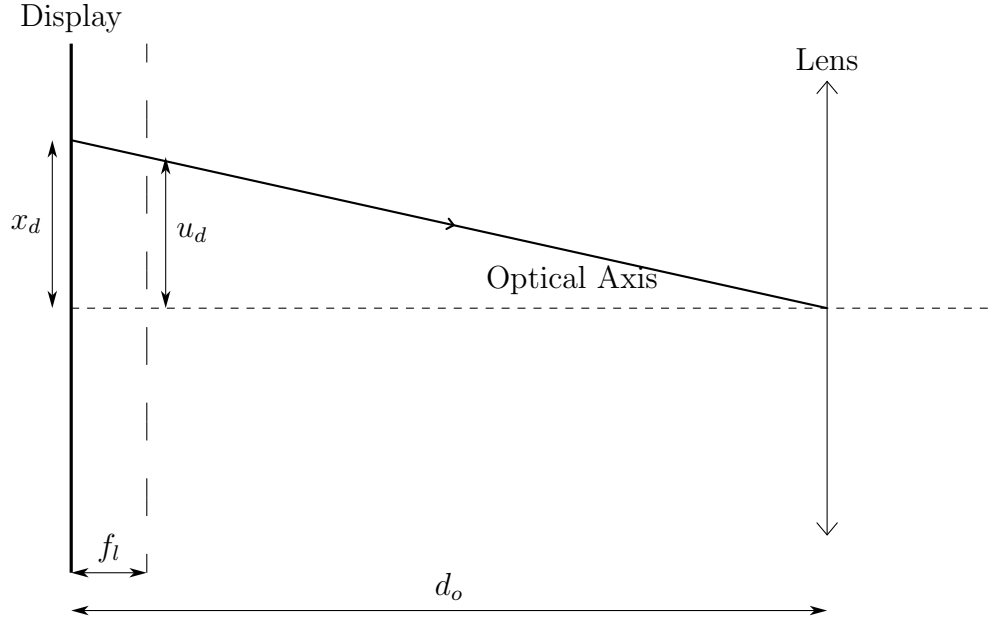
Figure B.1: Pinhole Selection in Forward Optimization

retina, hitting the point x_r. The focal plane is the plane at which light rays originating from the plane land on the retina. The position of the focal plane is equal to the near point distance of the eye, n, which is the closest point the user's eye is able to focus on. Figure B.2 demonstrates the ray-tracing process. The equations which describe the ray tracing process are derived from fig. B.2 and are given by eq. (B.2).

$$
\begin{aligned}
x_f &= x_d - \frac{u_d - x_d}{f_l} \cdot (n - d_o) \\
x_l &= x_f + \frac{u_d - x_d}{f_l} \cdot n \\
x_r &= -\frac{x_f d_i}{n}
\end{aligned}
\tag{B.2}
$$

After tracing each ray, rays which hit the lens outside the aperture are removed. Next, count the number of rays which originate at the jth display pixel and end at the ith retinal pixel. This gives the ijth entry in the matrix. Each row of the matrix is normalized by the total number of rays hitting the ith retinal pixel.
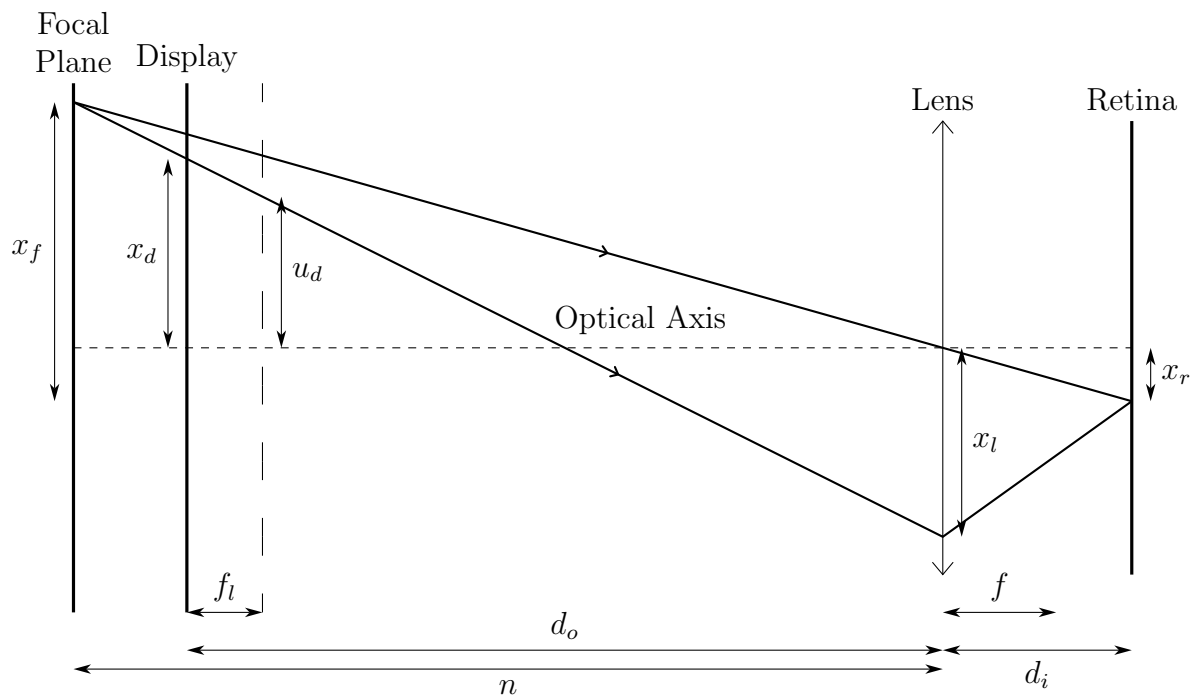
Figure B.2: Forward Optimization Ray Diagram

Appendix C

Lightfield Blur Matrix Construction

The lightfield blur matrix is constructed in the same manner as which the lightfield projection matrix is constructed. The only difference is that when determining the index i, we not only consider the spatial location that the light ray hits on the retina, but also the angular location on the retina that it hits. This is accomplished by sub-dividing the retina into a^2 equally sized segments, where a is the number of angular views. Once the retinal indices are assigned, the rest of the Lightfield Blur matrix construction is identical to the ligthfield projection matrix construction.