Mathematical Techniques for Image Interpolation

Todd Wittman

Department of Mathematics

University of Minnesota

Submitted for completion of the Mathematics Department oral exam.

Abstract

We discuss the problem of interpolating visually acceptable images at a higher resolution. We first present the interpolation problem and why linear interpolation filters are inadequate for image data. To represent the major mathematical approaches to image processing, we discuss and evaluate five different image interpolation methods. First, we present a PDE-based method derived from the anisotropic heat equation. Next, we discuss the extension of Mumford-Shah inpainting to image interpolation. A wavelet-based method that detects edges based on correlations across sub-bands is discussed. To represent the machine learning community, we discuss a method inspired by Locally Linear Embedding (LLE) that interpolates image patches by finding close matches in a training set. Finally, we present a novel statistical filter based on Non Local (NL) Means denoising that interpolates and removes noise by using global image information.

1. Introduction

A digital image is not an exact snapshot of reality, it is only a discrete approximation. This fact should be apparent to the average web surfer, as images commonly become blocky or jagged after being resized to fit the browser. The goal of image interpolation is to produce acceptable images at different resolutions from a single low-resolution image. The actual resolution of an image is defined as the number of pixels, but the effective resolution is a much harder quantity to define as it depends on subjective human judgment and perception. The goal of this paper is to explore different mathematical formulations of this essentially aesthetic quantity.

The image interpolation problem goes by many names, depending on the application: image resizing, image upsampling/downsampling, digital zooming, image magnification, resolution enhancement, etc. The term super-resolution is sometimes used, although in the literature this generally refers to producing a high-resolution image from multiple images such as a video sequence. If we define interpolation as "filling in the pixels in between," the image interpolation problem can be viewed as a subset of the inpainting problem (see Figure 1.1).

The applications of image interpolation range from the commonplace viewing of online images to the more sophisticated magnification of satellite images. With the rise of consumer-based digital photography, users expect to have a greater control over their digital images. Digital zooming has a role in picking up clues and details in surveillance images and video. As high-definition television (HDTV) technology enters the marketplace, engineers are interested in fast interpolation algorithms for viewing traditional low-definition programs on HDTV. Astronomical images from rovers and probes are received at an extremely low transmission rate (about 40 bytes per second), making the transmission of high-resolution data infeasible [2]. In medical imaging, neurologists would like to have the ability to zoom in on specific parts of brain tomography images. This is just a short list of applications, but the

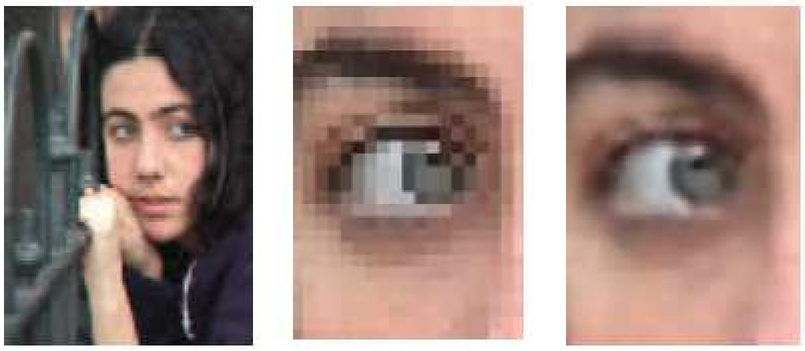

Figure 1.1: Image interpolation using Navier-Stokes inpainting. Left: original image. Center: close-up of eye in image. Right: interpolated image. Taken from [1].

wide variety cautions us that our desired interpolation result could vary depending on the application and user.

1.1. **The Image Interpolation Problem.** In this section, we will establish the notation for image interpolation used throughout the paper. Suppose our image is defined over some rectangle $\Omega \subset \Re^2$. Let the function $f : \Omega \to \Re$ be our ideal continuous image. In an abstract sense, we can think of f as being "reality" and Ω as our "viewing window." Our observed image u_0 is a discrete sampling of $f$at equally spaced points in the plane. If we suppose the resolution of u_0 is $\delta x \times \delta y$, we can express u_0 by

$$u_0(x, y) = C_{\delta x,\delta y}(x, y)f(x, y), \quad (x, y) \in \Omega \tag{1}$$

where C denotes the Dirac comb:

$$C_{\delta x,\delta y}(x, y) = \sum_{k,l \in Z} \delta(k\delta x, l\delta y), \quad (x, y) \in \Re^2.$$

The goal of image interpolation is to produce an image u at a different resolution $\delta x' \times \delta y'$. For simplicity, we will assume that the Euclidean coordinates are scaled by the same factor K:

$$u(x, y) = C_{\frac{\delta x}{K},\frac{\partial y}{K}}(x, y)f(x, y), \quad (x, y) \in \Omega. \tag{2}$$

Given only the image u_0, we will have to devise some reconstruction of f at the pixel values specified by this new resolution. We will refer to K as our zoom or magnification factor. Obviously, if $K = 1$ we trivially recover u_0. The image u_0 is upsampled if K¿1 and downsampled if K¡1. In this paper, we will focus on the upsampling case when K¿1 is an integer.

Let $\Omega_K \subset \Omega$ denote the lattice induced by (2) for a fixed zoom K. Note that the lattice of the original image u_0 in (1) is Ω_1. Also note that for infinite magnification we obtain $\Omega_K \to \Omega$ as $K \to \infty$. For computation purposes, we can shift the lattices to the positive integers. So if the observed image u_0 is an mxn image,

$$\Omega_K = [1, 2, \dots, Km] \times [1, 2, \dots, Kn], \quad K \in Z_+.$$

Many interpolation techniques impose the constraint $\Omega_1 \subseteq \Omega_K$. In this case, only a subset of the pixels in Ω_K needs to be determined and the interpolation problem becomes a version of the inpainting problem.

Given the notation above, we can state the image interpolation problem succinctly: Given a low-resolution image $u_0 : \Omega_1 \to \Re$ and a zoom K¿1, find a high-resolution image $u : \Omega_K \to \Re$. Obviously, this is an ill-posed problem. We need to impose assumptions on the reconstruction of f in equation (2). The choice of interpolation technique depends on the choice of assumptions. In other words, we need a mathematical understanding of what constitutes our perception of "reality" f.

Interpolation methods differ in their mathematical description of a "good" interpolated image. Although it is difficult to compare methods and judge their output, we propose 9 basic criteria for a good interpolation method. The first 8 are visual properties of the interpolated image, the last is a computational property of the interpolation method.

(1) Geometric Invariance: The interpolation method should preserve the geometry and relative sizes of objects in an image. That is, the subject matter should not change under interpolation.
(2) Contrast Invariance: The method should preserve the luminance values of objects in an image and the overall contrast of the image.
(3) Noise: The method should not add noise or other artifacts to the image, such as ringing artifacts near the boundaries.
(4) Edge Preservation: The method should preserve edges and boundaries, sharpening them where possible.
(5) Aliasing: The method should not produce jagged or "staircase" edges.
(6) Texture Preservation: The method should not blur or smooth textured regions.
(7) Over-smoothing: The method should not produce undesirable piecewise constant or blocky regions.
(8) Application Awareness: The method should produce results appropriate to the type of image and order of resolution. For example, the interpolated results should appear realistic for photographic images, but for medical images the results should have crisp edges and high contrast. If the interpolation is for general images, the method should be independent of the type of image.
(9) Sensitivity to Parameters: The method should not be too sensitive to internal parameters that may vary from image to image.

Of course, these are qualitative and somewhat subjective criteria. We hope to develop a mathematical model of image interpolation and error analysis. In a sense, the methods discussed in this paper each present a mathematical model of these visual criteria.

1.2. **Linear Interpolation Filters.** The simplest approach is to assume that f in equation (2) is reconstructed by a convolution kernel $\varphi : \Re^2 \to \Re$ where

$\int \varphi(x,y)dydx = 1$. Then we can approximate f by

$$f \approx u_0 * \varphi.$$

Substituting this into (2) gives rise to a general linear interpolation filter

$$u(x,y) = C_{\frac{\delta x}{K},\frac{\partial y}{K}}(x,y) \cdot (u_0 * \varphi)(x,y), \quad (x,y) \in \Omega.$$

The simplest linear filters are the bilinear and bicubic interpolation, which assume the pixel values can be fit locally to linear and cubic functions, respectively [3, 4]. Along with simple nearest neighbor interpolation, these two filters are the most common interpolation schemes in commercial software. These methods are easy to code as matrix multiplications of u_0. However, an image contains edges and texture, in other words discontinuities. So the assumptions that pixel values locally fit a polynomial function will produce undesirable results. The bilinear and bicubic interpolation methods may introduce blurring, create ringing artifacts, and produce a jagged aliasing effect along edges (see Figure 1.2). The blurring effects arise from the fact that the methods compute a weighted average of nearby pixels, just as in Gaussian blurring. The aliasing effects arise because the linear filters do not take into consideration the presence of edges or how to reconstruct them.

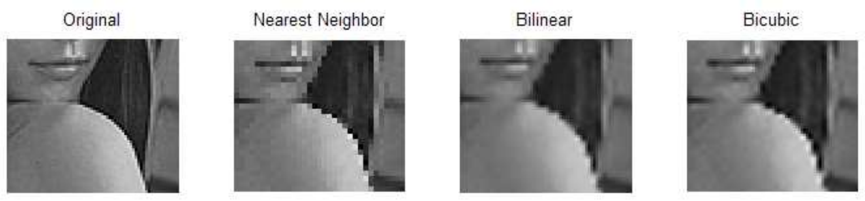

Figure 1.2: Part of Lena image downsampled and then upsampled by factor K=2.

Other linear interpolation filters include include quadratic zoom, the B-spline method, and zero-padding. But these schemes produce the same undesirable effects as the bilinear and bicubic methods, as documented in [5]. Linear filters differ in the choice of φ, which essentially determines how to compute the weighted average of nearby pixels. While this is a natural interpolation scheme for general data sets, this is not necessarily appropriate for *visual* data. In order to improve upon these linear filters, we need to consider interpolation methods that somehow quantify and preserve visual information.

1.3. **Which Methods to Consider?** Generally speaking, mathematical approaches to image processing can be divided into five categories:

1. PDE-Based Methods
(e.g heat diffusion, Perona-Malik, Navier-Stokes, mean curvature)
2. Variations of Energy
(e.g. Total Variation, Mumford-Shah, active contours)
3. Multiscale Analysis
(e.g. wavelets, Fourier analysis, Gabor analysis, Laplacian pyramids)
4. Machine Learning

(e.g. unsupervised learning, data mining, Markov networks)
5. Statistical / Probabilistic Methods
(e.g. Bayesian inference, Natural Scene Statistics, pattern theory)

We are trying to describe the field in broad terms, but not to rank or pigeonhole work in computer vision. Indeed, many techniques such as TV-wavelets inpainting certainly do not fit into one category. Also, these methods differ at the mathematical level, but not necessarily at the conceptual level. For example, some versions of the TV energy can be minimized by solving a PDE or by optimizing a variation of energy.

In our attempt to survey recent work in image interpolation and also display the variety of mathematics used, we will highlight one method from each of the five categories. In the rest of the paper, we will consider

1. A PDE-Based Approach: anisotropic heat diffusion [6]
2. A Variation of Energy Approach: Mumford-Shah inpainting [7]
3. A Multiscale Approach: wavelet-based interpolation [8]
4. A Machine Learning Approach: LLE-based neighbor embeddings [9]
5. A Statistical Approach: NL-means interpolation [10]

These methods are, in some sense, representative of the mathematical approaches to the image interpolation problem and, in a larger sense, to the field of image processing. For example, the heat equation is the most studied PDE in image processing and the Mumford-Shah energy has generated dozens, if not hundreds, of research papers. We will briefly describe the mathematics and motivation behind each method. Then we will present numerical results and discuss each method's advantages and drawbacks.

2. A PDE-Based Approach

A PDE-based approach evolves an image based on a specific driving differential equation. For example, Cha and Kim proposed an interpolation method based on the PDE form of the TV energy [11]. The most famous and well-studied PDE in image processing is the classical heat equation. Anisotropic heat diffusion has been successfully applied to image reconstruction and denoising and its behavior is well-known [12, 13]. Belahmidi and Guichard have proposed an interpolation scheme based on the classical heat diffusion model [6].

2.1. **Heat Diffusion.** The heat equation is a useful tool for smoothing noisy images. We assume that pixel values behave like temperature values and diffuse throughout the image. Diffusion is directed by the unit vectors $\vec{n}$ and $\vec{t}$, which are oriented by the gradient vector Du normal and tangent to the edges, respectively:

$$\vec{n} = \frac{Du}{|Du|} = \frac{\langle u_x, u_y \rangle}{\sqrt{u_x^2 + u_y^2}}, \quad \vec{x} = \frac{Du^{\perp}}{|Du|} = \frac{\langle u_y, -u_x \rangle}{\sqrt{u_x^2 + u_y^2}}.$$

Following the notation of Guichard and Morel [12], an image $u\left(t, \rightharpoonup x\right)$ is evolved according to the PDE

$$\frac{\partial u}{\partial t} = |Du|\, D^2 u\left(\vec{t}, \vec{t}\right) + g\left(|Du|\right) D^2 u\left(\vec{n}, \vec{n}\right) \tag{3}$$

where

$$D^2 u\left(\vec{v}, \vec{v}\right) = \vec{v}^T D^2 u \vec{v} = \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} u_{xx} & u_{xy} \\ u_{yx} & u_{yy} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}.$$

The function $g(s)$ is an "edge-stopping function" satisfying $0 \leqslant g \leqslant 1$ that is close to 0 when s is large and 1 when $s$is small. The most common choice is the Perona-Malik function

$$g(s) = \frac{1}{1 + (s/\lambda)^2}$$

where λ is a parameter set experimentally [13]. The effect of g is shown in the following theorem, which can be shown by direct calculation.

<u>Theorem 2.1</u> When $g \equiv 1$, equation (3) reduces to the Laplacian

$$\frac{\partial u}{\partial t} = \Delta u. \tag{4}$$

When $g \equiv 0$, equation (3) reduces to mean curvature motion

$$\frac{\partial u}{\partial t} = |Du|\, div\left(\frac{Du}{|Du|}\right) = |Du|\, curv(u). \tag{5}$$

In smooth regions, Du is small, $g$is close to 1, and the two terms of (3) have equal weight. The Laplacian of equation (4) will blur the image evenly by isotropic diffusion. Near edges, Du is large, g is close to 0, and the diffusion will occur along edges, smoothing the level lines but preserving the sharpness of the edges.

Belahmidi and Guichard adapted the heat equation (3) to image interpolation by adding a fidelity term [6]. The heat equation will still smooth the image while preserving edges, but the addition of a third term keeps the image u close to the original image u_0. The PDE and initial condition are

$$\begin{array}{l} \frac{\partial u}{\partial t} = |Du|\, D^2u\left(\vec{t}, \vec{t}\right) + g\left(|Du|\right) D^2u\left(\vec{n}, \vec{n}\right) - Pu + Zu_0 \\ u\left(0, \vec{x}\right) = Zu_0 \end{array} . \tag{6}$$

The operator $Z : \Omega_1 \to \Omega_K$ is the duplication zoom or nearest neighbor upsampling technique. The upsampled coarse image Zu_0 acts as the initialization. The projection operator P computes the average of the image u over the $K \times K$ stencil used in the upsampling Z. If we let $N(\vec{x})$ denote the $K \times K$ upsampling window containing pixel $\vec{x}$, we can write P as

$$P(\vec{x}) = \frac{1}{K^2} \int_{N(\vec{x})} u(\vec{y}) d\vec{y}.$$

The classical heat diffusion (3) has been well-studied, but it is unclear how the addition of the fidelity term in (6) affects the equation. Little is known about solution to the PDE (6), although some comments can be made in the viscosity framework. Writing $H\left(x, u, Du, D^2u\right)$ for the right-hand side of equation (6), a viscosity solution u satisfies $u = 0$ on $\partial\Omega_K$ and for all $v \in C^2\left(\Omega_K\right)$ we have:

i.) $H\left(x_0, u, Du, D^2u\right) \leqslant 0$ whenever $u - v$ has a local maximum at (x_0, t_0)

ii.) $H\left(x_0, u, Du, D^2u\right) \leqslant 0$ whenever $u - v$ has a local minimum at (x_0, t_0).

Under this definition, Belahmidi proved the following theorem in [14].

<u>Theorem 2.1</u> Suppose $g(s)$ is the Perona-Malik function and $u_0 \in C\left(\Omega_1\right)$. Then the PDE (6) with boundary condition $u = 0$ on $\partial\Omega_K$ admits a unique viscosity solution.

The proof is similar to the proof for viscous solutions to the Hamilton-Jacobi equation [15, p. 547]. Of course, this is of limited usefulness for natural images because the original image u_0 is almost certainly not continuous.

2.2. **Numerical Results.** Equation (6) can be discretized in a straightforward manner using finite differences. For choice of small time step Δt, we can write

$$u_{ij}^{n+1} = u_{ij}^{n} + \Delta t \left(|Du| \, D^2 u \left(\vec{t}, \vec{t} \right) + g \left(|Du| \right) D^2 u \left(\vec{n}, \vec{n} \right) - Pu + Zu_0 \right)_{ij}.$$

A von Neumann analysis of the 2D heat equation $u_t = \Delta u$ shows that we require $\frac{\Delta t}{(\Delta x)^2} < \frac{1}{4}$ to guarantee stability of an Euler numerical scheme [16]. Using this as a guideline, an image has spatial step $\Delta x = 1$ so we expect a rough upper bound $\Delta t < 0.25$. We used Neumann boundary conditions at the borders of the image.

Belahmidi and Guichard make a heuristic argument for the stopping time T. Running the heat equation on an image u at scale t is equivalent to convolution with a Gaussian kernel of standard deviation $\sqrt{2t}$. Since the length of a diagonal of a pixel's upsampled $K \times K$ window is $\sqrt{2}K$, [6] argues that the desired standard deviation should be $\sqrt{2}K$. So we set the stopping time $T = K^2$. Our experiments with the PDE-based method indicate that the image does not change much after this stopping time, so the image may have reached its steady-state by this time.

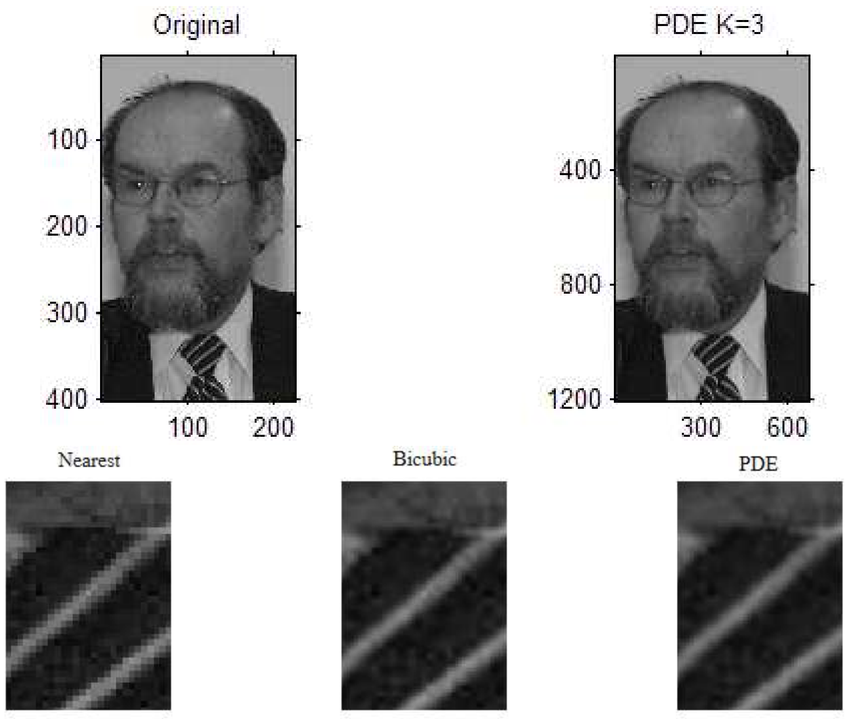

Figure 2.1: PDE-based upsampling with zoom K=3 and time step $\Delta t = 0.1$. Top Row: Magnification of Miller image at time T=9. Bottom Row: Close-up of section of image and comparison to linear filters.

The zooming method seems to do a good job smoothing edges, while maintaining the sharpness of the edges. In terms of aliasing edges, it seems to perform better than linear interpolation filters (see Figure 2.1). The PDE-based method seems to perform well on natural images, although some textures are over-smoothed.

If the parameter λ in the Perona-Malik function $g(s)$is set too small, the method will over-smooth textured regions, resulting in unrealistic images. We set λ very large to avoid this side-effect. This preserved textures, but it also preserved noise and ringing effects present in the original image (see Figure 2.2). Another side-effect, which is barely visible in the figure below, is that the PDE-based method changes the overall contrast of the image. This is because the diffusion across edges is limited, but still occurs. This may be undesirable side-effect in some applications, such as medical images where the gray value of brain matter is crucial.

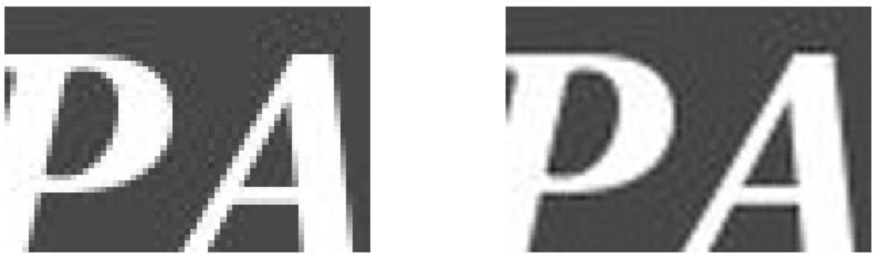
Figure 2.2: PDE-based upsampling of text image with zoom K=5 and time step $\Delta t = 0.1$. Left: original image. Right: zoomed image at time T=25.

3. A Variation of Energy Approach

Briefly, the variational approach to image processing expresses the energy or entropy of an image in terms of a closed-form functional. As in quantum physics, the ideal image should take on the lowest possible energy state. The Rudin-Osher-Fatemi model of the Total Variation (TV) energy has been shown to be effective in smoothing noisy regions while preserving edges in image restoration [17]. Frederic Guichard and Francois Malgouyres have successfully extended the TV model to image interpolation [5], [18]]. The most famous variation of energy, the Mumford-Shah model, has been applied to almost every image processing task [19]. In this section, we will look at its application to inpainting and extension to image upsampling.

3.1. **Mumford-Shah Inpainting.** The Mumford-Shah variation of energy is the most well-studied image model and one of the first attempts to understand vision tasks in a closed mathematical form [19]. Suppose we are given an image domain $\Omega \subset \Re^2$ and the original image $u_0 : \Omega \to \Re$. We let $u : \Omega \to \Re$ denote the new image and $\Gamma \subseteq \Omega$ its corresponding edge set. The Mumford-Shah (MS) energy is

$$E\left[u, \Gamma \left| u_0 \right.\right] = \frac{\lambda}{2} \int_{\Omega} \left(u - u_0\right)^2 dx + \frac{\gamma}{2} \int_{\Omega \backslash \Gamma} \left|\nabla u\right|^2 dx + \alpha H^1\left(\Gamma\right) \tag{7}$$

where H^1 denotes the one-dimensional Hausdorff meausre.

Each of the terms of (7) specifies a driving force used to produce an image u based on given information. The first term measures the fidelity or closeness to the original image u_0 in the least squares sense. The second term measures the smoothness or regularity of the image u away from the edges. The third term is roughly the length of the curve set Γ. These three terms together quantify the energy or entropy of a image / edge set pair. An ideal image u should possess minimum energy E.

The inpainting problem concerns filling in missing or damaged information in an image. The MS energy was adapted to the inpainting problem by Tsai, Yezzi, and Willsky [20] and later extended by Selim Esedoglu and Jackie Shen [7]. Suppose we are given an image domain $\Omega \subset \Re^2$, a mask or domain $D \subset \Omega$ where information is missing, and the original image $u_0 : \Omega \backslash D \to \Re$. Our goal is to generate an image $u : \Omega \to \Re$ with information filled in the domain D. Let $\Gamma \subseteq \Omega$ denote the edge set corresponding to image u. Modifying equation (7) to include the domain D gives us

$$(8) \quad E\left[u, \Gamma \,|u_0, D\right] = \frac{\lambda}{2} \int_{\Omega} 1_{\Omega \backslash D}\left(x\right)\left(u - u_0\right)^2 dx + \frac{\gamma}{2} \int_{\Omega \backslash \Gamma} \left|\nabla u\right|^2 dx + \alpha H^1\left(\Gamma\right).$$

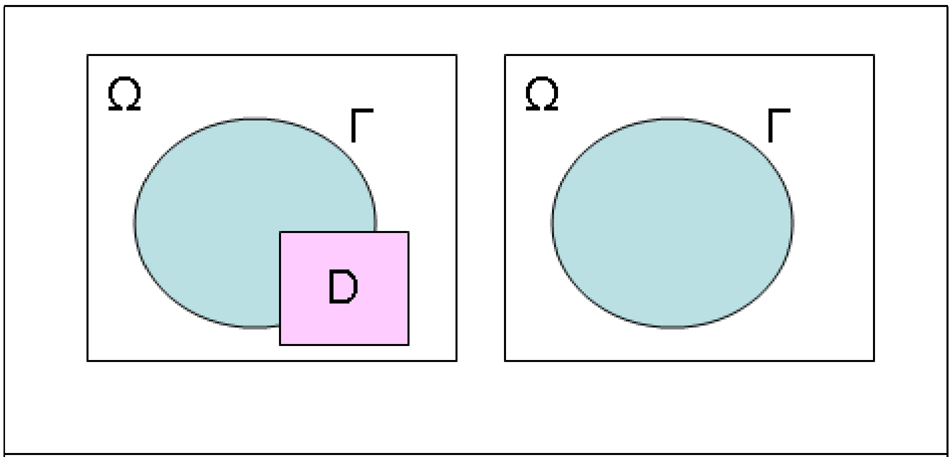

Figure 3.1: Left, image domain with inpainting domain D. Right, desired result.

Esedoglu and Shen showed that equation (8) can be minimized by adapting the Γ-convergence approximation of Ambrosio and Tortorelli [21]. Since calculating the edge set Γ can be problematic, we construct an "edge signature function" $z_\varepsilon : \Omega \to [0, 1]$. For fixed $\varepsilon > 0$, we construct a continuous function z which is 1 almost everywhere except within an ε-neighborhood of Γ, where it equals 0. Rewriting the edge set in terms of z, the energy in (8) becomes

$$(9) \quad E\left[u, z \,|u_0, D, \varepsilon\right] = \frac{\lambda}{2} \int_{\Omega} 1_{\Omega \backslash D}\left(x\right)\left(u - u_0\right)^2 dx + \frac{\gamma}{2} \int_{\Omega} z^2 \left|\nabla u\right|^2 dx + \alpha \int_{\Omega} \left(\varepsilon \left|\nabla z\right|^2 + \frac{\left(1 - z\right)^2}{4\varepsilon}\right) dx.$$

The corresponding Euler-Lagrange equations for (9) are

$$(10) \quad \begin{array}{c} \lambda 1_{\Omega \backslash D}\left(x\right)\left(u - u_0\right) - \gamma \nabla \cdot \left(z^2 \nabla u\right) = 0 \\ \left(\gamma \left|\nabla u\right|^2\right) z + \alpha\left(-2\varepsilon \Delta z + \frac{z - 1}{2\varepsilon}\right) = 0 \end{array}.$$

We can impose Neumann boundary conditions $\frac{\partial u}{\partial \vec{n}} = \frac{\partial z}{\partial \vec{n}} = 0$. Esedoglu and Shen simplify the system in (10) by introducing the differential operators

$$\begin{aligned} L_z &= -\nabla \cdot z^2 \nabla + \tfrac{\lambda 1_{\Omega \setminus D}(x)}{\gamma} \\ M_u &= \left(1 + \tfrac{2\varepsilon\gamma}{\alpha} |\nabla u|^2\right) - 4\varepsilon\Delta \end{aligned} \quad . \tag{11}$$

Then (10) can be written as

$$L_z u = \frac{\lambda 1_{\Omega \setminus D}(x)}{\gamma} u_0 \quad \textit{and} \quad M_u z = 1. \tag{12}$$

This system can be solved using an iterative solver such as Gauss-Jacobi.

The difficulty comes in setting the experimental parameters. We can certainly normalize to assume one of the parameters is one, say $\gamma = 1$. If the parameter λ is too large, the fidelity term will receive more weight and the image will not deviate from u_0. If the parameter α is too large, the length term will receive more weight and edges will consist of straight lines only or the edges will disappear completely. Both terms need to be set relative to $\gamma = 1$ or else the image will be too smooth or too blocky. The Γ-convergence approximation also introduces a parameter ε, although [7] indicates that inpainting will not be very sensitive to this parameter.

As mentioned in Section 1.1, a solution to the inpainting problem can be adapted to the image zooming problem [1]. We simply specify our mask D to be empty pixels between pixels in u_0. For a zoom factor K, adjacent pixels in the original lattice Ω_1 should be separated by K-1 pixels in the finer lattice Ω_K.

3.2. **Numerical Results.** The MS interpolation scheme is very sensitive to the parameters. Figure 3.2 shows an interpolation by factor K=2 using $\alpha = \gamma = \lambda = 1$. The method converges very quickly to an answer. While the edges in the resulting image are sharp, the regions away from edges are over-smoothed and

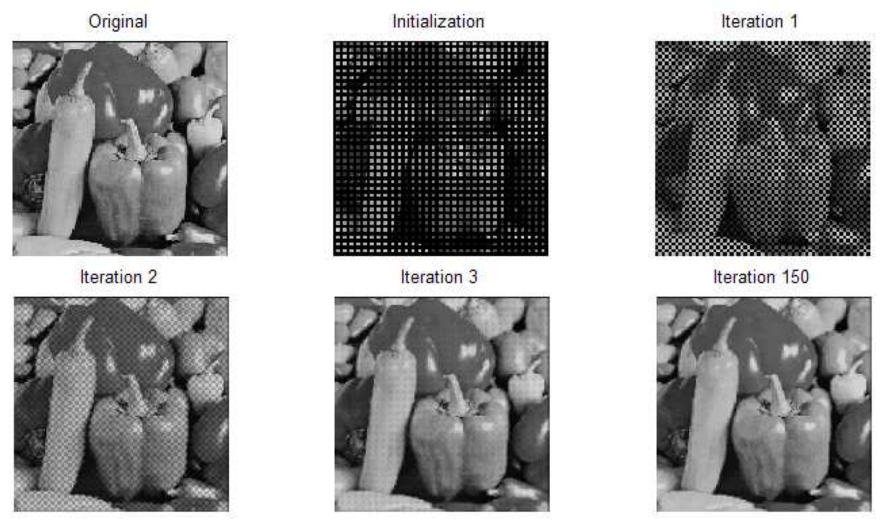

Figure 3.2: Iterations of Mumford-Shah interpolation with zoom K=2.

the textures are lost. Unfortunately, the parameters also affect how quickly the algorithm converges. With these particular values, MS interpolation converged within 10 iterations to a solution.

Figure 3.3 presents results on a high-resolution face image that was downsampled by a factor K=3. We chose the parameters $\alpha = 10, \gamma = \lambda = 1$. MS interpolation blurs the textured region in the shirt and the fine edges of the glasses is less sharp. The edges appear to be less jagged than in nearest neighbor interpolation. In this case, bicubic interpolation seems better at preserving fine structures and textures.

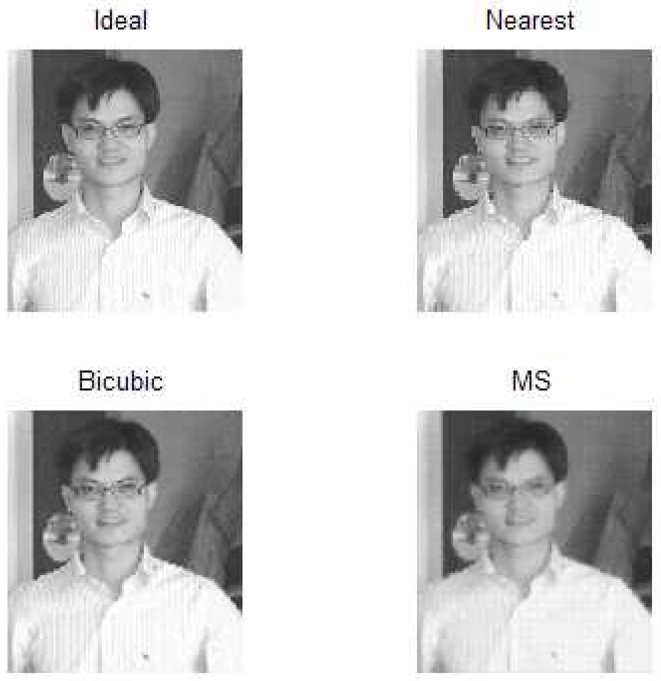

Figure 3.3: MS interpolation with zoom K=3 of Jackie image.

To demonstrate the performance on edges, we ran the MS interpolation on a simple triangle image as in [18]. The nearest neighbor scheme produces a visible staircase effect along the edges. Bicubic interpolation blurs the edges and produces some ringing effects. The result of TV minimization is taken from [18]. The TV image shows some ringing artifacts near the edges and rounds the corners slightly. With the parameter α chosen large, MS interpolation will sharpen the edges. The upper two corners of the triangle are blurred somewhat.

3.3. **Mumford-Shah-Euler Inpainting.** The length term in equation (8) will naturally prefer straight edges. This means that for large enough values of $K$and α, the inpainting procedure may choose to complete edges in large regions with straight lines rather than smooth curves (see Figure 3.5). To correct for this [7]

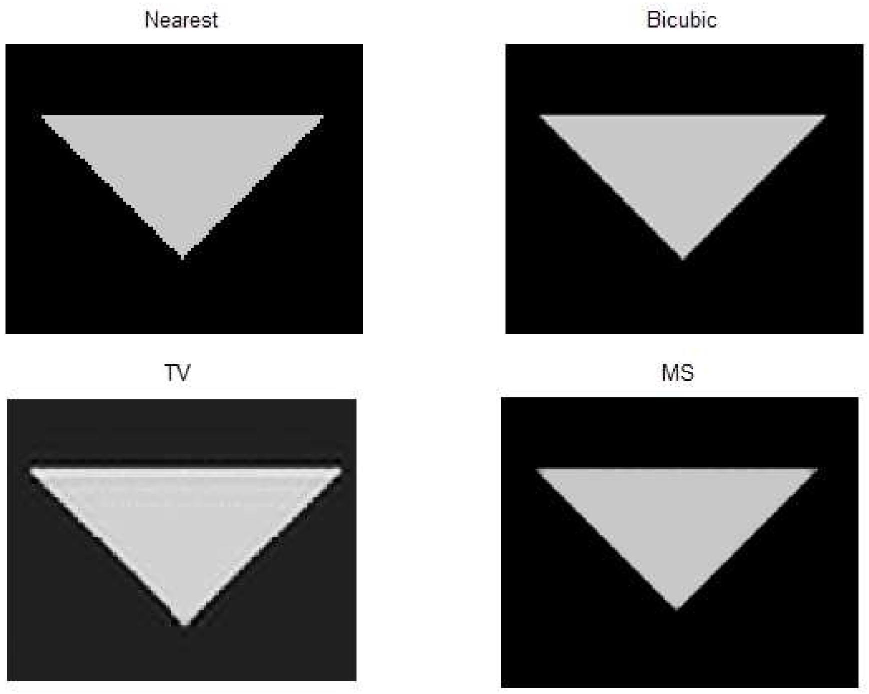

Figure 3.4: Zoom K=4 on triangle image. Bottom-left: TV result taken from [18].

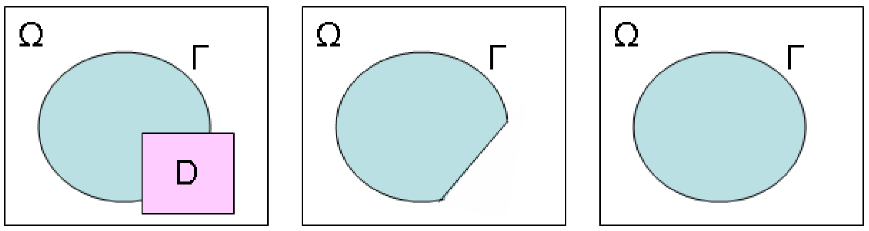

Figure 3.5: Left, image with inpainting domain. Center, result of Mumford-Shah inpainting. Right, result of Mumford-Shah-Euler inpainting.

suggest adding a term measuring the curvature κ of the edges. Esodlu and Shen refer to this as the Euler elastica term, after Leonhard Euler's 1744 work exploring curvature. Combining this with the length term in equation (8) gives rise to the

Mumford-Shah-Euler energy

(13)
$$E\left[u,\Gamma\left|u_0,D\right.\right]=\frac{\lambda}{2}\int_{\Omega}1_{\Omega\backslash D}\left(x\right)\left(u-u_0\right)^2dx+\frac{\gamma}{2}\int_{\Omega\backslash\Gamma}\left|\nabla u\right|^2dx+\int_{\Gamma}\left(\alpha+\beta\kappa^2\right)ds$$

where ds denotes the length differential of the edge set.

De Georgi outlined a Γ-convergence approximation of the Euler elastica term [22]. Esedoglu and Shen showed the effectiveness of this energy in producing visually acceptable images from highly damaged or noisy images. For interpolation, using this modified energy may help prevent aliasing or incorrectly interpolated curves. This is an avenue we hope to explore in the future.

4. A Multiscale Approach

A multiscale approach tries to break an image down into its most basic components of information and express the image in scales of those building blocks. Multiscale analysis seems a natural fit for image interpolation, with image upsampling viewed as determining finer scales of image detail to add to a low-resolution image. Wavelets and their variants have received much attention for image interpolation, although most of the work has focused on image super-resolution: interpolating a high-resolution image from an image sequence rather than a single image. These techniques do not necessarily carry over to single image super-resolution, as the sequence generally contains much more information than a single image. The techniques are also highly dependent on precise sub-pixel registration of the low-resolution images in the sequence [23, 24]. Most of the wavelet-based work on single image interpolation has focused on detecting extrema and singularities in the wavelet transform. In this section, we describe a work by Carey, Chuang, and Hemami that focuses on producing crisp well-defined edges in the interpolant [25].

4.1. **Wavelets and Hölder Regularity.**

Carey et. al. begin by defining the smoothness of an image in terms of Hölder regularity of the wavelet transform. We say that a function $f:\Re\rightarrow\Re$ has Hölder regularity with exponent $\alpha=n+r$, $n\in N,\quad 0\leqslant r<1$ if there exists a constant C satisfying

$$\left|f^{(n)}(x)-f^{(n)}(y)\right|\leqslant C\left|x-y\right|^r\quad x,y\in\Re. \tag{14}$$

Functions with a large Hölder exponent will be both mathematically and visually smooth. Locally, an interval with high regularity will be a smooth region and an interval with low regularity will correspond to roughness, such as at an edge in an image. To extend this concept to edge detection in the wavelet domain, we need a technique for detecting local Hölder regularity from the wavelet coefficients.

Let ψ be a compactly-supported discrete wavelet function, such as a Daubechies wavelet. The discrete wavelet transform is computed by projecting a signal onto translations and dilations of the mother wavelet ψ:

$$\psi_{k,l}(x)=\psi\left(2^kx-l\right)\quad k,l\in Z. \tag{15}$$

The wavelet transform coefficients $w_{k,l}$ at scale k and offset $l$are given mathematically as an inner product with the mother wavelet:

$$w_{k,l}=\left\langle f,\psi_{k,l}\right\rangle. \tag{16}$$

Numerically, these coefficients are computed using a filter bank with a scaling function ϕ appropriate to the mother wavelet. The dyadic wavelet filter bank repeatedly divides the signal at scale k into an approximation signal a_k and a detail signal d_k, also called the averages and differences (see Figure 4.1). The coefficients of d_k are precisely the wavelet coefficients $w_{k,l}$.

The following theorem by Ingrid Daubechies establishes the connection between wavelet coefficients and Hölder regularity [26, p. 300].

Theorem 4.1 Let $x_0 \in \Re$ and S be a set of index pairs (k,l) such that for some $\varepsilon > 0$ we have $(x_0 - \varepsilon, x_0 + \varepsilon) \subset \operatorname{supp}(\psi_{k,l})$. A signal has local Hölder regularity with exponent α in the neighborhood of $(x_0 - \varepsilon, x_0 + \varepsilon)$ if there exists a constant C such that

$$\max_{(k,l)\in S} |w_{k,l}| \leqslant C 2^{-k\left(\alpha+\frac{1}{2}\right)} \quad . \tag{17}$$

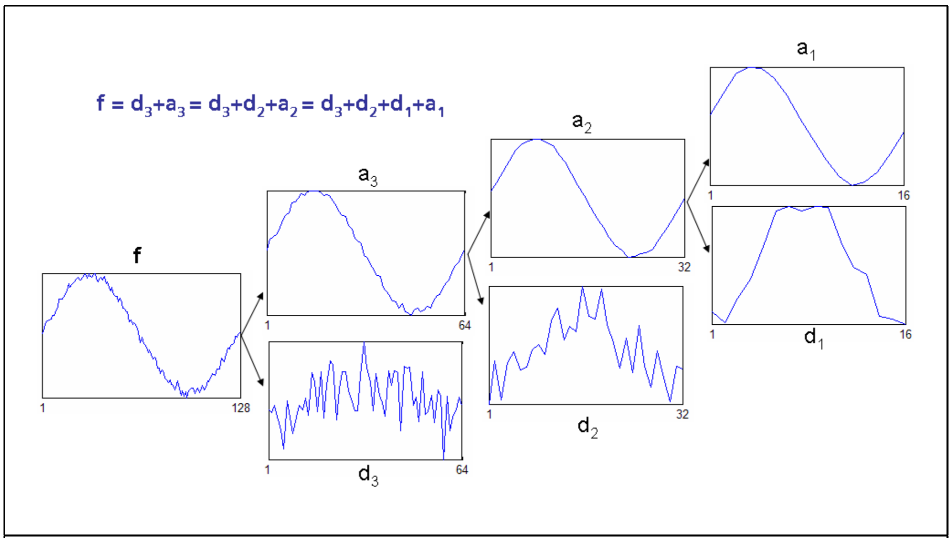

Figure 4.1: Discrete 3-level wavelet decomposition of noisy sine wave signal.

Theorem 4.1 alone is not sufficient for determining the local Hölder regularity, because it requires computation of two unknown constants C and α. It has been observed experimentally that regions in a signal with low regularity tend to have greater similarity across scales. Let $d_m(t)$ and $d_n(t)$ denote the wavelet sub-bands at scales 2^m and 2^n. The correlation between sub-bands is given by

$$Corr\left(d_m(t), d_n(t)\right) = \int_{\Re} d_m(\tau) d_n(\tau - t) d\tau. \tag{18}$$

Applying Theorem 4.1 twice to this definition yields the following theorem.

Theorem 4.2 Let $f : \Re \to \Re$ be C^∞, except possibly in a neighborhood of the origin, where it has Hölder regularity with exponent α. The correlation between sub-bands $d_m(t)$ and $d_n(t)$ satisfies

$$\left|Corr\left(d_m(t), d_n(t)\right)\right| \leqslant C 2^{-(m+n)\left(\alpha+\frac{1}{2}\right)}. \tag{19}$$

Theorem 4.2 shows that regions with high regularity will exhibit low correlation across scales, and vice-versa. In other words, an edge will result in extrema in the wavelet coefficients across several scales, while extrema in smooth regions will not persist across scales (see Figure 4.2).

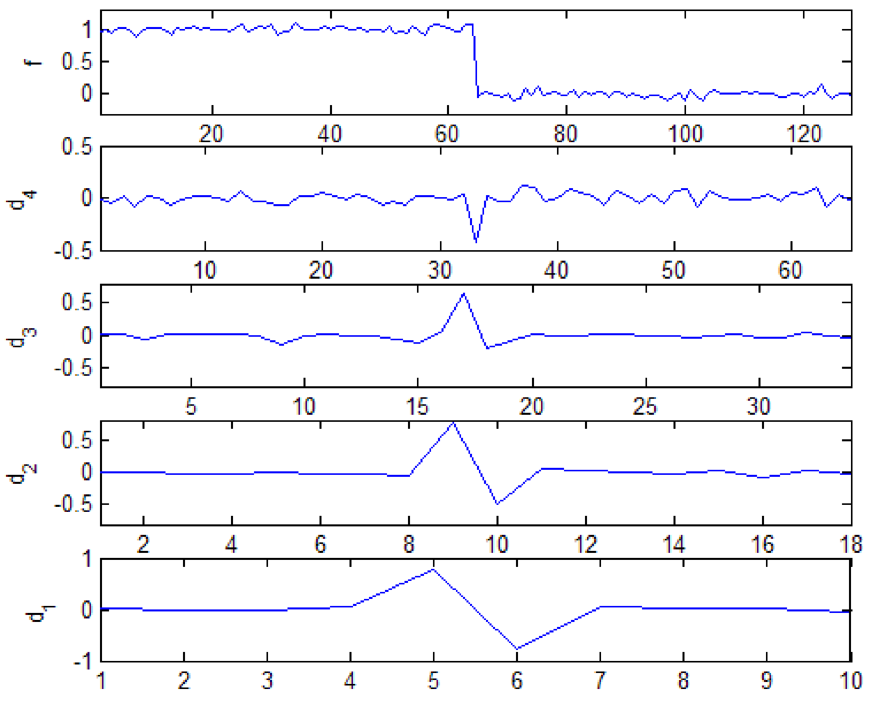

Figure 4.2: 4-level Wavelet decomposition of noisy step function f.

The two previous theorems give a heuristic for estimating the local regularity of a signal by examining the correlation across wavelet sub-bands. Carey et. al. claim that at a strong edge in a signal, the inequalities in both theorems will be close to equality. By (17), in an interval containing a strong edge the logarithm of the maximum coefficient magnitudes should be close to linear across scales. The parameters C and α can then be estimated using equality in (17) and (19).

4.2. **Wavelet-Based Interpolation.** Suppose we are given an image $u_0 : \Omega_1 \longrightarrow \Re$ and its corresponding L-level discrete wavelet decomposition. Synthesizing a new sub-band d_{L+1} will produce a new image $u : \Omega_2 \longrightarrow \Re$ that is twice the size of the original image. Since the theorems above apply to 1D data, Carey et. al. proceed by first processing the image data across each row and appending the signals into a "row-interpolated image." The same processing step is then applied to the columns of this new image, with the end result being u.

Carey et. al. suggest interpolating the sub-band signal d_{L-1} rather than the finest original sub-band d_L because the finest band generally contains too much noise information. As an initialization for d_{L+1}, the detail signal d_{L-1} is upsampled by a factor 4 using cubic spline interpolation. For each subinterval of the signal, the algorithm then determines similarity across scale by computing the linear regression across sub-bands of the maximum coefficient magnitude. If the linear correlation is strong, then the interval should contain an edge and the linear regression will predict the magnitude of the coefficient at sub-band d_{L+1}. The template from $d_{L-1}$is used except at edges, where the signal is modified to achieve equality in (17).

On a small set of test images, [25] demonstrate that their wavelet-based interpolation method results in higher Peak Signal-to-Noise Ratio (PSNR) than the standard bilinear and bicubic methods. However, visually the methods exhibit little difference. The wavelet-based method seems to sharpen the edges, but the textured and smooth regions of the image are blurred. This effect is understood because the interpolation step is simply 1D cubic interpolation, except at strong edges. This method requires the original image u_0 to be large enough to show a significant amount of information across many wavelet scales. Also, the technique lacks resizing flexibility because it assumes the zoom factor K is a multiple of 2.

This wavelet-based method reduces to the linear bicubic filter, except at strong edges. It is best suited for images with strong, well-defined edges separating smooth regions. One possible refinement to this method would be to incorporate textured information. Several papers have demonstrated that wavelet coefficient magnitudes can be used to quantify and classify textures [27, 28]. It would be interesting to incorporate this idea into texture interpolation, resulting in a sharpened image that is visually pleasing as well.

5. A Machine Learning Approach

As image processing research advances, researchers are realizing that details, textures, and other visual nuances of images cannot be expressed in compact mathematical form. In the last few years, researchers have given more attention to machine learning approaches to guide the computer to learn meaningful information from a set of training images. For image interpolation, a set of high-resolution images and its corresponding downsampled versions are provided, with the goal of learning how to connect the low-resolution version to its high-resolution counterpart. William Freeman and his group at Mitsubishi Labs have developed an approach based on Markov networks and belief propagation networks [29]. Bryan Russell, one of Freeman's students, extended this approach by incorporating priors into the belief propagation networks, which results in realistic textured images with sharper edges [30]. Most recently Chang, Yeung, and Xiong developed a learning system inspired by dimensionality reduction techniques, which we highlight below [9].

5.1. **Locally Linear Embedding (LLE).** In the last five years, much attention has been given to mathematical non-linear dimensionality reduction (NLDR) methods, also called manifold learning techniques. Given a high-dimensional data set X, the goal is to interpolate a lower dimensional data set Y that preserves the neighborhoods of the original data set in a geometrically meaningful way. In 2000, Lawrence Saul and Sam Roweis proposed the Locally Linear Embedding (LLE) manifold learning technique [31]. For each data point in X, LLE computes the nearest neighbors and then projects the neighborhood to Y by assuming that the neighborhood is planar. This technique has proven effective experimentally in reducing the dimensionality of data sets in a geometrically meaningful manner (see Figure 5.1).

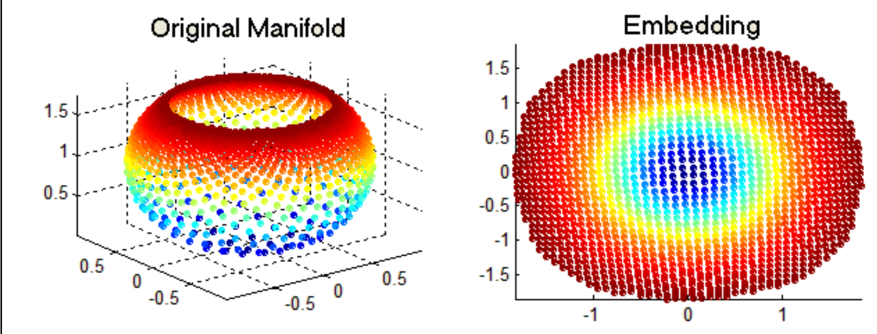

Figure 5.1: LLE dimensionality reduction. Left: original 3D spherical data set. Right: 2D data set computed by LLE.

Once a data set Y is determined, it is possible to add a new data point $\vec{x}$ to the manifold X and add its projection $\vec{y}$ to Ywithout recomputing the entire embedding from X to Y. Saul and Roweis suggest one solution for this out-of-sample extension is to first compute the N nearest neighbors $\{\vec{x}_i\}_{i=1}^N$ of $\vec{x}$ in X. Next we compute normalized weights $\{w_i\}_{i=1}^N$ that best form a linear combination of $\vec{x}$: $\vec{x} \approx \sum_{i=1}^N w_i \vec{x}_i$. Finally, we construct the interpolated point $\vec{y}$ by using these same weights in a linear combination of $\{\vec{y}_i\}_{i=1}^N$, the data points in Y corresponding to the x_i's: $\vec{y} \approx \sum_{i=1}^N w_i \vec{y}_i$. This procedure motivates a machine learning technique for comparing a given image to the training data set. The key difference is that we are given a low-resolution image and interpolate a high-dimensional image, so we need to *increase* the dimensionality of our data points.

5.2. **LLE-Based Interpolation.** Suppose we are given a collection of low-resolution image patches $X = \{\vec{x}_i\}_{i=1}^M$ and their corresponding high-resolution patches $Y = \{\vec{y}_i\}_{i=1}^M$, where images are expressed in raster order as vectors. This training set could be prepared by dividing a set of high-resolution images into patches Y and downsampling the images to patches X. The training set images should be carefully chosen to reflect the textures and patterns that will be seen in the interpolation phase and the downsampling rate should be the desired zoom K. Given a new image patch $\vec{x}$, the goal is to find its corresponding high-resolution image $\vec{y}$.

Inspired by LLE's out-of-sample extension scheme, Chang et. al. propose the following interpolation. For each low-resolution image patch $\vec{x}$, we perform the following steps:

Step 1. Find the N nearest neighbors $\{\vec{x}_i\}_{i=1}^N$ of $\vec{x}$ in the data set X. The metric could be the Euclidean distance, although more sophisticated image difference metrics could be devised.

Step 2. Compute weights $\{w_i\}_{i=1}^N$ that minimize the reconstruction error:

$$\varepsilon = \left\| \vec{x} - \sum_{i=1}^{N} w_i \vec{x}_i \right\|^2 \tag{20}$$

subject to the constraint

$$\sum_{i=1}^{N} w_i = 1.$$

Note that this minimization is only performed over the neighborhood of $\vec{x}$, so we could enforce $w_i = 0$ for any data point $\vec{x}_i$ not in the neighborhood. We can solve this constrained least squares problem by computing a Gram matrix

$$G = \left(\vec{x}\vec{1}^T - A\right)^T \left(\vec{x}\vec{1}^T - A\right)$$

where A is a matrix containing the neighbors $\{\vec{x}_i\}_{i=1}^N$ as its columns. Expressing the weights as a vector $\vec{w}$, the closed form solution of (20) is

$$\vec{w} = \frac{G^{-1}\vec{1}}{\vec{1}^T G^{-1}\vec{1}}. \tag{21}$$

Equivalently, we could solve $G\vec{w} = \vec{1}$ and then normalize the weights $\sum\limits_{i=1}^{N} w_i = 1$.

Step 3. Project $\vec{x}$ to its high-dimensional image patch by computing

$$\vec{y} = \sum_{i=1}^{n} w_i \vec{y}_i$$

where the $\vec{y}_i$'s are the high-dimensional patches corresponding to the $\vec{x}_i$'s.

After completing these steps on each image patch, we have obtained a collection of upsampled image patches which can be arranged into a high-resolution image. However, these patches are not independent since they should form a single image. To help enforce continuity between adjacent patches, the training set $X$is formed by selecting overlapping patches from the training images. Overlapping the image patches is a common trick that is used in many machine learning vision algorithms [29]. Since this still does not guarantee continuity between the computed high-resolution patches, the high-resolution image is constructed by averaging pixel values in the overlapping regions.

Note that if we use the raw pixel values, as suggested above, our method will be sensitive to changes in luminance. That is, if we supply a test image that is brighter than the training images, the first step of the interpolation will not match correct textures to the given image patch. Chang et. al. work around this problem by using the relative luminance changes in their low-resolution patches X. Each pixel in a low-resolution patch is replaced with a 4D feature vector consisting of finite difference approximations of the first and second order gradients. This helps the algorithm find neighbors with similar patterns rather than similar luminances. Since this will prevent us from determining the overall luminance value of the interpolated high-resolution image, the mean luminance value of each high-resolution patch in the training set Y is subtracted from all pixel values. In step 3, the target high-resolution patch $\vec{y}$ is constructed and the mean luminance value of the original low-resolution patch $\vec{x}$ is added to $\vec{y}$.

5.3. **Numerical Results.** We implemented the LLE-based interpolation by selecting a set of high-resolution photographs to form the set Y and downsampling them by a specified zoom factor K to obtain our low-resolution training set X. For our image patch sizes, we used 3x3 windows for the low-resolution images and $3K$x$3K$ windows for the high-resolution images. Using the relative luminance changes as our feature vector, each vector in Xhad length $4 \cdot 3^2 = 36$ and each high-resolution vector in Y had length $9K^2$. The low-resolution patches were selected from the images with an overlap of 1 pixel width between adjacent patches. The high-resolution patches necessarily had an overlap width of K pixels. These are the same window sizes used in [9]. As suggested by the authors, we quadrupled the size of the training set by using rotations of the image patches (0°, 90°, 180°, 270°). This is making use of the assumption that texture patches are often rotationally invariant. One of our training sets consisting of face images in shown in Figure 5.2. The training time is very time-consuming. This particular data set of 5 images took 91 minutes to prepare.

In the interpolation phase, we use N=5 nearest neighbors. Figure 5.3 shows the interpolation result on a face image with zoom K=3. The interpolation phase is rather slow, since the nearest neighbor computation is rather expensive and grows quadratically with the size of the training set. This particular image took roughly 20 minutes to compute. Some aliasing and discretization effects can be seen in the original image and the interpolated image is noticeably smoother. The interpolated image is somewhat blocky however, reflecting the square windows used in the interpolation.

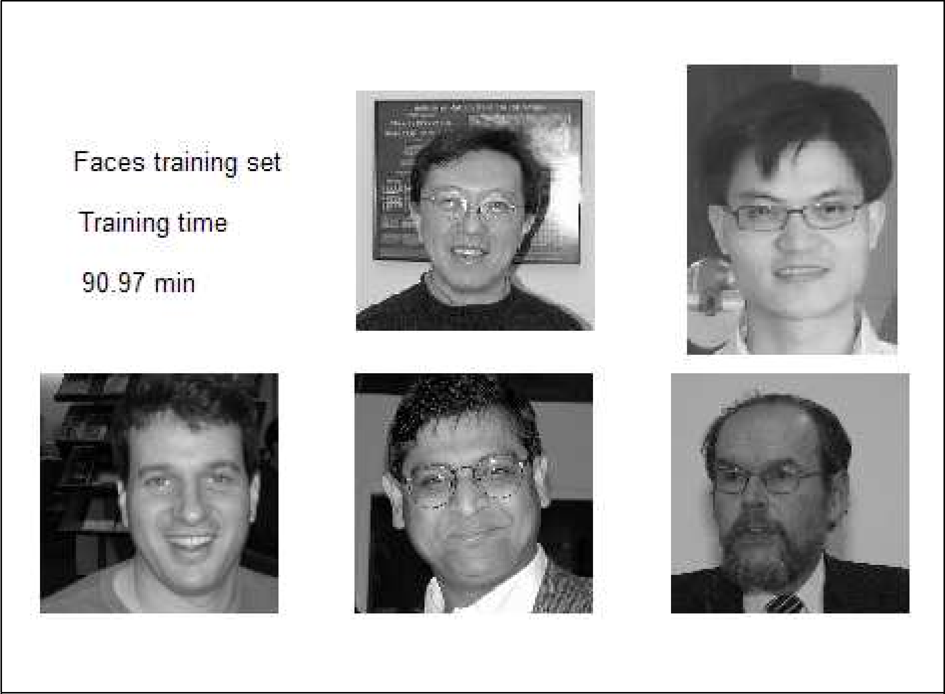

Figure 5.2: Training set used for LLE-based interpolation.

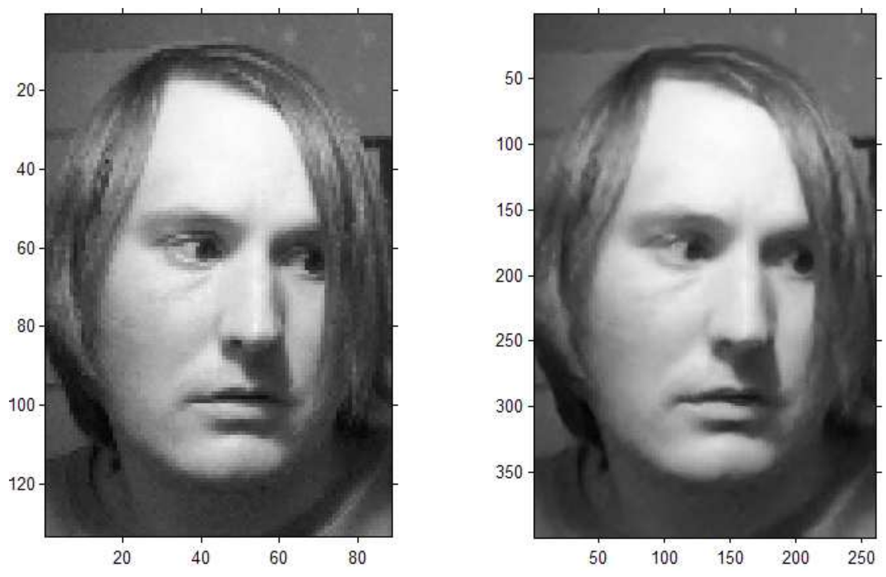

Figure 5.3: LLE-based interpolation of a face image with zoom K=3 and training set shown in Figure 5.2. Left: original image. Right: LLE interpolated image.

If we magnify a piece of the image in Figure 5.3, we can better see the blocky nature of the reconstruction. Figure 5.4 shows a close-up of the eye. The LLE interpolation definitely exhibits some aliasing, whereas the bilinear and bicubic filters smooth the image better. Despite this effect, LLE does seem to do a good job interpolating texture.

The major drawback of LLE interpolation and machine learning methods in general is that we require the generation of a good training set. In this case, the training set should reflect the textures that will be seen in the test image. That is, if we want to interpolate faces then the training set should consist of face images. Figure 5.5 shows the result of interpolating a text image using the face image training set. The text in the image is blurred and the overall contrast of the image is changed.

Not only should the training set reflect the type of image interpolated, but the selected images should also reflect the order of magnitude resolution desired. For example, the texture of a brick wall will change drastically depending on the viewer's distance from the wall. After images are selected, generating the training manifolds X and Y is very time-consuming. This data preparation could be done as a pre-processing step, provided the zoom factor K is known. The downsampling rate and patch sizes depend on K, so this factor must be fixed before training begins. Selecting and preparing a training set requires prior knowledge of the type of image to be interpolated, the resolution of the images, and the desired zoom factor. While

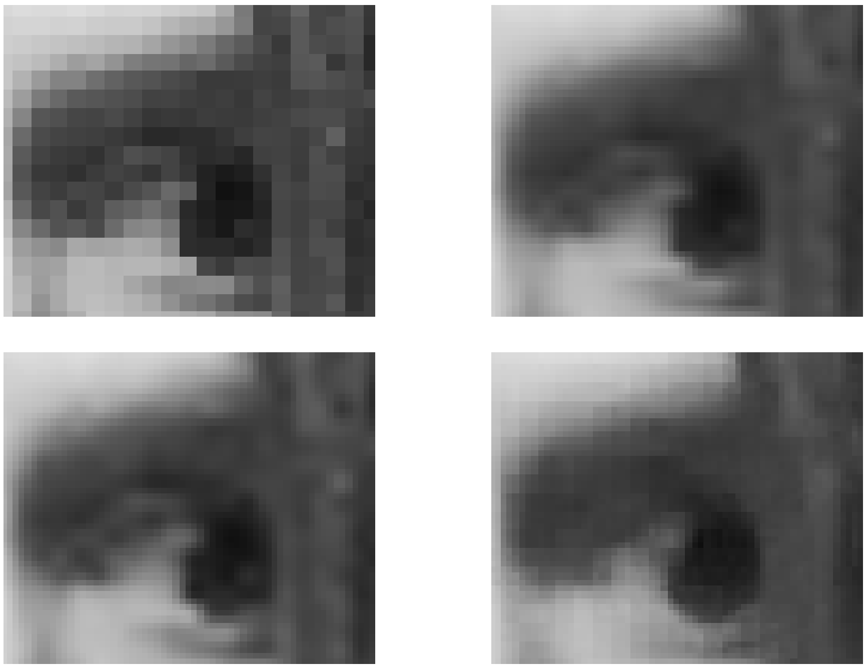

Figure 5.4: Close-up of eye in Figure 5.3 with zoom K=3. Top left, nearest neighbor. Top right, bilinear. Bottom left, bicubic. Bottom right, LLE-based interpolation.

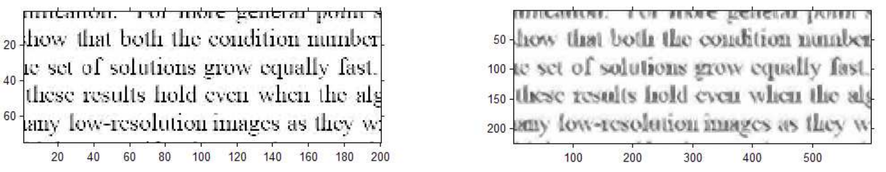
how that both the condition number
e set of solutions grow equally fast.
these results hold even when the alg
any low-resolution images as they w
20
40
60
20 40 60 80 100 120 140 160 180 200
50
100
150
200
100 200 300 400 500

Figure 5.5: Text image interpolated by LLE with zoom K=3. The training set in Figure 5.2 was used. Left: original image. Right: LLE interpolated image.

this information is not generally available beforehand, there are applications in which these parameters are known, such as MR image interpolation.

6. A Statistical Approach

Many, if not all, approaches to image processing can be interpreted as having a statistical or probabilistic motivation. Certainly, the linear interpolation filters mentioned in Section 1.2 are statistical in nature, devising a convolution kernel that produces a weighted sum of neighboring pixels. Several researchers, particularly in psychology, have focused on developing a statistical theory of images and patterns [32] and recent efforts have tried to incorporate this information into interpolation [30]. Interpolating textured images is related to the problem of texture synthesis, which is based on computing local statistics that segment and classify the image textures [33]. Several efforts have been made to develop Bayesian and MAP estimators for constructing a super-resolved image from a sequence of low-resolution images [34]. In this section, we will present a simple statistical filter based on global image statistics that simultaneously denoises and interpolates a textured image.

6.1. **Local vs. Global Interpolation.** All of the previous methods we have discussed thus far are based on local image statistics and properties. The PDE and variational methods are based on very local finite difference calculations. The wavelet interpolation method seeks to detect local singularities in the image. Even LLE-based interpolation uses only a small 3x3 window of the given image as its basis for interpolation, even though this window is compared to other small windows in a large training set. However, textured images will often contain repeatable and identifiable patterns throughout the image.

Although the previous methods preserved edges and structures well, they had a much harder time interpolating texture. Except for LLE interpolation, all the methods tended to over-smooth textured regions. This may be because the PDE, variational, and wavelet methods can be written in a simple closed form, but natural textured images defy compact mathematical explanation. LLE interpolation could only reproduce the texture if the texture was present in the training set at the desired order of resolution.

In summary, we have observed a simple fact: local interpolation schemes do not preserve textures. This motivates the creation of an interpolation scheme based on global image statistics. Conveniently, a statistical filter based on global statistics has already been recently been developed for image denoising. Appropriately, its creators refer to it as the Non Local (NL) filter.

6.2. **NL-Means Denoising.** In a 2004 preprint, J.M. Morel and his students Antoni Buades and Bartomeu Coll proposed a new statistical filter for denoising images that uses the information present in the entire image [10]. Suppose we are given a noisy grayscale image $u_0 : \Omega \rightarrow \Re$. For each pixel $\vec{x} \in \Omega$, we define a local neighborhood $N(\vec{x}) \subseteq \Omega$ as a subset satisfying two simple properties:

(i.) $\vec{x} \in N(\vec{x})$

(ii.) $\vec{x} \in N(\vec{y}) \Rightarrow \vec{y} \in N(\vec{x})$.

There are many possible choices of topology that will satisfy these two properties. Note that a simple NxN window, with N>1 odd, centered over pixel $\vec{x}$ will suffice. Each neighborhood describes the local pattern or texture surrounding $\vec{x}$. If $\vec{x}$ is a noise point, then to determine the proper value of $u_0(\vec{x})$ we should consider the pixel values $u_0(\vec{y})$ surrounded by neighborhoods $N(\vec{y})$ similar to $N(\vec{x})$. Not knowing the pattern of the image a priori, we assume that the image neighborhoods are distributed according to a Gaussian distribution. This gives rise to the NL-means filter:

$$u(\vec{x}) = \frac{1}{Z(\vec{x})} \int_{\Omega} u_0(\vec{y}) \exp\left(-\frac{\| u_0(N(\vec{x})) - u_0(N(\vec{y})) \|_2^2}{h^2} \right) d\vec{y} \tag{22}$$

where $Z(\vec{x})$ is the normalization factor

$$Z(\vec{x}) = \int_{\Omega} \exp\left(-\frac{\|u_0(N(\vec{x})) - u_0(N(\vec{y}))\|_2^2}{h^2}\right) d\vec{y}.$$

The norm in (22) can be any matrix norm, such as the Frobenius norm or any L^p matrix norm. Morel et. al. recommend the L^2 matrix norm.

The filtering parameter h controls the weight pixel values receive and needs to be set carefully. If h is too small, the image u will closely resemble the original image. If h is too large, then pixel values with dis-similar neighborhoods will contribute to the value of $u(\vec{x})$ and the result will resemble Gaussian blurring. Intuitively, h acts like a standard deviation of the neighborhood distribution. Given the Gaussian nature of equation (22), we found experimentally that a good choice is $h = \sqrt{2}\sigma$ where σ is the standard deviation of the pixel values in u_0.

Morel et. al. demonstrated that the NL-means filter successfully denoises textured images. The authors showed that on test cases NL-means outperformed classical denoising methods, including Gaussian smoothing, the Wiener filter, the TV filter, wavelet thresholding, and anisotropic heat diffusion. They showed that under certain assumptions on the noise distribution, NL-means minimizes the additive noise present in the original image.

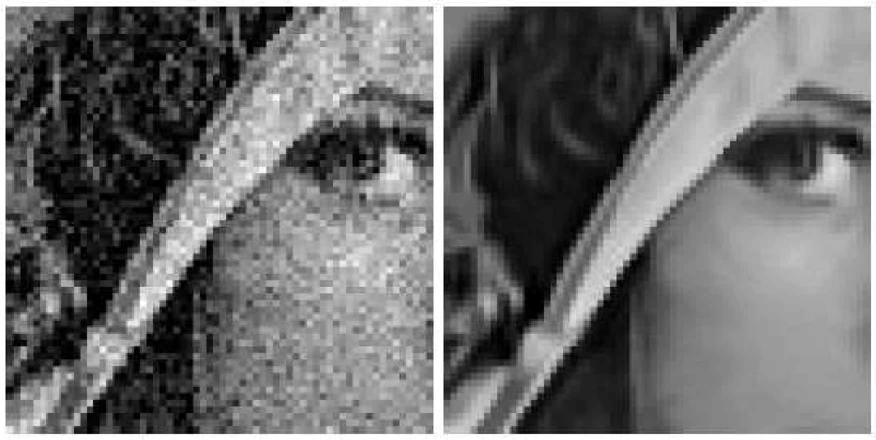

Figure 6.1: NL-means denoising on part of the Lena image. Left: noisy image. Right: image after NL-means denoising. Taken from [10].

6.3. **NL-Means Interpolation.** Based on the simple, elegant denoising filter in (22), we formulate a statistical filter for image interpolation. Suppose we have a low-resolution, possibly noisy, image $u_0 : \Omega_1 \to \Re$ and a version of u_0 upsampled by a factor K, $v : \Omega_K \to \Re$. Here, v will act as our reference image on the finer lattice Ω_K. The interpolation method for obtaining v could be any chosen method, although the nearest neighbor interpolation would be a suitable choice since it does not introduce any image artifacts or additional noise.

Similar to the NL-means denoising case, we wish to compare neighborhoods in the image u_0 that will allow us to interpolate pixel values that reproduce local patterns and textures. To interpolate to the finer lattice, we should compare neighborhoods in v to neighborhoods in the original image u_0. Locally, we may not be able to correctly interpolate the texture of an image. However, the downsampling equation (1) that created the image u_0 may have sampled the texture in a non-uniform fashion so that texture information may be present in one portion of the image that is not present in another.

To motivate this comparison, consider the following scenario. Suppose we are interpolating a low-resolution photograph of a brown-eyed woman. Suppose that the downsampling procedure that transferred the real scene to a camera image did not sample the black pupil in the left eye. When we attempt to zoom in on the left eye, we will have to decide what pixel value to fill in-between the brown pixels of the iris. If we use any of the interpolation schemes described previously, they will use local information to fill in the missing pixel with brown. However, a more natural way to fill in the missing pixel value is to look at the right eye which, if we're lucky, may have sampled the black pupil in the low-resolution photograph. NL-means would compare the neighborhoods throughout the image, decide that the right eye's neighborhood closely resembles the left eye's, and give a large weight to the black pupil contained in the right eye. Note that this example is different than the denoising case described in Section 6.2. The missing pupil in the left eye was not due to noise, it was due to the coarse lattice of the original image.

There is a small issue in comparing neighborhoods on the coarse lattice Ω_1 to neighborhoods on the finer lattice Ω_K. Since the interpolation procedure essentially places empty pixels between pixels, we should think of the neighborhoods as being spread out in a similar manner when we move to a finer grid. Suppose we have a fixed zoom factor $K$and a neighborhood topology on the original image lattice given by $N\left(\vec{x}\right)$. We define a K-neighborhood $N_K\left(\vec{x}\right) \subseteq \Omega_K$ as the set of pixels mapped from $N\left(\vec{x}\right)$ by the upsampling Dirac comb in equation (2). Note that for K=1, $N_1\left(\vec{x}\right) = N\left(\vec{x}\right) \subseteq \Omega_1$. Figure 6.2 illustrates K-neighborhoods where the original neighborhood topology is a 3x3 pixel square. We can think of the K-neighborhood as placing K-1 empty pixels between each pixel in the original neighborhood.

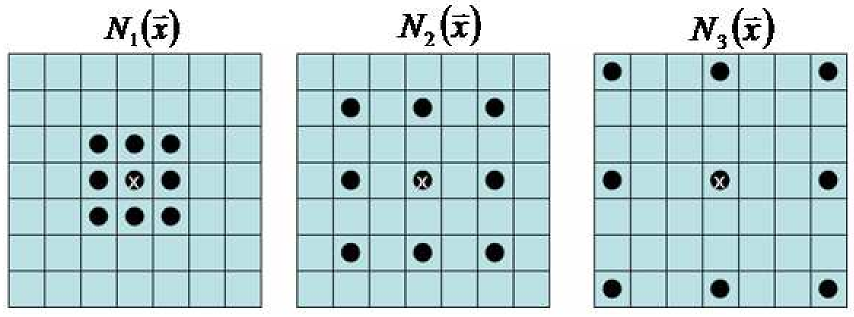

Figure 6.2: Illustration of K-neighborhoods for a 3x3 pixel square topology

Taking the image v as our initialization on the finer lattice Ω_K and using our definition of the K-neighborhood, we can adapt equation (22) to interpolate an image $u : \Omega_K \to \Re$. The NL-means interpolation filter becomes

$$(23)\quad u(\vec{x}) = \frac{1}{Z(\vec{x})} \int_{\Omega_1} u_0(\vec{y}) \exp\left(-\frac{\left\| v(N_K(\vec{x})) - u_0(N_1(\vec{y})) \right\|_2^2}{h^2} \right) d\vec{y}, \quad \vec{x} \in \Omega_K$$

where again $Z(\vec{x})$ is the normalization factor

$$Z(\vec{x}) = \int_{\Omega_1} \exp\left(-\frac{\left\| v(N_K(\vec{x})) - u_0(N_1(\vec{y})) \right\|_2^2}{h^2} \right) d\vec{y}.$$

Again, the fitting parameter h needs to be set experimentally. We used $h = \sqrt{2}\sigma$ as a starting guess, where σ is the standard deviation of the pixel values in u_0. However, we found that this value did not work for all images. If $h$was too small, the image u would show little change from its initialization v. If h was too large, the resulting interpolated image would be blurred. But for an appropriate h, NL-means would simultaneously upsample and denoise the image.

We found that for some natural images with little patterned or textured data, NL-means would perform very poorly regardless of the value of $h$and would fill in many pixels with value zero. Quite simply, if there is no pattern to learn, then NL-means will return a value 0. So we adjusted the filter slightly by adding the initialization point $v(\vec{x})$ to the calculation of $u(\vec{x})$. The pixel $v(\vec{x})$ will necessarily have weight 1 in the filter calculation, so the filter equation (23) becomes

$$(24)\quad u(\vec{x}) = \frac{1}{1 + Z(\vec{x})} \left[1 + \int_{\Omega_1} u_0(\vec{y}) \exp\left(-\frac{\left\| v(N_K(\vec{x})) - u_0(N_1(\vec{y})) \right\|_2^2}{h^2} \right) d\vec{y} \right]$$

with the same normalization constant $Z(\vec{x})$ as before. With this adjustment, even if the image contains no discernible pattern, then NL-means should return the image v. Note that we could use any interpolation scheme to determine v, so we may view NL-means interpolation as a refinement step that can be added to another interpolation scheme.

6.4. **Numerical Results.** For our experiments we used a 5x5 pixel square topology or, when the original image was large enough, a 7x7 pixel square. We used Neumann boundary conditions to determine the neighborhoods of pixels at the border. The nearest neighbor interpolation scheme was used to produce the initial image v. Because each pixel compares its neighborhood to the neighborhood of every other pixel, NL-means interpolation is quadratic in the number of image pixels. The computation time is high and may take several minutes to several minutes to run, depending on the image size.

For most images, we used the parameter value $h = \sqrt{2}\sigma$, although we needed to adjust this value for some images. Figure 6.3 shows the result of applying NL-means with $h = \sqrt{2}\sigma$ to a Brodatz texture. The NL-means image appears less discretized than the bicubic image, but is also more blurred.

Figure 6.4 shows the ability of NL-means to simultaneously remove noise and interpolate an image. The original image contained ringing artifacts from the image conversion process. The edges are sharper than in nearest neighbor interpolation and the ringing artifacts are removed.

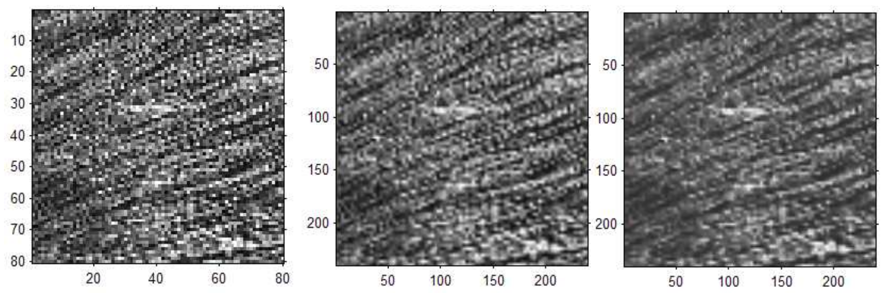

Figure 6.3: Interpolation of Brodatz fabric texture with zoom K=3. Left: Nearest neighbor interpolation. Center: Bicubic interpolation. Right: NL-means interpolation.

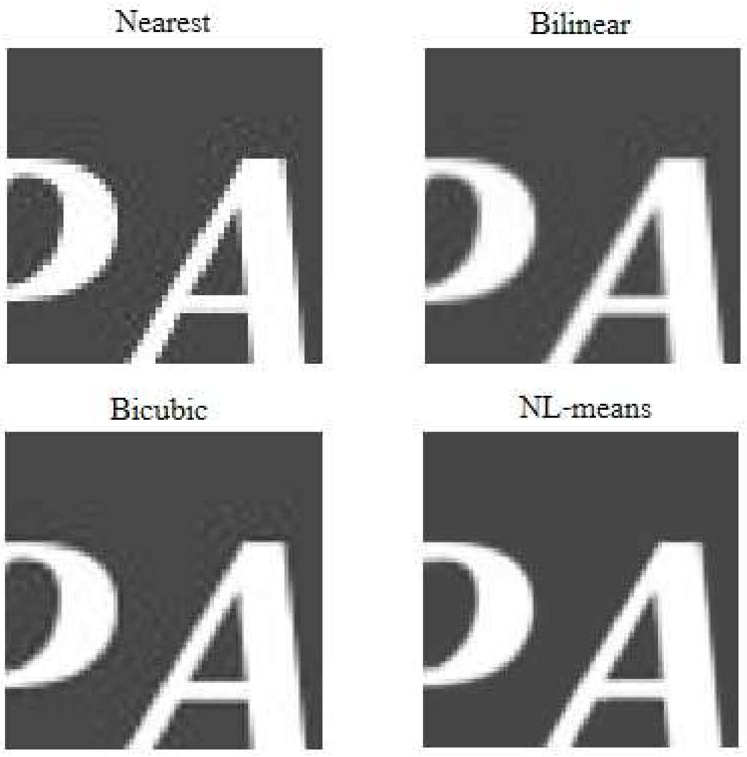

Figure 6.4: NL-means interpolation of ringed image with zoom K=3 compared to linear interpolation filters.

Executing the interpolation and denoising processes simultaneously may have certain advantages over performing them separately. If denoising is performed first, then denoising may also remove fine structures which will be on the level of noise in a low-resolution image. If interpolation is performed first, then noisy data will also be interpolated and the larger noise points will be harder to remove. Figure 6.5 illustrates this concept. In the image at bottom right, the salt and pepper noise points are made larger by the bicubic interpolation and also blurred into the background. The NL-means denoising algorithm is unable to remove the noise, creating a stippled black background. NL-means interpolation is more successful in recovering the pure black background. However, it was harder to remove the noise near the edges because fewer neighborhoods in the image matched these neighborhoods of pixels near the edge.

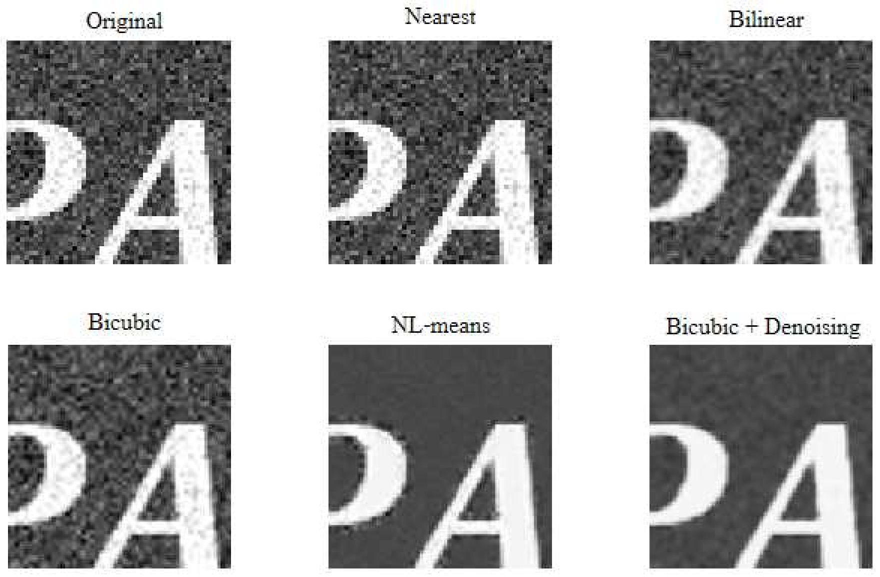

Figure 6.5: Interpolation of a noisy image. The image at bottom-right underwent bicubic interpolation followed by NL-means denoising.

If the parameter h is too large, the image will be blurred and fine structures may be lost. In Figure 6.6, NL-means preserves the edges on the striped texture well and the stripes are smoothed. However, the fine detail of the shirt collar is lost. The original image resolution was a mere 60x60 pixels, which limited the number of neighborhoods that could be compared to. This is the paradox of relying on global information for image zooming: in order to correctly interpolate a high-resolution image, the low-resolution image must be fairly large to begin with.

When a portion of an image is zoomed upon, NL-means could use the entire original image for its comparison neighborhoods. Figure 6.7 shows the result on an MR brain image. The entire MR image was used for the neighborhood calculation. The resulting image has smoothed homogeneous regions, while still giving some hint at texture. The edges, fine structures, and contrast of the image are preserved.

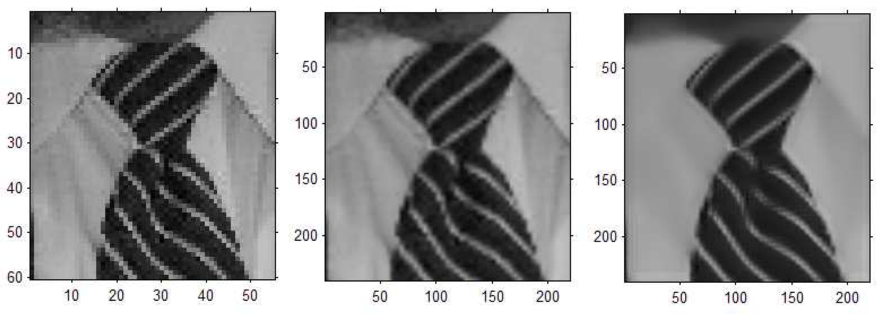

Figure 6.6: NL-means interpolation of textured image with zoom K=4. Left: original image. Center: Bicubic interpolation. Right: NL-means interpolation.

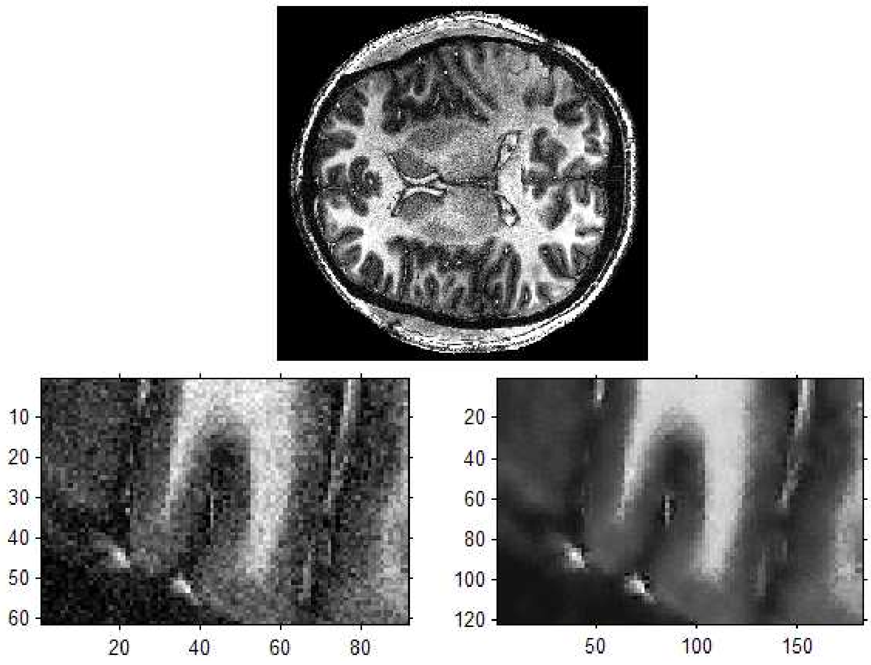

Figure 6.7: NL-means zooming of portion of MR brain image. Top: original MRI. Bottom-left: lower left corner of brain. Bottom-right: NL-means zoom with K=3.

6.5. **Further Research on NL-Means Interpolation.** Our results at this point are preliminary, but NL-means interpolation is very promising and could yield a truly global approach to interpolation. The algorithm is very sensitive to the value of the filter parameter h and this warrants more investigation. We will also experiment with different interpolation schemes for producing the initialization image v.

It may be possible to incorporate the downsampling or camera model into the algorithm. For example, suppose we know the downsampling is preceded by convolution with a Gaussian point spread function (PSF). When comparing neighborhoods, it may be worthwhile to replicate the camera model by convolving $v$with the Gaussian PSF. Another adjustment which may be promising is to use neighborhoods $N(\vec{x}) \setminus \vec{x}$ in our comparisons. In the case when $\vec{x}$ is a noise point, it might prove more meaningful to compare the neighborhood surrounding the point but not the point itself.

As in the last figure, NL-means can use image information that is not part of the portion of the image to be zoomed. It might be feasible to extend this idea to consider neighborhoods in other images, as LLE-based interpolation does. We might also use rotated neighborhoods to better interpolate texture. NL-means also might prove useful for super-resolution: producing a single high-resolution image from a sequence of low-resolution images. Most super-resolution schemes require accurate registration of the image sequence, which can be troublesome if the resolution is very low or the objects in the image undergo more than translations and rotations. NL-means does not require image registration, only a large set of neighborhoods to compare, and may prove useful for super-resolution.

7. Conclusions and Further Research

In this paper, we discussed 4 existing interpolation techniques and presented 1 new technique. Keeping in mind the 9 criteria we introduced in Section 1.1, we briefly summarize the advantages and drawbacks of the methods as follows:

- **Heat diffusion interpolation**: Preserves and smooths edges well, but may over-smooth textured regions and change contrast levels.
- **Mumford-Shah inpainting**: Very sensitive to parameters, which alter both the resulting image and number of iterations required. Also tends to over-smooth textured regions. May result in aliasing, but it may be possible to correct this with extension to Mumford-Shah-Euler model.
- **Wavelet-based interpolation**: Preserves and sharpens edges, but not textures. Reduces to bicubic interpolation in textured or smooth regions. Can only double the resolution of the image.
- **LLE-based interpolation**: The training set needs to be carefully selected to represent the type of images, textures, and order of resolution that will be needed. Tends to create small blocky regions. Unclear if it outperforms bilinear or bicubic interpolation.
- **NL-means interpolation**: Interpolates texture, but not specifically set up to sharpen edges. Best suited for large, textured images. Can simultaneously interpolate and remove noise, but may remove fine structures as well. May result in aliasing or blurring. Sensitive to value of parameter h.

As mentioned in Section 6.1, most interpolation schemes act only on local information and fail to interpolate texture well. This motivated the idea behind the NL-means interpolation in Section 6.3. However, the NL-means interpolation scheme does not always produce satisfactory results, especially on small natural images. We plan to study the NL-means interpolation scheme in more depth and experiment with different versions.

Besides experimentation with the Mumford-Shah-Euler inpainting and NL-mean interpolation, other issues in image interpolation that we plan to study include the research areas listed below. We plan to focus our initial efforts on the first two application-specific problems.

- **Interpolating specific images**: For specific applications involving specific types of images, it may be possible to design a superior interpolation scheme. For example, some research has been conducted on interpolating text images [35]. When the image type is known beforehand, it may be possible to carefully craft a training set for a machine learning algorithm. Another application would be medical imaging, such as MR brain images. The MR community appears to be strictly against machine learning techniques, refusing to let other patient's visual information affect another's. However, the NL-means interpolation may prove useful for MR data.
- **Interpolating specific geometries:** Many interpolation schemes focus on interpolating edges, keeping lines straight, and correctly interpolating curves. It may be interesting to extend this to interpolating desired geometries. For example, a barcode image should consist of only straight lines, right angles, and distinct black and white regions. One possible approach would be to adjust the Mumford-Shah energy. Rather than adding an Euler's elastica term to the energy, we could add a corner-fitting force or some other prior to encourage a specific geometry.
- **Texture interpolation**: Most interpolation algorithms tend to over-smooth or simply ignore textured regions. It appears that texture information needs to be based on more than just local information. NL-means interpolation is promising, but much more work needs to be done to refine the scheme to avoid blurring and aliasing. Wang and Mueller showed that it is possible to interpolate using ideas from texture synthesis research [33]. It may be possible to incorporate mathematical work on quantifying and classifying texture, such as textons [36, 27] or wavelet coefficients [28]. Wei and Levoy have devloped texture synthesis algorithms based on neighborhood comparisons similar in nature to the NL-means filter [37]. Several schemes for textured inpainting based on neighborhood comparisons have been proposed recently and these may also prove enlightening for refining NL-means interpolation [38, 39].
- **Color image interpolation:** In this paper, we dealt exclusively with grayscale images. Most papers on interpolation also do this, as [9] indicates that only the luminance channel Y of the YIQ color space needs to be interpolated with accuracy. The chromaticity channels I and Q are generally copied over with a nearest neighbor upsampling. But working on one channel ignores potentially useful information. In NL-means denoising, [10] indicates that they can actually achieve superior results and use smaller neighborhoods when working with color images. It may be possible to similarly improve the results of NL-means interpolation.
- **Incorporating the camera model**: In Section 1.1, we assumed that the image was produced by a strict nearest neighbor downsampling. In general, the downsampling procedure involves a convolution with a blur kernel or

some other type of pointwise averaging. Aly and Dubois gave preliminary results for incorporating the camera model into TV interpolation [40]. It may be possible to incorporate this into the Mumford-Shah energy in a similar manner or to adjust the neighborhood comparison in NL-means.

- **Performance analysis**: When considering multiple interpolation approaches, it is certainly desirable to have a means of judging performance. The most common approach is to downsample a high-resolution, interpolate the low-resolution image to the same level, and compute the mean square error. But this error rate does not necessarily reflect how visually pleasing the image is, nor does it quantify edge sharpness. Other authors have suggested looking at SNR or PSNR, but again these values may not reflect human perception. It may be possible to measure quantities such as edge sharpness, smoothness, and aliasing. It may also be possible to repeatedly downsample and interpolate images to determine at what level the interpolation method breaks. If the human observer is the ultimate judge, perhaps it is would be worthwhile to conduct surveys to judge and compare methods.

References

[1] Marcelo Bertalmio, A.L. Bertozzi, and Guillermo Sapiro. *Navier-Stokes, fluid dynamics, and image and video inpainting.* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2001, p. 355-362.

[2] Paul Davis. *Mathematics and Imaging.* Mathematical Awareness Week Theme Essay, 1998. Online at http://www.mathaware.org/mam/98/articles/theme.essay.html.

[3] Robert Keys. *Cubic convolution interpolation for digital image processing.* IEEE Transactions on Acoustic, Speech, and Signal Processing, Vol. 29, December 1981, p. 1153-1160.

[4] H.S. Hou and H.C. Andrews. *Cubic splines for image interpolation and digital filtering.* IEEE Transactions on Acoustic, Speech, and Signal Processing, Vol. 26, 1978, p. 508-517.

[5] Francois Malgouyres and Frederic Guichard. *Edge direction preserving image zooming: A mathematical and numerical analysis.* SIAM Journal of Numerical Analysis, Vol. 39, No. 1, 2001, p. 1-37.

[6] Abdelmounim Belahmidi and Frederic Guichard. *A partial differential equation approach to image zoom.* Proceedings of International Conference on Image Processing, January 2004.

[7] Selim Esedoglu and Jianhong Shen. *Digital inpainting based on the Mumford-Shah-Euler image model.* European Journal of Applied Mathematics, Vol. 13, 2002, p. 353-370.

[8] Knox Carey, Daniel Chuang, and Sheila Hemami. *Regularity-preserving image interpolation.* IEEE Transactions on Image Processing, Vol. 8, No. 9, 1999, p. 1293-1297.

[9] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. *Super-resolution through neighbor embedding.* Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2004, p. 275-282.

[10] Antoni Buades, Bartomeu Coll, and J.M. Morel. *On image denoising methods.* CMLA Preprint, CMLA 2004-15, 2004.

[11] Youngjoon Cha and Seongjai Kim. *Edge-forming methods for image zooming.* Journal of Mathematical Imaging and Vision, accepted 2005.

[12] Frederic Guichard and Jean-Michel Morel. *Image Analysis and PDE's.* IPAM GBM Tutorial, March 2001.

[13] Pietro Perona and Jitendra Malik. *A scale space and edge detection using anisotropic diffusion.* Proceedings of IEEE Workshop on Computer Vision, 1987, p. 16-22.

[14] Abdelmounim Belahmidi. *PDEs applied to image restoration and image zooming.* PhD thesis, Universite de Paris XI Dauphine, 2003.

[15] Lawrence Evans. *Partial Differential Equations.* AMS Press, 1999.

[16] William Henshaw. *Time step determination for PDEs with application to programs written with Overture.* Lawrence Livermore National Lab Tech Report UCRL-MA-134290, June 2002.

[17] Leonid Rudin, Stanley Osher, and Emad Fatemi. *Nonlinear total variation based noise removal algorithms.* Physica D, Vol. 60, 1992, p. 259-268.

[18] Francois Malgouyres. *Increase in the resolution of digital images: Variational theory and applications.* PhD thesis, Ecole Normale Superieure de Cachan, 2000.

[19] David Mumford and J. Shah. *Optimal approximations by piecewise smooth functions and associated variational problems.* Communications of Pure and Applied Mathematics, Vol. 42, 1989, p. 577-685.

[20] Andy Tsai, Anthony Yezzi, and Alan Willsky. *Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation and magnification.* IEEE Transactions on Image Processing, Vol. 10, No. 8, 2001, p. 1169-1186.

[21] L. Ambrosio and V.M. Tortorelli. *Approximation of functionals depending on jumps by elliptic functionals via* Γ*-convergence.* Communications of Pure and Applied Mathematics, Vol 43, 1990, p. 999-1036.

[22] E. De Giorgi. *Some remarks on* Γ*-convergence and least squares methods.* Composite Media and Homogenization Thoery, Birkhauser Press, 1991, p. 135-142.

[23] Nhat Nguyen and Peyman Milanfar. *A wavelet-based interpolation-restoration method for super-resolution.* Circuits, Systems, and Signal Processing, Vol. 19, No. 4, 2000, p. 321-338.

[24] Nicholas Ward. *Redundant discrete wavelet transform based super-resolution using sub-pixel image registration.* MS thesis, Air Force Institute of Technology, 2003.

[25] Knox Carey, Daniel Chuang, and Sheila Hemami. *Regularity-preserving image interpolation.* IEEE Transactions on Image Processing, Vol. 8, No. 9, 1999, p. 1293-1297.

[26] Ingrid Daubechies. *Ten Lectures on Wavelets.* SIAM Press, 1992.

[27] B. Julesz. *Textons, the elements of texture perception and their interactions.* Nature, Vol. 290, March 1981.

[28] Eero Simoncelli and J. Portilla. *Texture characterization via second-order statistics of wavelet coefficient amplitudes.* Proceedings of 5th IEEE Conference on Image Processing, October 1998.

[29] William Freeman, Thouis Jones, and Egon Pasztor. *Example-based super-resolution.* MERL Technical Report, TR 2001-30, August 2001.

[30] Bryan Russell. *Exploiting the sparse derivative prior for super-resolution.* M.S. thesis, MIT, 2003.

[31] Lawrence Saul and Sam Roweis. *Think globally, fit locally: Unsupervised learning of low dimensional manifolds.* Journal of Machine Learning Research, Vol. 4 2003, p. 119-155.

[32] Ulf Grenander. *Toward a theory of natural scenes.* Brown Technical Report, 2003.

[33] Lujin Wang and Klaus Mueller. *Generating sub-resolution detail in images and volumes using constrained texture synthesis.* Proceedings of IEEE Conference on Visualization, October 2004, p. 75-82.

[34] Simon Baker and Takeo Kanade. *Limits on super-resolution and how to break them.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, 2002, p. 1167-1183.

[35] David Capel and Andrew Zisserman. *Super-resolution of text image sequences.* Proceedings of International Conference on Pattern Recognition, 2000.

[36] David Mumford and Song-Chun Zhu. *Filters, random fields, and maximum entropy: Towards a unified theory for texture modeling.* International Journal of Computer Vision, Vol. 27, No. 2, March/April 1992, p. 1-20.

[37] Li-Yi Wei and Marc Levoy. *Fast texture synthesis using tree-structured vector quantization.* Proceedings of ACM SIGGRAPH, 2000.

[38] A. Criminsi, P. Perez, and K. Toyama. *Object removal by exemplar-based inpainting.* Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, June 2003, p. 721-728.

[39] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. *Fragment-based image completion.* Proceedings of ACM SIGGRAPH, July 2003.

[40] Hussein Aly and Eric Dubois. *Image up-sampling using total variation regularization with a new observation model.* IEEE Transactions on Image Processing, Vol. 14, No. 4, April 2005, p. 461-474.

8. Acknowledgements

I'd like to thank my advisor, Fadil Santosa, for suggesting this line of research and supporting me throughout my time in graduate school. I am grateful to Jianhong Shen for his comments on Section 3, Willard Miller for checking the accuracy of Section 4, and Gilad Lerman for his help on writing Section 5. I am also very grateful to Prof. Shen for providing the Mumford-Shah inpainting code that was used in Section 3. Finally, I'd like to thank my oral exam committee for listening to my presentation, reading my paper, and passing me: Fadil Santosa, Jianhong Shen, Gilad Lerman, and Vipin Kumar.