Optimized Quasi-Interpolators for Image Reconstruction

and

Consistent Volumetric Discretizations Inside Self-Intersecting Surfaces

by

Leonardo Sacht

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor in Mathematics
IMPA - Instituto Nacional de Matematica Pura e Aplicada
April, 2014

ACKNOWLEDGEMENTS

First of all I would like to thank my family for supporting me during all these years, especially my mother Etelca and my brother Gustavo.

I also want to thank Diego Nehab for being the advisor of my doctoral studies. Not only he helped me with this thesis but also supported and encouraged my career decisions.

I am grateful to Olga Sorkine-Hornung (ETH Zurich), Luiz Velho (IMPA) and Licio Bezerra (UFSC). Olga supervised the work in the second half of this thesis and taught me a lot during the year I spent at her lab. Luiz helped me since I joined IMPA in 2008 and we have worked together in collaborative projects since then. Licio was my first mentor when I was still an undergraduate student and his lessons have accompanied me since then.

I was very fortunate to work with great collaborators during these four years and without them I would not be able to develop the research I did. So I would like to thank Alec Jacobson (ETH Zurich), Daniele Panozzo (ETH Zurich), Rodolfo Lima (IMPA), Marcelo Cicconet (IMPA), Christian Schueller (ETH Zurich), Etienne Vouga (Harvard University), Paulo Cezar Carvalho (IMPA) and Marcelo Gattass (PUC-Rio).

I thank professors, colleagues and people that work at IMPA for making it a good study environment. I also thank the colleagues from the Interactive Geometry Lab (IGL) and ETH Zurich for the great moments I had the year I spent there.

I am grateful to my personal friends that helped me making these four years more pleasant. All friends from Rio de Janeiro, Zurich, Joinville, Belo Horizonte and Florianopolis contributed in some sense to help me reach my goal.

Finally, but not less important, I thank the Brazilian government for supporting three years of my doctoral studies and the Swiss government for supporting one year. In particular, I thank the agencies CNPq (Brazil) and FCS (Switzerland).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

RESUMO

Optimized Quasi-Interpolators for Image Reconstruction
and
Consistent Volumetric Discretizations Inside Self-Intersecting Surfaces

by

Leonardo Sacht

Orientador: Diego Nehab

Primeiro apresentamos uma revisão do problema de amostragem e reconstrução. Esta revisão é focada principalmente nos aspectos históricos e teóricos que serão fundamentais para apresentar nosso novo método. Nós então obtemos novos quasi-interpoladores para reconstrução contínua de imagens amostradas ao minimizar uma nova função objetivo que leva em conta o erro de aproximação no intervalo de Nyquist inteiro. Para alcançar este objetivo, nós otimizamos em relação a todos os graus de liberdade no esquema de aproximação. Consideramos três casos de estudo que oferecem diferentes balanços entre qualidade e custo computacional: um esquema linear, um quadrático e um cúbico. Experimentos com rotações e translações compostas confirmam que nossos novos quasi-interpoladores se comportam melhor do que o estado-da-arte por um custo computacional similar.

Nós então voltamos nossa atenção para um problema diferente no campo de processamento geométrico: o de gerar discretizações consistentes dentro de malhas de triângulos que se auto-intersectam. Com o objetivo de propor um algoritmo para este problema, nós primeiro introduzimos os principais conceitos de malhas tetraedrais, fluxos geométricos, energias de deformação e parametrização. Nós então observamos que a maioria dos passos no pipeline de processamento geométrico de superfícies como deformação, suavização, subdivisão e decimação podem criar auto-interseções. Processamento volumétrico de formas sólidas se torna difícil então, porque obter uma discretização volumétrica correta é impossível: os métodos de tetraedralização existentes requerem entradas do tipo watertight. Nós propomos um algoritmo que produz uma malha tetraedral que se sobrepõe consistentemente com as auto-interseções da superfície de entrada. Isto permite processamento volumétrico em modelos que se auto-intersectam. Nós usamos o fluxo de curvatura média conformalizado, que remove auto-interseçõs, e definimos um fluxo reverso intrinsicamente similar, que previne elas. Tetraedralizamos a superfície resultante e a mapeamos para dentro da superfície original. Nós demonstramos a efetividade de nosso método com aplicações em computação automática de pesos para deformação, simulação baseada em física e computação de distâncias geodésicas.

Keywords: interpolação, aproximação, tetraedralização.

ABSTRACT

Optimized Quasi-Interpolators for Image Reconstruction
and
Consistent Volumetric Discretizations Inside Self-Intersecting Surfaces

by

Leonardo Sacht

Chair: Diego Nehab

We first provide a review on the sampling and reconstruction problem. This review focuses mainly on theoretical and historic aspects that will be the fundamental to present our new framework. We then obtain new quasi-interpolators for continuous reconstruction of sampled images by minimizing a new objective function that takes into account the approximation error over the full Nyquist interval. To achieve this goal, we optimize with respect to all possible degrees of freedom in the approximation scheme. We consider three study cases offering different trade-offs between quality and computational cost: a linear, a quadratic, and a cubic scheme. Experiments with compounded rotations and translations confirm that our new quasi- interpolators perform better than the state-of-the-art for a similar computational cost.

We then turn our attention to a different problem on the field geometry processing: the one of generating consistent tetrahedral discretizations inside self-intersecting triangle meshes. With the goal of proposing an algorithm for this problem, we first introduce the main concepts of tetrahedral meshes, geometric flows, deformation energies and parametrization. We then observe that most steps in the geometry processing pipeline for surfaces, like deformation, smoothing, subdivision and decimation, may create self-intersections. Volumetric processing of solid shapes then becomes difficult, because obtaining a correct volumetric discretization is impossible: existing tet-meshing methods require watertight input. We propose an algorithm that produces a tetrahedral mesh that overlaps itself consistently with the self-intersections in the input surface. This enables volumetric processing on self-intersecting models. We leverage conformalized mean-curvature flow, which removes self-intersections, and define an intrinsically similar reverse flow, which prevents them. We tetrahedralize the resulting surface and map the mesh inside the original surface. We demonstrate the effectiveness of our method with applications to automatic skinning weight computation, physically based simulation and geodesic distance computation.

Keywords: interpolation, approximation, tetrahedralization.

CHAPTER I

Introduction

In this thesis we present the two most recent works of its author. More specifically, we provide the necessary background and all the details contained in the following references:

- *Sacht and Nehab* (2014): Sacht, L., and D. Nehab, *Optimized quasi-interpolators for image reconstruction*, IMPA Technical Report A5739/2014, 2014.

- *Sacht et al.* (2013): Sacht, L., A. Jacobson, D. Panozzo, C. Schuller, and O. Sorkine-Hornung, *Consistent volumetric discretizations inside self-intersecting surfaces*, Computer Graphics Forum (SGP 2013), 32(5), 147-156.

Although these papers share common aspects, the reader can safely read chapters II and III and chapters IV and V independently. Chapter VI points to future directions on both topics.

The main aspect shared by both papers is the use of solid mathematical background to solve fundamental problems from visual computing. We are going to explore different fields from mathematics, such as approximation theory, functional analysis, optimization, numerical analysis, differential geometry and some others. Although not prioritary, we also present implementation details for our new algorithms.

The need for a solid background in different areas made us write two additional chapters (II and IV). The target audience of this thesis comprehends graduate students and researchers from mathematics, computer science and engineerings.

1.1 Optimized quasi-interpolators for image reconstruction

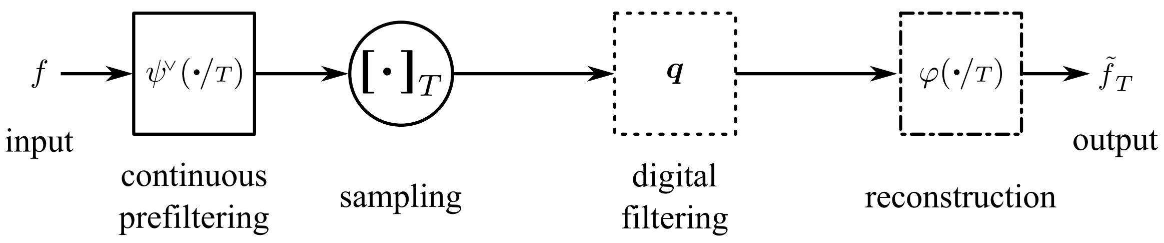


Figure 1.1: The sampling and reconstruction pipeline. The input f is subject to four different stages and produces an output approximation $\tilde{f}_T$. Chapter II reviews the literature and presents the main results regarding how well $\tilde{f}_T$ approximates f.

Figure 1.1 illustrates the material presented in Chapter II. We introduce the main concepts involved in the pipeline of sampling a function and obtaining a continuous reconstruction for it. An input function f is convolved with a prefilter ψ and sampled in a uniform grid. The resulting samples are convolved with a digital filter $\boldsymbol{q}$ and continuously reconstructed by a mixed convolution with a generator φ. This problem is fundamental in image processing, finite elements and many others. We provide the history of this problem including the fundamental work by *Shannon* (1949) and *Strang and Fix* (1971), the involved theory and notations, and comparative results of the different existing schemes.

After understanding the fundamentals of sampling and reconstruction we present a new method that is superior to previous ones on the task of reconstructing natural images. Figure 1.2 shows the superiority of our method compared to the best in the literature designed for the same task. We start this chapter by explaining flaws in

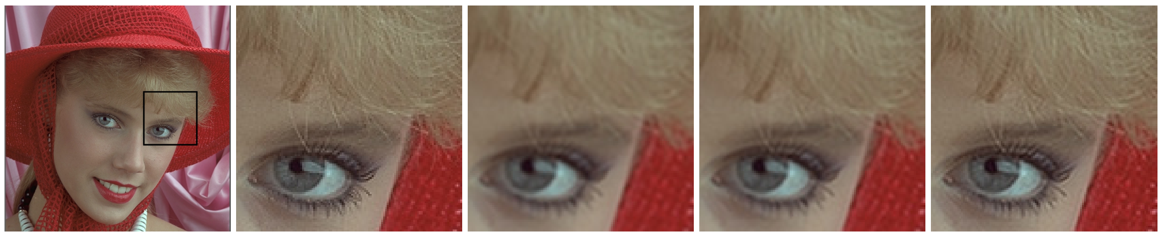

(a) Input image and detail (b) *Blu et al.* (2001) (c) *Condat et al.* (2005) (d) Our result

Figure 1.2: Comparison between state-of-the-art quadratic quasi-interpolators with similar computational cost. The test consists of applying 40 cumulative translations to the input. Our new quadratic is better at preserving detail. PSNR: (a) ∞, (b) 32.938, (c) 34.149, (d) 36.443.

previous methods and presenting criteria to obtain better digital filters and generators for the problem. This leads us to an optimization problem that we discuss and describe a solution. We then present the resulting interpolators and comparisons against the state-of-the-art, which clearly confirm the higher quality obtained by our method.

1.2 Consistent volumetric discretizations inside self-intersecting surfaces

We then present the algorithm illustrated in figure 1.3 for tetrahedralization of self-intersecting triangle meshes. In Chapter IV we describe the fundamentals of geometric flows, tetrahdralizations and volumetric deformation energies. In each part we also describe our choices for the pipeline in figure 1.3 and why they are the most appropriate for our tetrahedralization algorithm. Then in Chapter V we describe our algorithm, its implementation and results.

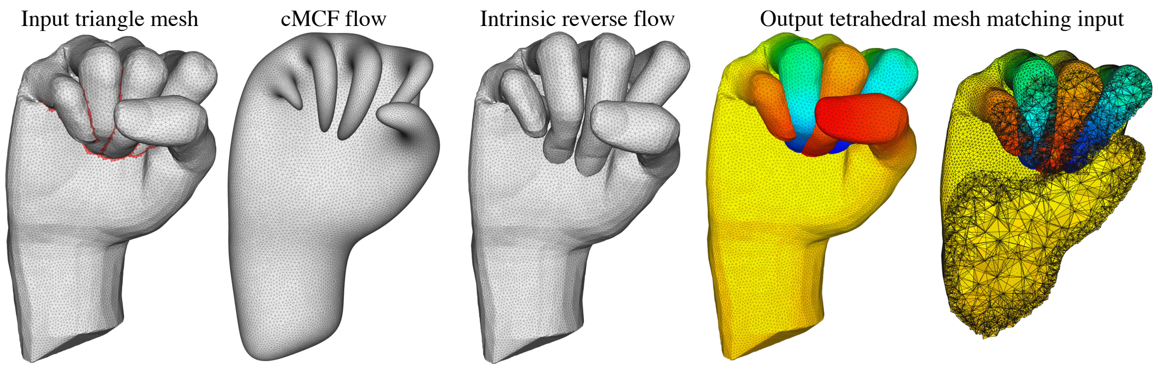


Figure 1.3: The triangle mesh of the *Hand* forms a closed surface, but contains nearly 2000 intersecting triangle pairs (indicated in red). Our method flows the surface according to a geometric flow until all self-intersections are removed. Then we *reverse* the flow so that shape intrinsics are restored but self-intersections are avoided. Finally we can tet-mesh inside this surface and map the mesh so that it matches the original surface.

1.3 Concluding Remarks

Finally, we describe in Chapter VI directions for future work on both image and geometry processing. Some of them were already partially developed and we show partial results for them. One of the algorithms is for real-time high quality image processing based on the theory presented in Chapters II and III. The other algorithm is for generation of cage meshes that enclose dense triangle meshes for the tasks of mulitdrig computations and deformation.

CHAPTER II

Sampling and Reconstruction

2.1 Introduction

This chapter is devoted to present the theory that we are going to use to propose our new approximation scheme in the next chapter. This literature review has also a historic component, since we present results in chronological order so the reader can understand how the theory was developed through time.

We either provide a proof or point to a reference that contains the proof for each theorem. We opted for proving the theorems that we could not find the proof elsewhere, or that had a proof that was not clearly organized in a way we could give a clear reference to the reader.

The results we present in the sequel apply to unidimensional functions, i.e., functions $f : \mathbb{R} \to \mathbb{C}$. The algorithms obtained in 1D are applied to 2D (images) in a tensor product fashion and separately for each color channel.

Figure 2.1 shows the modern approach to sampling and reconstruction that we are going to describe in this chapter. The precise definition of each stage in the pipeline is given in Section 2.2. Intuitively, an approximation $\tilde{f}_T$ to f is obtained as follows. In the first two stages, f is subjected to a continuous convolution with an *analysis filter* ψ (a.k.a. *prefilter*), and then sampled with constant sample spacing T. The traditional role of the prefiltering stage is to eliminate from f frequencies above the

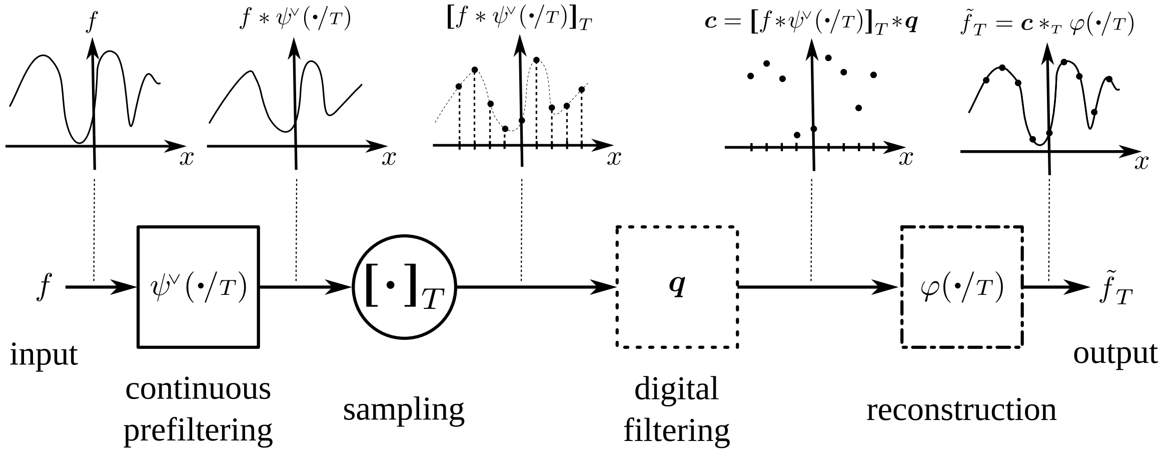


Figure 2.1: All stages of the modern sampling and reconstruction pipeline. Top plots illustrate the output of each state. The theory we are going to present in this chapter quantifies the approximation error between $\tilde{f}_T$ and f.

Nyquist rate $\frac{0.5}{T}$ so as to avoid aliasing in the sampled sequence. The remaining stages apply a *digital filter* $\boldsymbol{q}$ to the samples (by discrete convolution), and then build $\tilde{f}_T$ by combining shifted copies of a *generating function* φ, each scaled by a filtered sample.

2.2 Notation

Let $f : \mathbb{R} \to \mathbb{C}$ be a function. We say that $f \in L_2$ if

$$\int_{-\infty}^{\infty} \left|f(x)\right|^2 \mathrm{d}x < \infty. \tag{2.1}$$

In this case, we denote the quantity in (2.1) by $\|f\|_{L_2}$.

The complex conjugate of $z = a + bi$ is given by $z^* = a - bi$.

The Fourier transform of f and its inverse are given by

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x\omega}\,\mathrm{d}x, \quad \text{and} \tag{2.2}$$

$$f(x) = \int_{-\infty}^{\infty} \hat{f}(\omega) e^{2\pi i x\omega}\,\mathrm{d}\omega. \tag{2.3}$$

A sequence $\boldsymbol{q} : \mathbb{Z} \to \mathbb{C}$ belongs to ℓ_2 if $\sum_{k\in\mathbb{Z}} |q_k|^2 < \infty$. The discrete-time Fourier transform (DTFT) of $\boldsymbol{q}$ is defined by

$$\hat{\boldsymbol{q}}(\omega) = \sum_{k\in\mathbb{Z}} q_k e^{-2\pi i\omega k}. \tag{2.4}$$

Sampling f on a grid with spacing T amounts to obtaining

$$\boldsymbol{[}f\boldsymbol{]}_T := \big[\ldots, f(-T),\, f(0),\, f(T), \ldots\big]. \tag{2.5}$$

The flip of a function f and of a digital filter $\boldsymbol{q} = [\ldots, q_{-1},\, q_0,\, q_1, \ldots]$ are defined as

$$f^{\vee}(x) = f(-x) \quad \text{and} \quad (\boldsymbol{q}^{\vee})_i = \boldsymbol{q}_{-i} \tag{2.6}$$

Given functions f and g, and sequences $\boldsymbol{c}$ and $\boldsymbol{q}$, the continuous, discrete, and mixed convolutions are given respectively by

$$(f * g)(x) = \int_{-\infty}^{\infty} f(t) g(x-t)\,\mathrm{d}t, \tag{2.7}$$

$$(\boldsymbol{c} * \boldsymbol{q})_n = \sum_{k\in\mathbb{Z}} c_k\, q_{n-k}, \quad \text{and} \tag{2.8}$$

$$(\boldsymbol{q} *_T f)(x) = \sum_{k\in\mathbb{Z}} q_k\, f(x - kT). \tag{2.9}$$

With these notations, we can conveniently express the output of the pipeline described in figure 2.1:

$$\tilde{f}_T = [\![f * \psi^\vee(\,\cdot/T)]\!]_T * \boldsymbol{q} *_T \varphi(\,\cdot/T) \tag{2.10}$$

The auto-correlation of a function φ can be written equivalently as

$$a_\varphi(x) = (\varphi * \varphi^\vee)(x) = \int_{-\infty}^{\infty} \varphi(t)\,\varphi(t-x)\,\mathrm{d}t. \tag{2.11}$$

It is frequently sampled into a discrete sequence

$$\boldsymbol{a_\varphi} = [\![a_\varphi]\!]. \tag{2.12}$$

The convolution inverse of $\boldsymbol{q}$, when it exists, is another sequence denoted by $\boldsymbol{q}^{-1}$ such that

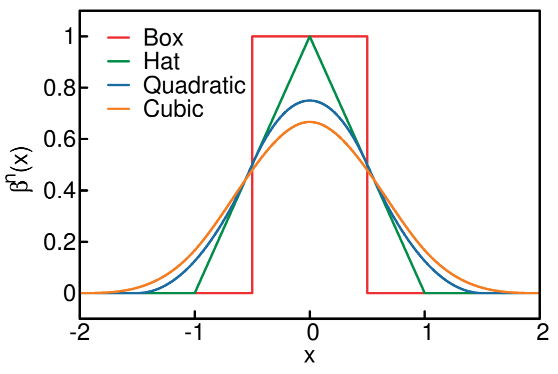
$$\boldsymbol{q} * \boldsymbol{q}^{-1} = \boldsymbol{\delta} = [\ldots, 0, 0, 1, 0, 0, \ldots]. \tag{2.13}$$

The digital filter $\boldsymbol{q}$ is said do be FIR (for Finite Impulse Response) if it has a finite number of non-zero entries. It is said to be IFIR (for Inverse of Finite Impulse Response) if it is the inverse of a FIR digital filter. Finally, $\boldsymbol{q}$ is said to be FIR-IFIR if it can be written as $\boldsymbol{q} = \boldsymbol{q}_1 * \boldsymbol{q}_2$, where $\boldsymbol{q}_1$ is FIR and $\boldsymbol{q}_2$ is IFIR.

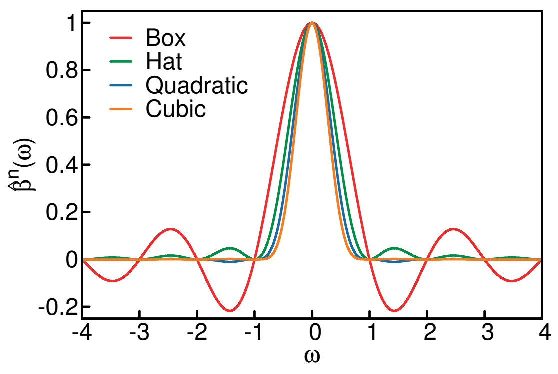
The subspace of functions generated by shifted copies of the generator φ widened to match the grid with spacing T is defined by

$$V_{\varphi,T} = \left\{ \tilde{f} = \boldsymbol{c} *_T \varphi(\,\cdot/T) \;\middle|\; \forall \boldsymbol{c} \in \ell_2 \right\}. \tag{2.14}$$

The B-spline is the most fundamental family of generators and will be useful to



(a) Spatial representation



(b) Fourier representation

Figure 2.2: Plot of the spatial and Fourier representations of β^0 (a.k.a. box function), β^1 (a.k.a. hat function or linear B-spline), β^2 (a.k.a. quadratic B-spline) and β^3 (a.k.a. cubic B-spline).

illustrate many concepts in the following sections. It can be recursively defined as

$$\beta^0(x) = \begin{cases} 1, & |x| < 0.5 \\ 0.5, & |x| = 0.5 \\ 0, & |x| > 0.5 \end{cases}, \quad \text{and} \qquad (2.15)$$

$$\beta^n = \beta^{n-1} * \beta^0, \quad n \geqslant 1.$$

The corresponding Fourier transforms are given by

$$\hat{\beta}^n(\omega) = \operatorname{sinc}(\omega)^{n+1} = \left(\frac{\sin(\pi\omega)}{\pi\omega}\right)^{n+1}, \quad n \geqslant 0. \qquad (2.16)$$

We plot some of the B-spline family members in figure 2.2.

We use several concepts when describing a function $\rho : \mathbb{R} \to \mathbb{R}$. It is *symmetric* if $\rho^\vee = \rho$; it has *support* $W > 0$ if W is the length of the smallest interval I for which $\rho(x) = 0$, $\forall x \notin I$; it is *interpolating* if $[\![\rho]\!] = \boldsymbol{\delta}$; it has regularity R if R is the greatest integer for which $f \in C^R$. Finally, if ρ is a piecewise polynomial function, its degree N is the greatest degree of its polynomial pieces.

We commonly use definitions such as (2.5), (2.9) and (2.14) omitting the parameter

T. In these cases, $T = 1$ should be understood.

2.3 Admissibility

We make some assumptions on the scheme described in Figure 2.1. These assumptions ensure that each step of the pipeline generates a suitable input for the next and that the final output $\tilde{f}_T$ is well defined.

For a given real number $r > 0$ we define the Sobolev space $\mathbf{W}_2^r$ as the set of functions f that satisfy $\int_{-\infty}^{\infty}(1+\omega^2)^r \left|\hat{f}(\omega)\right|^2 \mathrm{d}\omega < \infty$.

Assumption II.1. *Regarding the input function f we assume*

$$f \in \mathbf{W}_2^r, \quad \textit{for some } r > \frac{1}{2}. \tag{2.17}$$

Example II.2. Functions $f \in L_2$ that have limited band, i.e., functions for which there exist $M > 0$ such that $\hat{f}(\omega) = 0$, $\forall \omega \in \mathbb{R} \setminus [-M, M]$. Also functions f such that $\hat{f}$ has exponential decay at infinity.

Assumption II.3. *Regarding the prefilter ψ we assume*

$$\max_{\omega \in \mathbb{R}} \left|\hat{\psi}(\omega)\right| = \left\|\hat{\psi}\right\|_\infty \leqslant K < \infty, \quad \textit{for some } K \in \mathbb{R}. \tag{2.18}$$

Example II.4. $\psi = \beta^n$, since $\left|\hat{\beta}^n(\omega)\right| \leqslant 1$, $\forall \omega \in \mathbb{R}$. Or the Dirac's delta δ, for which $\left|\hat{\delta}(\omega)\right| = 1$, $\forall \omega \in \mathbb{R}$

The next theorem shows the importance of assumptions (2.17) and (2.18).

Theorem II.5. *Let $T > 0$. If $f \in \mathbf{W}_2^r$, for some $r > \frac{1}{2}$, and there exists $K \in \mathbb{R}$ such that $\left\|\hat{\psi}\right\|_\infty \leqslant K < \infty$ then $\left[f * \psi^\vee(\cdot/T)\right]_T \in \ell_2$.*

Proof: See (*Blu and Unser*, 1999b), Appendix C, part A.

$\blacksquare$

Assumption II.6. *The digital filter $\boldsymbol{q}$ is either FIR, IFIR or FIR-IFIR (see section 2.2 for details).*

Example II.7. Since β^n has compact support, $\boldsymbol{q} = \left[\beta^n\right]$ and $\boldsymbol{q} = \left[a_{\beta^n}\right]$ are examples of FIR filters. On the other hand, $\boldsymbol{q} = \left[\beta^n\right]^{-1}$ and $\boldsymbol{q} = \left[a_{\beta^n}\right]^{-1}$ are examples of IFIR filters. Another example which is both FIR and IFIR is the Kronecker delta $\boldsymbol{\delta}$ (2.13).

We show in the sequel that, under this assumption, the output after digital filtering with $\boldsymbol{q}$ remains in ℓ_2.

Theorem II.8. *Let $\boldsymbol{c} \in \ell_2$ and $\boldsymbol{q}$ be either FIR, IFIR or FIR-IFIR digital filters. Then $\boldsymbol{c} * \boldsymbol{q} \in \ell_2$.*

Proof: We start by the case where $\boldsymbol{q}$ is FIR. Using Parseval's theorem (in the first equality) and the product identity $\widehat{\boldsymbol{c} * \boldsymbol{q}}(\omega) = \hat{\boldsymbol{c}}(\omega) \cdot \hat{\boldsymbol{q}}(\omega)$ we get

$$\begin{aligned}\sum_{k\in\mathbb{Z}} \left|\boldsymbol{c} * \boldsymbol{q}\right|_k^2 &= \left\|\hat{\boldsymbol{c}}(\omega) \cdot \hat{\boldsymbol{q}}(\omega)\right\|_{L_2} = \int_{-\frac{1}{2}}^{\frac{1}{2}} \left|\hat{\boldsymbol{c}}(\omega)\hat{\boldsymbol{q}}(\omega)\right|^2 d\omega \\ &= \int_{-\frac{1}{2}}^{\frac{1}{2}} \left|\hat{\boldsymbol{c}}(\omega)\right|^2 \left|\sum q_k e^{-2\pi i\omega k}\right|^2 d\omega.\end{aligned} \tag{2.19}$$

Since $\boldsymbol{q}$ is FIR, there are finitely many q_k's that are nonzero, making $\sum q_k e^{-2\pi i\omega k}$ a finite sum of continuous functions and hence continuous. So it does reach a maximum in $[-1, 1]$. But since it is 1-periodic this maximum holds over all $\mathbb{R}$. Let

$$M = \max_{\omega\in\mathbb{R}} \left|\sum q_k e^{-2\pi i\omega k}\right|^2. \tag{2.20}$$

From (2.19) we then have

$$\sum_{k\in\mathbb{Z}} \left|\boldsymbol{c} * \boldsymbol{q}\right|_k^2 \leqslant M \int_{-\frac{1}{2}}^{\frac{1}{2}} \left|\hat{\boldsymbol{c}}(\omega)\right|^2 d\omega = M \sum_{k\in\mathbb{Z}} |c_k|^2 < \infty, \tag{2.21}$$

where we used again Parseval's theorem for the last equality. This proves that $\boldsymbol{c} * \boldsymbol{q} \in \ell_2$, when $\boldsymbol{q}$ is FIR. When $\boldsymbol{q}$ is IFIR, its Fourier transform is given by

$$\hat{\boldsymbol{q}}(\omega) = \frac{1}{\sum q_k e^{-2\pi i \omega k}}. \tag{2.22}$$

The denominator above cannot be zero, since $\boldsymbol{q}$ is IFIR and thus invertible. Then replacing the constant M in (2.20) by

$$M = \max_{\omega \in \mathbb{R}} \left| \frac{1}{\sum q_k e^{-2\pi i \omega k}} \right|^2 \tag{2.23}$$

(this maximum exists for the same reasons) leads to the same conclusion. When $\boldsymbol{q}$ is FIR-IFIR (say $\boldsymbol{q} = \boldsymbol{q}_1 * \boldsymbol{q}_2$), where $\boldsymbol{q}_1$ is FIR and $\boldsymbol{q}_2$ is IFIR, one simply observes that $\boldsymbol{c} * \boldsymbol{q} = \boldsymbol{c} * (\boldsymbol{q}_1 * \boldsymbol{q}_2) = (\boldsymbol{c} * \boldsymbol{q}_1) * \boldsymbol{q}_2$ and each convolution belongs to ℓ_2.

■

Assumption II.9. *Regarding the generator φ, we assume there exist constants $A, B \in \mathbb{R}$ such that*

$$0 < A \leqslant \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) \leqslant B < \infty, \quad \textit{almost everywhere}, \tag{2.24}$$

where $\boldsymbol{a}_{\boldsymbol{\varphi}} = [a_\varphi]$ is the sampled auto-correlation of φ.

Example II.10. $\varphi = \beta^1$. In this case,

$$a_\varphi = \beta^1 * \beta^{1^\vee} = \beta^1 * \beta^1 = \beta^3 \tag{2.25}$$

and

$$\boldsymbol{a}_{\boldsymbol{\varphi}} = [a_\varphi] = [\beta^3] = [\ldots, 0, 0, \frac{1}{6}, \frac{4}{6}, \frac{1}{6}, 0, 0, \ldots]. \tag{2.26}$$

Thus,

$$\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) = \frac{1}{6}e^{-2\pi i\omega(-1)} + \frac{4}{6}e^{2\pi i\omega\cdot 0} + \frac{1}{6}e^{-2\pi i\omega\cdot 1} = \frac{4}{6} + \frac{1}{3}\sin(2\pi\omega). \tag{2.27}$$

Since $-1 \leqslant \sin(2\pi\omega) \leqslant 1$, we have

$$A := \frac{1}{3} = \frac{4}{6} - \frac{1}{3} \leqslant \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) \leqslant \frac{4}{6} + \frac{1}{3} = 1 := B. \tag{2.28}$$

As we show in the sequel, assumption (2.24) implies that $V_{\varphi,T}$ (2.14) is a closed subspace of L_2. This ensures that each element in $V_{\varphi,T}$ has a unique stable representation and that the orthogonal projection of any $f \in L_2$ on $V_{\varphi,T}$ exists (we elaborate further on the orthogonal projection in section 2.5).

Theorem II.11. *Let $T > 0$ and φ be a generator such that*

$$0 < A \leqslant \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) \leqslant B < \infty, \quad \textit{almost everywhere}, \tag{2.29}$$

for some $A, B \in \mathbb{R}$. Then $V_{\varphi,T}$ is a closed subspace of L_2.

Proof: For simplicity we assume $T = 1$. We use a result from (*Aldroubi and Unser*, 1994b) (theorem 2 in their paper) that says that (2.29) is equivalent to

$$A\|\boldsymbol{c}\|_{\ell_2}^2 \leqslant \|\boldsymbol{c} * \varphi\|_{L_2}^2 \leqslant B\|\boldsymbol{c}\|_{\ell_2}^2, \quad \forall \boldsymbol{c} \in \ell_2. \tag{2.30}$$

The interested reader should have knowledge on measure theory to understand their proof. Below we provide a simpler proof of the implication (2.29) $\Rightarrow$ (2.30):
A simpler expression for $\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega)$ can be obtained using Poisson's summation formula:

$$\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) = \sum_{n\in\mathbb{Z}} (\varphi * \varphi^{\vee})(n) e^{-2\pi i\omega n} = \sum_{k\in\mathbb{Z}} \mathcal{F}\big((\varphi * \varphi^{\vee})(\cdot) e^{-2\pi i\omega\cdot}\big)(k) = \sum_{k\in\mathbb{Z}} \left|\hat{\varphi}(\omega + k)\right|^2 \tag{2.31}$$

We can then obtain the following expression for $\left\|\boldsymbol{c} * \varphi\right\|_{L_2}^2$:

$$\begin{aligned}
\left\|\boldsymbol{c} * \varphi\right\|_{L_2}^2 &= \int_{-\infty}^{\infty} \left|\widehat{\boldsymbol{c} * \varphi}(\omega)\right|^2 \mathrm{d}\omega = \int_{-\infty}^{\infty} \left|\hat{\boldsymbol{c}}(\omega)\right|^2 \left|\hat{\varphi}(\omega)\right|^2 \mathrm{d}\omega \\
&= \sum_{k\in\mathbb{Z}} \int_0^1 \left|\hat{\boldsymbol{c}}(\omega+k)\right|^2 \left|\hat{\varphi}(\omega+k)\right|^2 d\omega = \sum_{k\in\mathbb{Z}} \int_0^1 \left|\hat{\boldsymbol{c}}(\omega)\right|^2 \left|\hat{\varphi}(\omega+k)\right|^2 d\omega \qquad (2.32) \\
&= \int_0^1 \left|\hat{\boldsymbol{c}}(\omega)\right|^2 \sum_{k\in\mathbb{Z}} \left|\hat{\varphi}(\omega+k)\right|^2 d\omega = \int_0^1 \left|\hat{\boldsymbol{c}}(\omega)\right|^2 \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) d\omega
\end{aligned}$$

Using the hypothesis $A \leqslant \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) \leqslant B$ almost everywhere we conclude the proof of (2.29) $\Rightarrow$ (2.30).

To prove that V_φ is well-defined (as a subset of L_2) we observe that all members $\boldsymbol{c} * \varphi \in V_\varphi$ satisfy

$$\left\|\boldsymbol{c} * \varphi\right\|_{L_2}^2 \leqslant B\left\|\boldsymbol{c}\right\|_{\ell_2}^2 < \infty \qquad (2.33)$$

and thus belong to L_2. To see that V_φ is a linear subspace of L_2 we observe that the null function $\mathit{0} \in L_2$ can be expressed as $\mathit{0} = \mathbf{0} * \varphi$ and thus belongs to V_φ. Given $\boldsymbol{c}_1 * \varphi, \boldsymbol{c}_2 * \varphi \in V_\varphi$ we have $\boldsymbol{c}_1 * \varphi + \boldsymbol{c}_2 * \varphi = (\boldsymbol{c}_1 + \boldsymbol{c}_2) * \varphi \in V_\varphi$ and $\lambda \cdot (\boldsymbol{c}_1 * \varphi) = (\lambda \cdot \boldsymbol{c}) * \varphi \in V_\varphi$.

To prove that V_φ is closed, let $\{\boldsymbol{c}_n * \varphi\}_n \subseteq V_\varphi$ be a sequence that converges in L_2. We must prove that $\lim_{n\to\infty} \boldsymbol{c}_n * \varphi \in V_\varphi$. Since the inequality $\left\|\boldsymbol{c}_n\right\|_{\ell_2}^2 \leqslant A^{-1}\left\|\boldsymbol{c}_n * \varphi\right\|_{L_2}^2$ holds and $\{\boldsymbol{c}_n * \varphi\}_n$ is a Cauchy sequence (because it is convergent), then $\{\boldsymbol{c}_n\}_n$ is a Cauchy sequence in ℓ_2. Since ℓ_2 is a complete metric space, then $\{c_n\}_n$ is convergent, say $\lim_{n\to\infty} \boldsymbol{c}_n = \boldsymbol{c}_\infty$. Now, the inequality $\left\|\boldsymbol{c} * \varphi\right\|_{L_2}^2 \leqslant B\left\|\boldsymbol{c}\right\|_{\ell_2}^2$ implies $\lim_{n\to\infty} \boldsymbol{c}_n * \varphi = \boldsymbol{c}_\infty * \varphi \in V_\varphi$.

■

2.4 Shannon's theorem

One of the most fundamental results regarding sampling and reconstruction is due to *Shannon* (1949). This result states that if the input f is band limited then it is possible to obtain an output $\tilde{f}_T$ identical to f, as long as T is small enough and the right choices for ψ, $\boldsymbol{q}$ and φ are made.

Theorem II.12. *Let f be a function for which there exists $W > 0$ such that $\hat{f}(\omega) = 0$, $\forall \omega \in \mathbb{R} \setminus \big[-W, W\big]$. Then the approximation scheme presented in figure 2.1 produces $\tilde{f}_T = f$, if $T \leqslant \frac{1}{2W}$, $\psi = \delta$, $\boldsymbol{q} = \boldsymbol{\delta}$ and $\varphi(x) = \operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$.*

Proof: (*Shannon*, 1949), theorem 1.

■

One can check that the admissibility conditions described in section 2.3 are verified for the choices $\psi = \delta$, $\boldsymbol{q} = \boldsymbol{\delta}$ and $\varphi = \operatorname{sinc}$. But this approximation scheme is not used in practice. The first main reason is that signals usually do not have limited band. For example, any color discontinuity in an image generate frequency content tending to infinity. The other drawback is that $\varphi = \operatorname{sinc}$ does not have compact support (and has slow decay), making a mixed convolution $\boldsymbol{c} *_T \varphi$ (see (2.9)) impossible to be implemented in practice.

2.5 Orthogonal projection

The computational cost of the mixed convolution $\boldsymbol{c} *_T \varphi$ (the last step in figure 2.1) is dictated by the support of φ. This is why people opt for generators with small support. The result $f = \tilde{f}_T$ for $\varphi = \operatorname{sinc}$ is no longer achievable (it would not be even for $\varphi = \operatorname{sinc}$ when f is not band-limited, which is usually the case). Since perfect reconstruction is not available, what would be the closest member of V_φ to a given function f?

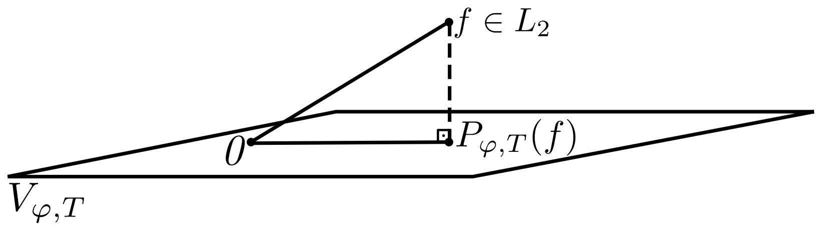


Figure 2.3: When the admissibility conditions are fulfilled, the input $f \in L_2$ can be orthogonally projected into $V_{\varphi,T}$ to obtain $P_{\varphi,T}(f)$. The residual $P_{\varphi,T}(f)-f$ is orthogonal to $V_{\varphi,T}$.

In terms of the L_2 metric, this is the orthogonal projection, i.e., the function $\boldsymbol{c} *_T \varphi \in V_{\varphi,T}$ that is the solution for

$$\underset{\boldsymbol{c}\in\ell_2}{\arg\min} \left\|f - \boldsymbol{c} *_T \varphi\right\|_{L_2} =: P_{\varphi,T}(f). \tag{2.34}$$

When assumption II.9 is satisfied, $V_{\varphi,T}$ is a closed subspace of L_2 (theorem II.11) and the orthogonal projection of a function $f \in L_2$ into $V_{\varphi,T}$ exists. In figure 2.3, we illustrate one of the characteristics of the orthogonal projection $P_{\varphi,T}(f)$: the residual $P_{\varphi,T}(f) - f$ is orthogonal to $V_{\varphi,T}$.

In the sequel we derive an expression for the orthogonal projection and show that it fits the pipeline described in figure 2.1. The derivation of such expression can be done in many ways, but we reproduce the proof by *Nehab and Hoppe* (2014) since it fits and illustrates better the notation we are using in this work.

Theorem II.13. *Let φ be a generating function that satisfies (2.24), $T > 0$ and $f \in L_2$. Then the orthogonal projection of f in $V_{\varphi,T}$ is achieved setting in the approximation scheme*

$$\psi = \varphi, \quad \boldsymbol{q} = [\![a_\varphi]\!]^{-1}, \tag{2.35}$$

i.e.,

$$P_{\varphi,T}(f) = \left[f * \varphi^{\vee}(\,\cdot/T) \right]_T * \left[a_\varphi \right]^{-1} *_T \varphi(\,\cdot/T). \tag{2.36}$$

Proof: For simplicity we take $T = 1$. The property $f - P_\varphi(f) \perp V_\varphi$ is equivalent to

$$\big\langle f - P_\varphi(f), \varphi(\cdot - k) \big\rangle = 0, \quad \forall k \in \mathbb{Z}, \tag{2.37}$$

where the inner product is given by $\langle g, h\rangle = \int_{-\infty}^{\infty} g(x)h(x)^* \, \mathrm{d}x$. (2.37) can be rewritten as

$$\left[f - P_\varphi(f) * \varphi^{\vee} \right] = \mathbf{0}. \tag{2.38}$$

If $P_\varphi(f) = \boldsymbol{c} * \varphi$, the last expression is equivalent to

$$\left[f * \varphi^{\vee} \right] = \left[\boldsymbol{c} * \varphi * \varphi^{\vee} \right] \Leftrightarrow \left[f * \varphi^{\vee} \right] = \boldsymbol{c} * \left[\varphi * \varphi^{\vee} \right] \Leftrightarrow \boldsymbol{c} = f * \varphi^{\vee} * \left[\varphi * \varphi^{\vee} \right]^{-1} \tag{2.39}$$

and we conclude that the orthogonal is obtained by setting

$$\psi = \varphi, \quad \boldsymbol{q} = \left[\varphi * \varphi^{\vee} \right]^{-1} = \left[a_\varphi \right]^{-1}. \tag{2.40}$$

■

2.6 Strang-Fix conditions

Now that we have obtained the best approximation (with respect to the L_2 norm) of a given function $f \in L_2$ in $V_{\varphi,T}$, a natural question arises: how close to f is this approximation? One way of characterizing this concept is by defining the approximation order.

The generator φ has *approximation order* L if L is the greatest positive integer for which there exists a constant $C > 0$ such that

$$\left\|f - P_{\varphi,T}(f)\right\|_{L_2} \leqslant C \cdot T^L \cdot \left\|f^{(L)}\right\|_{L_2}, \forall f \in \mathbf{W}_2^L. \tag{2.41}$$

Intuitively, this means that as T goes to zero, the "distance" between f and its closest function in $V_{\varphi,T}$ decays to zero as fast as T^L. We define approximation order of the orthogonal projection operator $P_{\varphi,T}$ as the same thing as the approximation order of φ.

Strang and Fix presented the main characterizations for the approximation order concept in their seminal papers (*Strang*, 1971) and (*Strang and Fix*, 1973). They showed that approximation order L is equivalent to $V_{\varphi,T}$ being able to reproduce polynomials of degree $L-1$. Simple criteria on $\hat{\varphi}$, the Fourier transform of the generator φ, were also presented. We summarize their main equivalences below.

Theorem II.14. *Let φ be a generating function that satisfies (2.24) and has compact support*[1]*. The following statements are equivalent:*

(i) $p \in V_{\varphi,T}$*, for all polynomials p of degree $L-1$,* $\forall T > 0$*.*

(ii) $\exists \varphi_{QI} \in V_\varphi$ *such that*

$$\forall p \textit{ polynomial of degree } L-1, \quad \sum_{k\in\mathbb{Z}} p(k)\varphi_{QI}(x-k) = p(x) \tag{2.42}$$

(in this case, φ_{QI} is called a quasi-interpolator).

(iii) $\hat{\varphi}(0) \neq 0$ *and* $\hat{\varphi}^{(m)}(k) = 0$, $\forall k \in \mathbb{Z} \setminus \{0\}$ *and* $\forall m \in \{0, \ldots, L-1\}$.

(iv) φ *has approximation order L.*

[1]Strang and Fix initially assumed φ compactly supported, but their result was extended for noncompact φ with sufficient polynomial decay at infinity (*Jia and Lei*, 1993).

Proof: (ii) $\Leftrightarrow$ (iv): See "Approximation Theorem" in (*Strang*, 1971).

(iii) $\Leftrightarrow$ (iv): Theorem 1 in (*Strang and Fix*, 1973).

(ii) $\Rightarrow$ (i): Let $p(x) = a_0 + a_1 x + \ldots + a_{L-1} x^{L-1}$ be a polynomial of degree $L-1$ and $T > 0$. We must prove that $p \in V_{\varphi,T}$, assuming (ii). Observing that $p(Tx) = a_0 + a_1 Tx + \ldots + a_{L-1} T^{L-1} x^{L-1}$ is also a polynomial of degree $L-1$ we know that $p(Tx) = \sum_{k \in \mathbb{Z}} p(Tk) \varphi_{QI}(x-k)$. Substituting $y = Tx$, we obtain the identity $p(y) = \sum_{k \in \mathbb{Z}} p(Tk) \varphi_{QI}\left(\frac{y}{T} - k\right)$, $\forall y \in \mathbb{R}$. Denoting $\varphi_{QI} = \boldsymbol{c} * \varphi$ (since $\varphi_{QI} \in V_\varphi$) and $p_T[k] = p(Tk)$, $\forall k \in \mathbb{Z}$ we have

$$p = (p_T * \boldsymbol{c}) *_T \varphi(\,\bullet/T) \tag{2.43}$$

and thus $p \in V_{\varphi,T}$.

(i) $\Rightarrow$ (ii): Let p be a polynomial of degree $L-1$. Hypothesis (i) implies $p \in V_{\varphi,1} = V_\varphi$ and thus there exists $\boldsymbol{c} \in \ell_2$ such that

$$p = \boldsymbol{c} * \varphi. \tag{2.44}$$

We then have

$$[p] * \varphi = [\boldsymbol{c} * \varphi] * \varphi = \boldsymbol{c} * [\varphi] * \varphi = [\varphi] * \boldsymbol{c} * \varphi = [\varphi] * p. \tag{2.45}$$

Assuming $[\varphi]$ to be invertible (to be shown in the sequel) we have

$$[p] * [\varphi]^{-1} * \varphi = p \tag{2.46}$$

and taking $\varphi_{QI} = [\varphi]^{-1} * \varphi$ we conclude the proof.

It is left to prove that $[\varphi]$ is invertible. Assume the contrary, i.e., there exists $\boldsymbol{c} \neq \mathbf{0}$

such that

$$[\![\varphi]\!] * \boldsymbol{c} = [\![0]\!]. \tag{2.47}$$

Convolving both sides with $[\![\varphi^\vee]\!]$ we obtain

$$[\![\varphi^\vee]\!] * [\![\varphi]\!] * \boldsymbol{c} = [\![\varphi^\vee]\!] * \mathbf{0} = \mathbf{0}. \tag{2.48}$$

But $[\![\varphi^\vee]\!] * [\![\varphi]\!] = [\![\varphi * \varphi^\vee]\!] = [\![a_\varphi]\!]$ and this implies $[\![a_\varphi]\!]$ not invertible, what contradicts (2.24).

■

2.7 Approximation schemes

The orthogonal projection may not be achievable in practice since it assumes it is possible to choose the prefilter as $\psi = \varphi$. For many image processing tasks the prefilter ψ is part of the acquisition device and images come already discretized as $[\![f * \psi^\vee(\,\cdot/T)]\!]_T$. In these cases, it is possible to apply approximation schemes different from the orthogonal projection. Although sub-optimal with respect to the L_2-norm, these schemes may have good approximation properties, as we show in the next sections.

Oblique projection: Fixed both the prefilter ψ and the generator φ, we seek an approximation $P_{\varphi\perp\psi}(f) = \boldsymbol{c} * \varphi$ of the input f into space V_φ with the residual orthogonal to V_ψ (rather than being orthogonal to V_φ, as for the orthogonal projection). In terms of the notation we are using, this means:

$$\begin{aligned}\mathbf{[}(f - P_{\varphi\perp\psi}(f)) * \psi^{\vee}\mathbf{]} = \mathbf{0} \Leftrightarrow \mathbf{[}f * \psi^{\vee}\mathbf{]} &= \mathbf{[}P_{\varphi\perp\psi}(f) * \psi^{\vee}\mathbf{]} \\ \Leftrightarrow \mathbf{[}f * \psi^{\vee}\mathbf{]} &= \mathbf{[}\boldsymbol{c} * \varphi * \psi^{\vee}\mathbf{]}\end{aligned} \tag{2.49}$$

To find the coefficient vector $\boldsymbol{c}$, we take it out of the convolution operator (this is a property of convolutions) and obtain

$$\mathbf{[}f * \psi^{\vee}\mathbf{]} = \boldsymbol{c} * \mathbf{[}\varphi * \psi^{\vee}\mathbf{]} \Leftrightarrow \boldsymbol{c} = \mathbf{[}f * \psi^{\vee}\mathbf{]} * \mathbf{[}\varphi * \psi^{\vee}\mathbf{]}^{-1} \Leftrightarrow \boldsymbol{c} = \mathbf{[}f * \psi^{\vee}\mathbf{]} * \mathbf{[}a_{\varphi,\psi}\mathbf{]}^{-1}, \tag{2.50}$$

where we have assumed the sampled correlation $\mathbf{[}a_{\varphi,\psi}\mathbf{]}$ is invertible. In other words, to obtain the oblique projection we have to prefilter the input f with ψ (which was fixed in the first place), digitally filter with $\mathbf{[}a_{\varphi,\psi}\mathbf{]}^{-1}$ and reconstruct with φ. This strategy is also known as *consistent sampling* (*Unser and Aldroubi*, 1994).

Cardinal Interpolation: Another common situation is when the input is already sampled without any pre filtering, i.e., $\psi = \delta$ and $\mathbf{[}f*\psi^{\vee}\mathbf{]} = \mathbf{[}f\mathbf{]}$. With this information only, we can seek an approximation $\tilde{f} = \boldsymbol{c} * \varphi$ that matches the input function f at integers, i.e., $\mathbf{[}f\mathbf{]} = \mathbf{[}\tilde{f}\mathbf{]}$. This method is known as *cardinal interpolation* and is obtained as follows:

$$\mathbf{[}f\mathbf{]} = \mathbf{[}\tilde{f}\mathbf{]} \Leftrightarrow \mathbf{[}f\mathbf{]} = \mathbf{[}\boldsymbol{c} * \varphi\mathbf{]} \Leftrightarrow \mathbf{[}f\mathbf{]} = \boldsymbol{c} * \mathbf{[}\varphi\mathbf{]} \Leftrightarrow \boldsymbol{c} = \mathbf{[}f\mathbf{]} * \mathbf{[}\varphi\mathbf{]}^{-1}, \tag{2.51}$$

where we have assumed $\mathbf{[}\varphi\mathbf{]}$ to be invertible. Thus, we have to sample f at integers, convolve with the digital filter $\boldsymbol{q} = \mathbf{[}\varphi\mathbf{]}^{-1}$ and reconstruct with φ.

Quasi-Interpolation: Given a generator φ and $\psi = \delta$, $\varphi_{QI} = \boldsymbol{q} * \varphi$ is said to be a *quasi-interpolator of order* L, if it reproduces exactly all polynomials of degree L, i.e.,

$$\big(\mathbf{[}p\mathbf{]} * \boldsymbol{q} * \varphi\big)(x) = p(x), \quad \forall x \in \mathbb{R}, \quad \forall p \text{ polynomial of degree } L-1. \tag{2.52}$$

When φ satisfies the Strang-Fix conditions for approximation order L (theorem II.14), such a quasi-interpolator exists (condition (ii) in the theorem). Moreover, it is possible to prove that, under the same assumptions, the cardinal interpolator $\mathbf{[}\varphi\mathbf{]}^{-1} * \varphi$ is a quasi-interpolator of order L. There may be more than one quasi-interpolator of order L in V_φ and we extensively explore this concept in the next chapter, where we propose novel quasi-interpolators for reconstruction of images.

2.8 Quantification of the L_2 error

In the previous sections we have presented different methods to produce an approximation $\tilde{f}_t$ for an input function f based on the sampling and reconstruction pipeline illustrated in figure 2.1. But we do not have a way of comparing the different schemes. The only quantification of the error presented until now is the approximation order and only for the orthogonal projection.

One of the drawbacks of the approximation order concept is that it assumes the sampling spacing T to be tuneable to be arbitrarily small. For some applications this is not reasonable. For example, in image processing where the pixel grid is fixed.

In the sequel we present a result due to *Blu and Unser* (1999b) that quantifies the error depending on T and that can be applied to different choices of ψ, $\boldsymbol{q}$ and φ, such as the ones that we have presented in the previous sections. This result will also lead us to different characterizations of the approximation error in the next sections.

Theorem II.15. *Let φ, ψ and $\boldsymbol{q}$ satisfy the admissibility conditions in section 2.3.*

For all $f \in \mathbf{W}_2^r$ with $r > \frac{1}{2}$, the approximation error is given by

$$\left\| f - \tilde{f}_T \right\|_{L_2} = \left(\int_{-\infty}^{\infty} \left| \hat{f}(\omega) \right|^2 E(T\omega)\, \mathrm{d}\omega \right)^{\frac{1}{2}} + e(f, T), \tag{2.53}$$

where $e(f,T) = o(T^r)$ and

$$E(\omega) = \left| 1 - \left(\hat{\boldsymbol{q}}(\omega)^* \hat{\psi}(\omega)^* \right) \hat{\varphi}(\omega) \right|^2 + \left| \hat{\boldsymbol{q}}(\omega) \hat{\psi}(\omega) \right|^2 \sum_{k \neq 0} \left| \hat{\varphi}(\omega + k) \right|^2 \tag{2.54}$$

Proof: See appendix C in (*Blu and Unser*, 1999b).

■

2.9 Approximation order

Similarly to the approximation order for the orthogonal projection, we define the concept of approximation order for the general scheme presented in figure 2.1. The scheme is said to have *approximation order* if there exists $C > 0$ such that

$$\left\| f - \tilde{f}_T \right\|_{L_2} \leqslant C \cdot T^L \cdot \left\| f^{(L)} \right\|_{L_2}, \forall f \in \mathbf{W}_2^L. \tag{2.55}$$

The precise quantification of the error given by (2.53) made possible to state the most general condition for a scheme to have approximation order L. We present this condition in the sequel.

For the rest of this section, we denote the combination of continuous prefilter and digital filter as

$$\tilde{\varphi} = \psi * \boldsymbol{q} \quad \left(\Leftrightarrow \hat{\tilde{\varphi}}(\omega) = \hat{\psi}(\omega) \cdot \hat{\boldsymbol{q}}(\omega) \right) \tag{2.56}$$

With this notation, the approximation scheme in figure 2.1 is said to be *quasi-*

biorthonormal of order L if

$$\hat{\tilde{\varphi}}(\omega)^{*}\hat{\varphi}(\omega+k)=\delta_{k}+O\big(\omega^{L}\big),\quad \forall k\in\mathbb{Z}. \tag{2.57}$$

This condition is weaker than traditional biorthonormality presented and analyzed by *Unser* (1996). The scheme is said to be *biorthonormal of order* L if φ has approximation order L and

$$\big\langle\varphi(x-k),\tilde{\varphi}(x-l)\big\rangle=\delta_{k,l},\quad \forall k,l\in\mathbb{Z}. \tag{2.58}$$

In compact notation, the latter condition is expressed simply as

$$[a_{\varphi,\tilde{\varphi}}]=\boldsymbol{\delta}. \tag{2.59}$$

Blu and Unser (1999b) show why biorthonormality of order L implies quasi-biorthonormality of order L. Condition (2.59) gives us a simple way to check that the cardinal interpolation scheme is biorthonormal of order L, given that the generator φ has order L and is symmetric:

$$\begin{aligned}[a_{\varphi,\tilde{\varphi}}]&=[\varphi*\tilde{\varphi}^{\vee}]=[\varphi*\boldsymbol{q}^{\vee}*\psi^{\vee}]=[\varphi*\big([\varphi]^{-1}\big)^{\vee}*\delta^{\vee}]\\&=[\varphi*[\varphi]^{-1}]=[\varphi]*[\varphi]^{-1}=\boldsymbol{\delta}.\end{aligned} \tag{2.60}$$

It is possible to prove that when φ has order L, then orthogonal and oblique projections are biorthonormal of order L and quasi-interpolation of order L is quasi-biorthonormal of order L.

We now present the main theorem of this section that relates approximation order of the general approximation scheme to quasi-biorthonormality of order L.

Theorem II.16. *Let φ, ψ and $\boldsymbol{q}$ satisfy the admissibility conditions in section 2.3. Additionally, we assume boudedness of φ and of $|x|^{2L}\big|\varphi(x)\big|$, and also require that*

$\int_{-\infty}^{\infty} \left|\tilde{\varphi}(x)\right| \mathrm{d}x < \infty$ and $\int_{-\infty}^{\infty} |x|^{2L}\left|\tilde{\varphi}(x)\right| \mathrm{d}x < \infty$. With these hypotheses we have the equivalence

$$(2.55) \Leftrightarrow \varphi \text{ and } \tilde{\varphi} \text{ are quasi-biorthonormal of order } L. \tag{2.61}$$

Proof: Appendix D in (*Blu and Unser*, 1999b).

■

We present another characterization for quasi-biorthonormality of order L that will be useful for our developments in the next chapter.

Theorem II.17. *Let φ, ψ and $\boldsymbol{q}$ satisfy the admissibility conditions in section 2.3. Then the following conditions are equivalent:*

(i) φ and $\tilde{\varphi}$ are quasi-biorthonormal of order L.

(ii) The error kernel (2.54) of the approximation scheme in figure 2.1 satisfies:

$$E(0) = E^{(1)}(0) = \ldots = E^{(2L-1)}(0) = 0. \tag{2.62}$$

Proof: By defining the functions

$$G_n(\omega) = \hat{\tilde{\varphi}}(\omega)^* \hat{\varphi}(\omega + 2n\pi),\ n \in \mathbb{Z} \tag{2.63}$$

we observe that equation (2.57) is also equivalent to

$$\begin{gathered} G_0(0) = 1,\ G_0^{(k)}(0) = 0,\ k = 1, \ldots, L-1, \\ G_n^{(k)}(0) = 0,\ k = 0, \ldots, L-1,\ n \neq 0. \end{gathered} \tag{2.64}$$

Also the error kernel (2.54) can now be re-written as

$$E(\omega) = \big(1 - G_0(\omega)\big)\big(1 - G_0(\omega)^*\big) + \sum_{n\neq 0} G_n(\omega)G_n(\omega)^* \tag{2.65}$$

We now prove the claimed equivalence.

(i) ⇒(ii): Let $k \in \mathbb{N}$. Using Leibniz's rule and (2.65) we have:

$$\begin{aligned} E^{(k)}(0) \;&=\; \sum_{m=0}^{k} \left[\binom{k}{m} \cdot \frac{d^m}{d\omega^m}\Big[\big(1 - G_0(\omega)\big)\Big]_{\omega=0} \cdot \frac{d^{k-m}}{d\omega^{k-m}}\Big[\big(1 - G_0(\omega)^*\big)\Big]_{\omega=0} \right] + \\ &\quad + \sum_{n\neq 0}\sum_{m=0}^{k} \binom{k}{m} G_n^{(m)}(0) G_n^{(k-m)}(0)^*. \end{aligned} \tag{2.66}$$

In the terms of the sum above, if $k < 2L$ then either $m < L$ or $k - m < L$. Using (2.64) we obtain $E^{(k)}(0) = 0$.

(ii) ⇒(i): We want to prove that equations (2.64) hold. Let us start proving for $G_n^{(0)}(0) = G_n(0)$:

$$\begin{aligned} 0 = E(0) = \big|1 - G_0(0)\big|^2 + \sum_{n\neq 0} \big|G_n(0)\big|^2 \Rightarrow \\ \Rightarrow \big|1 - G_0(0)\big|^2 = 0,\; \big|G_n(0)\big|^2 = 0 \Rightarrow G_0(0) = 1,\; G_n(0) = 0. \end{aligned} \tag{2.67}$$

In order to prove for $G_n^{(1)}(0)$, we use the hipothesis on $E^{(2)}(0)$:

$$\begin{aligned}
0 = E^{(2)}(0) &= \underbrace{\left(1 - G_0(0)\right)}_{0} \frac{d^2}{d\omega^2}\left[1 - G_0^*(\omega)\right]_{\omega=0} + \\
&\quad +2\frac{d}{d\omega}\left[1 - G_0(\omega)\right]_{\omega=0} \frac{d}{d\omega}\left[1 - G_0^*(\omega)\right]_{\omega=0} + \\
&\quad +\frac{d^2}{d\omega^2}\left[1 - G_0(\omega)\right]_{\omega=0} \underbrace{\left(1 - G_0^*(0)\right)}_{0} + \\
&\quad +\sum_{n\neq 0} \underbrace{G_n(0)}_{0} \frac{d^2}{d\omega^2}\left[G_n(\omega)^*\right]_{\omega=0} + \\
&\quad +\sum_{n\neq 0} 2\frac{d}{d\omega}\left[G_n(\omega)\right]_{\omega=0} \frac{d}{d\omega}\left[G_n(\omega)^*\right]_{\omega=0} + \\
&\quad +\sum_{n\neq 0} \frac{d^2}{d\omega^2}\left[G_n(\omega)\right]_{\omega=0} \underbrace{G_n(0)^*}_{0} \\
&= 2\left|G_0^{(1)}(0)\right|^2 + 2\sum_{n\neq 0}\left|G_n^{(1)}(0)\right|^2 \\
&\Rightarrow \; G_0^{(1)}(0) = 0,\; G_n^{(1)}(0) = 0,\; n \neq 0.
\end{aligned} \tag{2.68}$$

For the rest of the proof we only need to proceed by induction: to prove that $G_n^{(k)}(0) = 0$ $(1 < k < L)$ we use $E^{(2k)}(0) = 0$, expand the binomial sum as above and use the previously obtained values for $G_n(0), G_n^{(1)}(0), \ldots G_n^{(k-1)}(0)$.

■

2.10 Asymptotic constant

The concept of approximation order gives us a way of comparing different approximation schemes. According to (2.55), the higher the approximation order, the faster the error goes to zero as $T \to 0$. But how can two methods with the same approximation order be compared? One possibility is presented in the next theorem.

Theorem II.18. *Let φ, ψ and $\boldsymbol{q}$ satisfy the admissibility conditions in section 2.3 and form an approximation scheme that has approximation order L. Also, assume the input f to belong to W_2^r, $L < r_0 = \lfloor r \rfloor$ and the error kernel E to be $2r_0$ times* *differentiable with bounded* $(2r_0+1)$*-th derivative. Then*

$$\|f-\tilde{f}\|_{L_2} = C_{asymp}\|f^{(L)}\|_{L_2} T^L + o(T^L), \tag{2.69}$$

where

$$C_{asymp} = \sqrt{\frac{E^{(2L)}(0)}{(2L)!}}. \tag{2.70}$$

Proof: The Taylor development of the error kernel E gives us

$$E(\omega) = \sum_{k=0}^{r_0} \frac{E^{(2k)}(0)}{(2k)!}\omega^{2k} + o(\omega^{2r_0}) \tag{2.71}$$

(the even derivatives vanish since it is possible to prove that $E(\omega)=E(-\omega)$). Pugging this result into the expression (2.53), we obtain:

$$\begin{aligned} \|f-\tilde{f}\|_{L_2}^2 &= \int_{-\infty}^{\infty} |\hat{f}(\omega)|^2 E(T\omega)\,\mathrm{d}\omega + o(T^{2r_0}) \\ &= \int_{-\infty}^{\infty}\sum_{k=0}^{r_0} \frac{E^{(2k)(0)}}{(2k)!} T^{2k} |\hat{f}(\omega)|^2 \omega^{2k}\,\mathrm{d}\omega + o(T^{2r_0}) \\ &= \sum_{k=0}^{r_0} \frac{E^{(2k)(0)}}{(2k)!} T^{2k} \int_{-\infty}^{\infty} |\omega^k \hat{f}(\omega)|^2\,\mathrm{d}\omega + o(T^{2r_0}) \\ &= \sum_{k=0}^{r_0} \frac{E^{(2k)(0)}}{(2k)!} T^{2k} \int_{-\infty}^{\infty} \left|\widehat{f^{(k)}}(\omega)\right|^2\,\mathrm{d}\omega + o(T^{2r_0}) \end{aligned} \tag{2.72}$$

Since $r_0 > L$ and quasi-biorthonormality of order L holds (and then (2.62) holds) we have

$$\begin{aligned} \|f-\tilde{f}\|_{L_2}^2 &= \frac{E^{(2L)(0)}}{(2L)!}\|f^{(L)}\|_{L_2}^2 T^{2L} + o(T^{2L}) + o(T^{2r_0}) \\ &= \frac{E^{(2L)(0)}}{(2L)!}\|f^{(L)}\|_{L_2}^2 T^{2L} + o(T^{2L}) \end{aligned} \tag{2.73}$$

and thus

$$\|f-\tilde{f}\|_{L_2}^2 = \sqrt{\frac{E^{(2L)}(0)}{(2L!)}}\|f^{(L)}\|_{L_2}T^L + o\big(T^L\big). \tag{2.74}$$

■

This result was first proven in (*Blu and Unser*, 1999b) and we have simply rederived it here in a different way. The expression in (2.69) has a direct interpretation: for two approximation schemes with the same order, the one with the smallest asymptotic constant C_{asymp} will be the one with the error going faster to zero as $T \to 0$.

We observe that, following the derivation in the proof of theorem II.17, it is possible to obtain a more clear expression for C_{asymp} since

$$E^{(2L)}(0) = \sum_{k\in\mathbb{Z}} \big|G_n^{(L)}(0)\big|^2, \tag{2.75}$$

where the functions G_n are given by (2.63). Applying this to the particular case of quasi-interpolation ($\psi = \delta$, $\boldsymbol{q}$ and φ given digital filter and generator), we obtain

$$E^{(2L)}(0) = \sum_{k\in\mathbb{Z}} \big|\widehat{\varphi_{QI}}^{(L)}(0)\big|^2 \tag{2.76}$$

where $\widehat{\varphi_{QI}}(\omega) = \hat{\boldsymbol{q}}(\omega)\cdot\hat{\varphi}(\omega)$. This particular result was first presented by *Unser and Daubechies* (1997). In the same paper the authors also derive this constant for the orthogonal projection scheme:

$$E^{(2L)}(0) = \sum_{k\neq 0} \big|\widehat{\varphi}^{(L)}(0)\big|^2. \tag{2.77}$$

2.11 Chronology of approximation schemes

Based on the theory we have just presented, many approaches to obtain better schemes for sampling and reconstruction were proposed. We now review the ones we consider the most important:

- (*Blu et al.*, 2001): As we have already pointed out, the cost of the approximation scheme is mostly dictated by the support of the generator φ. With this observation and the claim that higher approximation order leads to better results the authors provide the full description of the generators φ that have minimum support, for a given approximation order. They call these generators MOMS, standing for Maximal order of minimum support.

  Theorem II.19. *For a given approximation order L, the smallest support generator $\varphi(x)$ is piecewise-polynomial of degree $L-1$ and its support is of size L. Moreover, the full class of these minimum-support functions is constrained within an L-dimensional vector space parametrized as*

  $$\varphi(x) = \sum_{n=0}^{L-1} \lambda_n \frac{d^n}{dx^n} \beta^{L-1}(x-a) \tag{2.78}$$

  where $\lambda_0 = 1$, and where a is an arbitrary shift parameter corresponding to the lower extremity of the support of $\varphi(x)$.

  Proof: Appendix A from (*Blu et al.*, 2001).

  ■

  Then the authors plug (2.78) into (2.77) and minimize the asymptotic constant with respect to λ_1, λ_2, ..., λ_{L-1}. The resulting generators are called O-MOMS (optimal MOMS).

- (*Thévenaz et al.*, 2000): The authors use the asymptotic constant to compare cardinal interpolation schemes (see section 2.7). They conclude that, although the O-MOMS were designed to reach optimality for the orthogonal projection, they are also the best available for cardinal interpolation. The second best was cardinal-interpolation with B-splines.

- (*Blu et al.*, 2004): According to (2.78) we can shift the reconstruction scheme by a and keep the same approximation order. This paper does this for the linear reconstruction scheme, it shifts the hat function by a. Then it finds the optimal shift by minimizing the asymptotic constant for the cardinal-interpolation scheme. Surprisingly, this scheme reaches the orthogonal projection asymptotic constant for reconstruction with the hat function.

- (*Condat et al.*, 2005): The authors fix the generator $\varphi = \beta^n$ and pre-filter $\psi = \delta$ (then they search for a quasi-interpolation scheme) and look for an IFIR digital filter $\boldsymbol{q}$. The criterium they adopt is

$$E(\omega) \approx E_{ls}(\omega), \quad \text{for } \omega \approx 0, \tag{2.79}$$

  where E_{ls} is the error kernel for the orthogonal projection scheme.

- (*Dalai et al.*, 2006): Take an approach very similar to (*Condat et al.*, 2005), but search for a FIR digital filter $\boldsymbol{q}$, instead of IFIR.

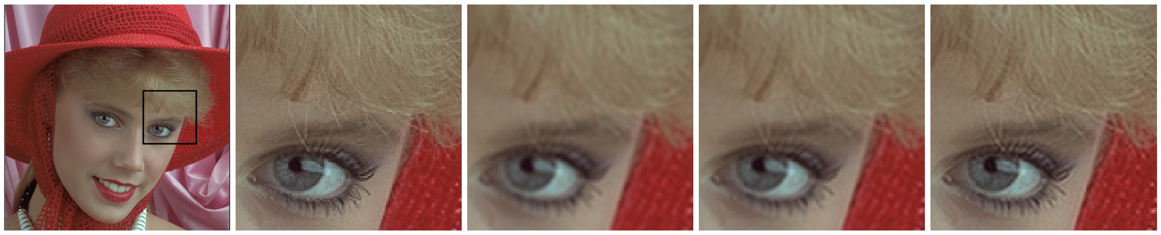
2.12 Concluding remarks

As we can see, the most important previous work adopted asymptotic criteria to design their schemes: either high approximation order, small asymptotic constant or error kernel small for small frequencies. Based on the observation that the asymptotic regime is not achievable in some contexts (such as image processing applications, where the sampling pixel grid if fixed), we propose a new scheme in the next chapter. Our results present higher quality when compared to all previous schemes.

The exposition in this chapter can be greatly complemented by (*Nehab and Hoppe*, 2014). Section 10 of this reference presents a lot of experiments qualitatively comparing different schemes (we focused only on quantitive and theoretical comparisons). Implementations details such as how to perform inverse discrete convolution (their section 4.2) are also given. Other practical details are presented in section 8.

CHAPTER III

Optimized Quasi-Interpolators for Image Reconstruction



(a) Input image and detail (b) *Blu et al.* (2001) (c) *Condat et al.* (2005) (d) Our result

Figure 3.1: Comparison between state-of-the-art quadratic quasi-interpolators with similar computational cost. The test consists of applying 40 cumulative translations to the input. Our new quadratic is better at preserving detail. PSNR: (a) ∞, (b) 32.938, (c) 34.149, (d) 36.443.

3.1 Introduction

The problem of obtaining an estimate for the value of a function at an arbitrary point, when given only a discrete set of sampled values, has a long history in applied mathematics (*Meijering*, 2002). A variety of operations commonly performed on images, such as rotations, translations, warps, and resolution change, require resampling. Efficient, high-quality reconstruction is therefore of fundamental importance in computer graphics and image processing applications.

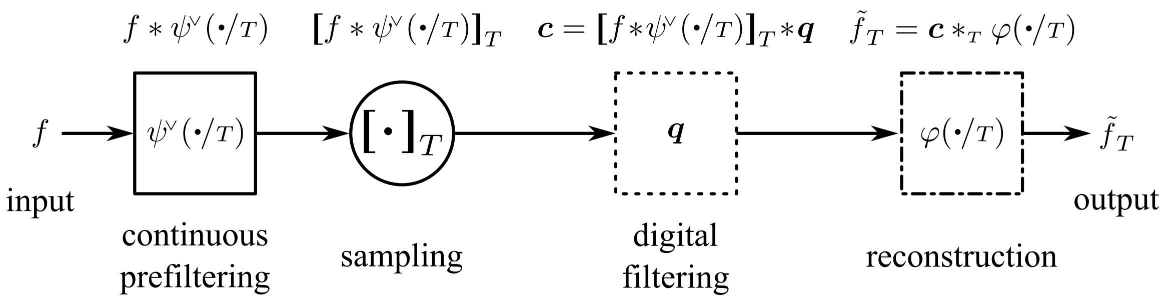


Figure 3.2: The sampling pipeline. The input f is first convolved with a scaled prefilter ψ and then sampled on a grid with fixed spacing T. We assume the image is available to us after sampling and we optimize for the quality of the reconstruction $\tilde{f}_T$ using all the degrees of freedom of the digital filter $\boldsymbol{q}$ and of the generator φ.

In this paper, we leverage recent results from the intersection of image processing and approximation theory to optimize for a new family of reconstruction schemes. Figure 3.1 shows a typical benchmark used to evaluate reconstruction quality. An input image is repeatedly translated so as to accumulate the errors due to multiple compound reconstruction steps. The figure compares the two best performing quadratic reconstruction schemes with the result of our method. Visual inspection suggests our method is better at preserving high-frequency content, and this is confirmed quantitatively by the perceptual SSIM (*Wang et al.*, 2004) metric as well as the PSNR metric. This success is the result of the greater number of degrees of freedom and the more realistic objective function we use in our optimization framework.

Figure 3.2 shows the modern approach to sampling and reconstruction (*Blu et al.*, 1999). The precise definition of each stage in the pipeline was given in section 2.2. Intuitively, an approximation $\tilde{f}_T$ to f is obtained as follows. In the first two stages, f is subjected to a continuous convolution with an *analysis filter* ψ (a.k.a. *prefilter*), and then sampled with constant sample spacing T. The traditional role of the prefiltering stage is to eliminate from f frequencies above the Nyquist rate $\frac{0.5}{T}$ so as to avoid aliasing in the sampled sequence. In the applications we discuss in this paper, we assume no knowledge or control over the prefilter ψ. In other words, either there was no prefilter (equivalently, $\psi = \delta$, the Dirac delta) or our goal is to approximate a previously prefiltered signal $f * \psi^\vee$ instead of f itself.

The remaining stages apply a *digital filter* $\boldsymbol{q}$ to the samples (by discrete convolution), and then build $\tilde{f}_T$ by combining shifted copies of a *generating function* φ, each scaled by a filtered sample. The digital filtering stage $\boldsymbol{q}$ is a recent addition to the sampling pipeline (*Unser*, 2000). It brings several advantages: it increases the range of approximation techniques that can be expressed, and gives more freedom to the design of generators φ with desirable approximation properties. Furthermore, it incurs no significant performance penalty.

The ideal sampling of *Shannon* (1949) is represented in the sampling pipeline by setting both the prefilter and generating function to the ideal low-pass filter (i.e., $\psi = \varphi = \text{sinc}$, the *sinus cardinalis*), and omitting the digital filtering stage (or equivalently, setting $\boldsymbol{q} = \boldsymbol{\delta}$, the Kronecker delta). For reasons that include its wide support even when windowed, its high computational cost, and results with an excessive amount of ringing, sinc has progressively lost favor to narrowly supported piecewise polynomial kernels, which bring performance and quality advantages (*Meijering et al.*, 2001).

A typical use for the digital filtering stage is in interpolation. Absent $\boldsymbol{q}$, the interpolation property eliminates degrees of freedom from φ that could be used for better purposes. These constraints can be moved to $\boldsymbol{q}$ instead (*Thévenaz et al.*, 2000), as in the case of interpolation by B-splines (*Unser et al.*, 1991). Another use for the digital filtering stage is in obtaining the best approximation of f for a given choice of generator φ. Since the functions of interest in many applications have considerable bandwidth outside the Nyquist interval $(-\frac{0.5}{T}, \frac{0.5}{T})$, there is no hope of reaching an exact reconstruction. The goal is instead to minimize the L_2 norm of the residual $\|f - \tilde{f}_T\|_{L_2}$, and this goal uniquely determines both ψ and $\boldsymbol{q}$. Setting $\boldsymbol{q}^\vee * \psi = \mathring{\varphi}$, the dual of the generator φ, we obtain the orthogonal projection of f into the space of functions spanned by shifted copies of the generator φ.

The order with which the residual of the orthogonal projection vanishes as we progressively reduce the sample spacing T is a property of the generating function φ (*Strang and Fix*, 1971). Piecewise polynomial generators with minimal support and optimal approximation order were completely characterized by *Blu et al.* (2001). To achieve the same approximation order as the orthogonal projection without access to ψ, as is our assumption, we must wisely select $\boldsymbol{q}$ (*de Boor*, 1990; *Blu and Unser*, 1999b). The results are *quasi-interpolators.* Optimal order digital filters $\boldsymbol{q}$ for a given φ have also been obtained (*Blu and Unser*, 1999a; *Condat et al.*, 2005; *Dalai et al.*, 2006).

Previous work has either assumed orthogonal projection and optimized for the generator, or assumed a given generator and optimized for the digital filter. As far as we know, our work is the first to jointly optimize all degrees of freedom in $\boldsymbol{q}$ and φ for the best quasi-interpolating scheme. Furthermore, inspired by *Schaum* (1993), our optimization framework takes into account the entire Nyquist interval. This is in contrast to the dominant strategy of focusing on the asymptotic behavior of the residual in the limit as $T \to 0$.

Our work results in a new family of piecewise polynomial quasi-interpolating schemes. Each scheme is given by a combination of digital filter $\boldsymbol{q}$ and generating function φ, and is optimal with regard to our metric. We run a variety of empirical tests to demonstrate that the resulting schemes yield superior reconstruction quality in typical tasks, when compared to the state-of-the-art, while maintaining competitive performance.

In addition to a historical review of related work, section 3.2 presents important concepts from approximation theory and the motivation for our work. Section 3.3 delves deeper into theory and substantiates our motivation with concrete examples. Section 3.4 presents our optimization framework. The resulting interpolators and comparisons against the state-of-the-art appear in section 3.5. We conclude in section 3.7 with directions for future research on this topic.

3.2 Related work

The classic result by *Shannon* (1949) states that the scheme described in figure 3.2 can result in $\tilde{f}_T = f$ when f's bandwidth is restricted to the Nyquist interval (-$\frac{0.5}{T}$, $\frac{0.5}{T}$) when we select $\psi = \delta$, $\boldsymbol{q} = \delta$ and $\varphi = \text{sinc}$. In the practical setting, when input functions are not necessarily band-limited and φ is required to be piecewise polynomial and compactly supported, Shannon's result does not apply and $f \neq \tilde{f}_T$.

For such cases, *Strang and Fix* (1971) established conditions under which the error $\|f - \tilde{f}_T\|_{L_2}$ goes to zero as a power of the sampling spacing T, when $\tilde{f}_T$ is the orthogonal projection of f into V_φ. They prove that $\tilde{f}_T$ has approximation order L if and only if V_φ contains all polynomials up to degree $L - 1$. They also prove that interpolating f in V_φ has the same approximation order. *Unser* (1996) proved that if ψ is $\boldsymbol{q} * \varphi$ are bi-orthogonal then the scheme in figure 3.2 has the same approximation order of φ. *Blu and Unser* (1999b) completed the characterization of approximation order L by proving it to be equivalent to ψ and $\boldsymbol{q} * \varphi$ being quasi-biorthonormal of order L.

Over the years, many different approaches have been proposed for designing good generators φ. When only samples of f are available ($\psi = \delta$), the typical choice is to design interpolating generators φ, with the claim that this would lead to better approximations (*Keys*, 1981; *Schaum*, 1993; *Dodgson*, 1997; *German*, 1997). These early works did not include the digital filter $\boldsymbol{q}$ in the approximation scheme (equivalent to taking $\boldsymbol{q} = \delta$).

Interpolation has the same approximation order L as φ (*Strang and Fix*, 1971). *Unser et al.* (1991) and *Blu et al.* (1999) advocate that the interpolation condition $\tilde{f}_T(k) = f(k), \forall k \in \mathbb{Z}$ is best enforced by the introduction of a digital filter $\boldsymbol{q} = \boldsymbol{[}\varphi\boldsymbol{]}^{-1}$ to the sampling pipeline. This digital filtering stage can be performed very efficiently (*Unser et al.*, 1991; *Nehab et al.*, 2011). The addition of the digital filtering stage leaves more freedom to design φ for increased approximation quality (*Thévenaz et al.*, 2000). The popular example is interpolation using B-splines ($\varphi = \beta^n$). B-splines have high approximation order, short support, and high regularity (*Unser*, 1999). The *cardinal* B-splines $\beta^n_{int} = [\![\beta^n]\!]^{-1} * \beta^n$ converge to sinc as n goes to infinity (*Aldroubi and Unser*, 1994a). Additionally, they are very efficient to use as pre-filters (*Heckbert*, 1986), digital filters (*Unser et al.*, 1993), and generators (*Sigg and Hadwiger*, 2005).

Explicit formulas for the asymptotic constant C_{asymp} (2.70) , associated with an approximation scheme were developed (*Möller et al.*, 1997; *Blu and Unser*, 1999b). *Blu et al.* (2001) parametrized all generators with minimum support and optimal approximation order in terms of B-splines and their derivatives. Using this parametrization, they obtained excellent generators that minimize the asymptotic constant of the orthogonal projection scheme (the O-MOMS). Although designed to be optimal for orthogonal projection, the O-MOMS were shown to be good cardinal interpolators as well (*Thévenaz et al.*, 2000). *Blu et al.* (2003) later provided a complete parametrization for generators φ in terms of their degree R, support W, regularity R, and order L. The general expression is a linear combination of B-splines and their convolution with certain distributions. Along this line of research, *Blu et al.* (2004) determined an optimal shift in the linear interpolation scheme such that it reaches the asymptotic constant of the orthogonal projection.

Interpolation is too strong a constraint. It completely defines the digital filter $\boldsymbol{q} = [\![\varphi]\!]^{-1}$. Giving up interpolation allows us to use the digital filter to improve reconstruction quality even further. These so called *quasi-interpolation* schemes were shown to have approximation order L whenever they exactly reproduce polynomials up to degree $L-1$ (i.e., $\tilde{f}_T(x) = f(x), \forall x \in \mathbb{R}$ if f is a polynomial of degree less than L) (*de Boor*, 1990; *Chui and Diamond*, 1990). This in turn will be true whenever the combination of prefilter and digital filter $\psi^\vee * \boldsymbol{q}$ have the same moments as the dual $\mathring{\varphi}^\vee$ up to order $L-1$ (*Blu and Unser*, 1999b). This equivalence was explored in the design of digital filters for quasi-interpolators based on B-spline generators $\varphi = \beta^n$: *Condat et al.* (2005) proposes an IFIR design, *Dalai et al.* (2006) an FIR design, and *Blu and Unser* (1999a) propose a combination of FIR and IFIR filters.

The approximation order L and constant C_{aymp} describe the asymptotic behavior of the residual as $T \to 0$. In practice, this will be the dominant effect only when we are able to reduce T arbitrarily, or when the input signal has a narrow band around zero. The main motivation for our work is our belief that neither of these conditions apply in typical image processing and computer graphics applications. A better goal is to minimize the residual under some appropriate metric. Although a perceptual metric would be ideal in some applications (*Zhang and Wandell*, 1996; *Wang et al.*, 2004), more powerful tools are available to work with the L_2 metric.

Error kernels $E(\omega)$ allow us to separate, in the computation of the value of the residual $\|f - \tilde{f}_T\|_{L_2}$, the influence of the input f and the influence of the approximation scheme. The general result states

$$\|f - \tilde{f}_T\|^2_{L_2} \approx \int_{-\infty}^{\infty} \left|\hat{f}(\omega)\right|^2 E(\omega)\,\mathrm{d}\omega. \tag{3.1}$$

Park and Schowengerdt (1983) obtained an expression for the error kernel when $\psi = \delta$, $\boldsymbol{q} = \delta$, and used it to determine optimal generators φ in the family of interpolating cubics. *Schaum* (1993) obtained a similar expression, but searched for more general generators φ and considered different classes of input spectra $\hat{f}$. A complete result in the form (3.1) for arbitrary ψ, $\boldsymbol{q}$, φ and $\hat{f}$ was obtained by *Blu and Unser* (1999b) using multiple generators, proving the equivalence between approximation order and quasi-biorthonormality. A version of (3.1) for a single generator φ (the case of interest in our work) was further detailed and analyzed by *Blu and Unser* (1999a).

Different works have had some success in optimizing for high approximation order L and low approximation constant C (which control the behavior of the residual in the limit $T \to 0$) as a proxy for lowering the magnitude of the error kernel in (3.1) (e.g., (*Thévenaz et al.*, 2000; *Blu et al.*, 2004)). In our work (see section 3.3) we provide concrete examples that show there is no direct connection between these goals. This is why we define our objective functions to minimize the expression in (3.1).

Unlike previous work, we obtain optimal quasi-interpolators by optimizing for, in addition to the degrees of freedom in the generator parametrization by (*Blu et al.*, 2003), all degrees of freedom in the digital filter. In other words, we jointly optimize for both φ and $\boldsymbol{q}$. Our generators and digital filters do not change depending on the input f, nor on the image-processing operation being performed. Interpolation schemes that are input-dependent include (*El-Khamy et al.*, 2005; *Kopf et al.*, 2013).

3.3 Theory and motivation

We base our optimization problem on the error kernel theorem presented in section 2.8, which we re-state below. It quantifies the L_2-error between the input and output functions in the approximating scheme of figure 3.2:

Theorem III.1. *Let φ, ψ and $\boldsymbol{q}$ satisfy the admissibility conditions in section 2.3. For all $f \in \mathbf{W}_2^r$ with $r > \frac{1}{2}$, the approximation error is given by*

$$\|f - \tilde{f}_T\|_{L_2} = \left(\int_{-\infty}^{\infty} |\hat{f}(\omega)|^2 E(T\omega)\,\mathrm{d}\omega \right)^{\frac{1}{2}} + e(f,T), \tag{3.2}$$

where $e(f,T) = o(T^r)$ and

$$E(\omega) = \left|1 - \left(\hat{\boldsymbol{q}}(\omega)^* \hat{\psi}(\omega)^*\right) \hat{\varphi}(\omega)\right|^2 + \left|\hat{\boldsymbol{q}}(\omega)\hat{\psi}(\omega)\right|^2 \sum_{k \neq 0} |\hat{\varphi}(\omega + k)|^2 \tag{3.3}$$

Proof: See appendix C in (*Blu and Unser*, 1999b).

■

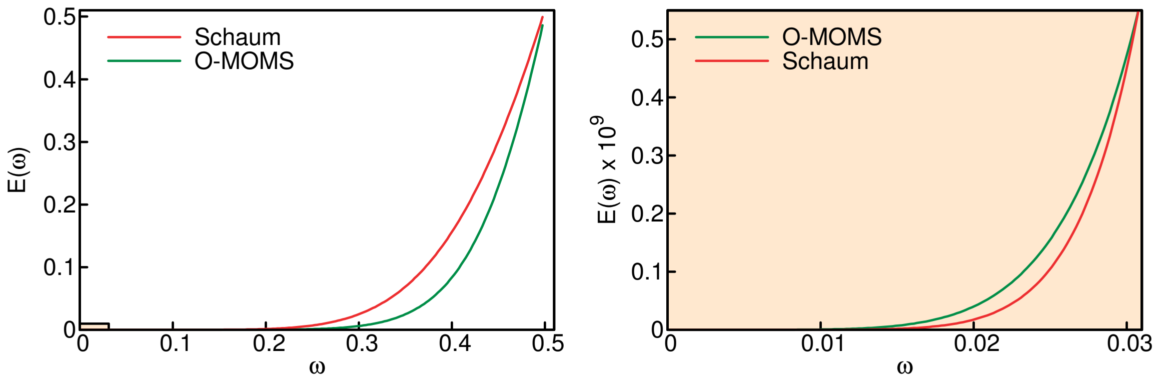
The residual term $e(f, T)$ vanishes in many situations such as in the case where f is band-limited in the Nyquist interval (*Blu and Unser*, 1999a). Setting this term aside, formula (3.2) tells us that when most of the energy of the input is concentrated at low frequencies relative to the sampling spacing (i.e., for frequencies such that $T\omega \to 0$), we can obtain a small residual by simply requiring the error kernel E to vanish near $T\omega \to 0$.

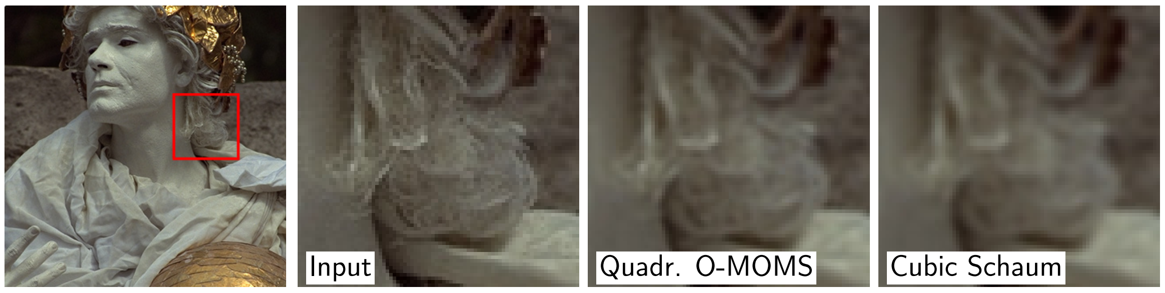
This condition is satisfied by schemes with $L > 0$. Indeed, approximation order L is equivalent to all derivatives of E up to degree $2L - 1$ vanishing at zero (theorem II.17). In turn, this causes the error kernel to behave as a polynomial of degree $2L$ near $\omega = 0$. This intuition led to significant effort being devoted towards the development of high approximation order schemes (*Keys*, 1981; *Unser*, 1996; *German*, 1997; *Blu et al.*, 2001).

When two different schemes have the same approximation order, the same intuition suggests using the asymptotic constant C_{asymp} that appears in (2.70) to select the best ones (*Blu et al.*, 2001; *Thévenaz et al.*, 2000; *Blu et al.*, 2004). This can again be related to the error kernel (3.3), since this constant is proportional to the coefficient of the leading $(2L)^{\text{th}}$ power of the polynomial approximation of the error kernel around $\omega = 0$ (*Blu and Unser*, 1999a).

We agree that a higher approximation order and small asymptotic constant can be important in many applications. However, as we show in figures 3.3 and 3.4, it is possible to find counter-examples for *both* criteria for exactly the applications that are typically used to showcase the approximation quality achieved by following them.

Figure 3.3 compares the 3rd-order (cardinal) quadratic interpolator OMOMS-2 (*Blu et al.*, 2001) with the 4th-order cubic local Lagrangian interpolator (*Schaum*, 1993) in an experiment that consists of 30 compounded rotations. At each rota-



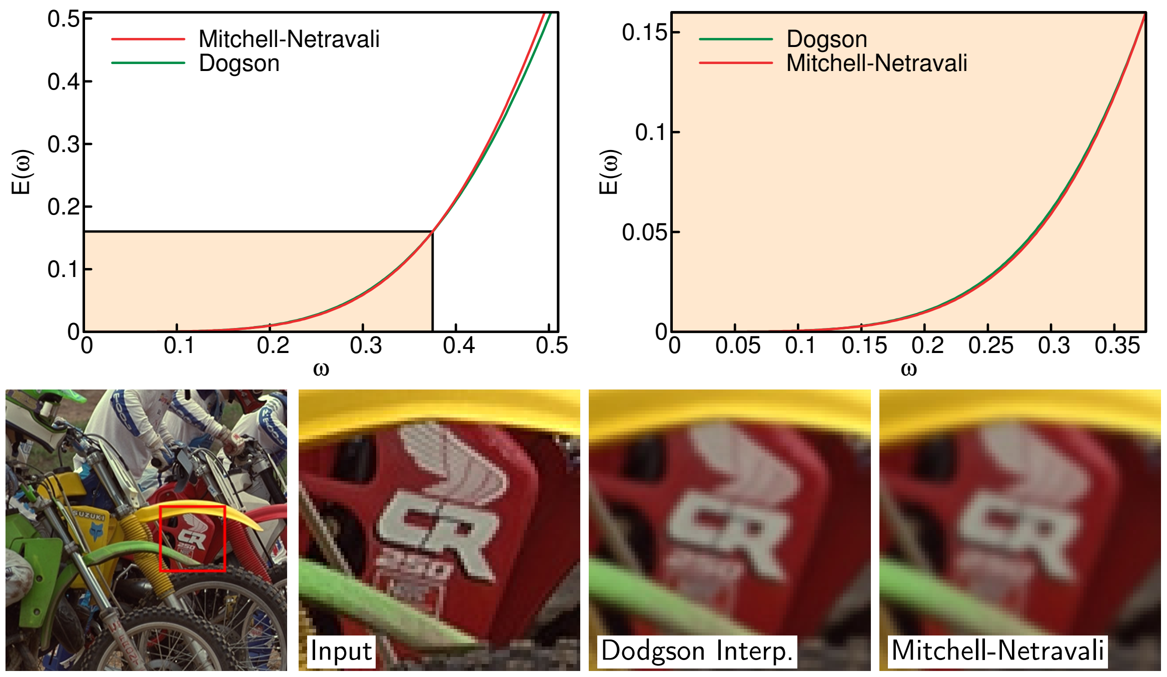


(a) PSNR $= \infty$, SSIM $=1.0$ (b) PSNR $= 32.21$, SSIM $= 0.94$ (c) PSNR $= 29.86$, SSIM $= 0.90$

Figure 3.3: Comparison between the quadratic O-MOMS, a 3rd-order interpolator proposed by (*Blu et al.*, 2001), and a 4th-order cubic by *Schaum* (1993). Even with its lower order, O-MOMS's error kernel shows a better behavior overall in most of the Nyquist interval (top left). Detail (top right) shows that Schaum's is only better for a tiny portion of the spectrum near the origin. Comparison of 30 consecutive rotations confirm the better approximation qualities of the O-MOMS interpolator.

tion step, the input image is interpolated, and sampled at a $\frac{360^o}{30} = 12^o$ angle. The result is used as input for next rotation step and so on until the image is back to its initial position, at which point it is compared to the original input. The PSNR and SSIM (*Wang et al.*, 2004) measures are higher (meaning higher quality) for the result with OMOMS-2 (Figure 3.3b), that has a lower order. The plot of the error kernels for both approximation schemes (Figure 3.3 top left) show that OMOMS-2 has a smaller value overall in the full Nyquist interval although it is worse for low frequencies (Figure 3.3 top right), the latter behavior being expected since it has lower approximation order.

In figure 3.4, we compare the performance of the quadratic interpolator designed



(a) PSNR $= \infty$, SSIM $=1.0$ (b) PSNR $= 24.48$, SSIM $= 0.84$ (c) PSNR $= 22.88$, SSIM $= 0.76$

Figure 3.4: Comparison between a quadratic interpolator proposed by *Dodgson* (1997) and the cubic by *Mitchell and Netravali* (1988) (not interpolating), both with approximation order 2. Error kernels show the overall better behaviour of Dodgson's interpolator in the full Nyquist interval. This is despite its poorer behaviour near the origin (top right), as predicted by its higher asymptotic constant. Comparison of 15 consecutive translations show the higher quality achieved by Dodgson's interpolator.

by *Dodgson* (1997) with the (non-interpolating) cubic proposed by *Mitchell and Netravali* (1988). Both these kernels have approximation order 2, so we would expect the one with smaller asymptotic constant to be better (the formula for the constant is provided in (*Blu and Unser*, 1999b)). In this case the constant for Dodgson's interpolator is slightly larger than Mitchell-Netravali's cubic's (by about 0.0004). Nevertheless, the compounded 15-translations in figure 3.4 show that Dodgson's interpolator generates a better result (Figure 3.4b). This is again due to a better behaviour in the full Nyquist interval (Figure 3.4 top left), although it is a bit worse for low frequencies (Figure 3.4 top right).

These counter-examples exist because the benchmarks violate the underlying assumption that the input frequency content is concentrated around $T\omega \to 0$. As is obvious from the images, they have significant frequency content away from $\omega \to 0$. Indeed, the input power spectrum for natural images tend to behave as

$$\left|\hat{f}(\omega)\right|^2 \approx \frac{1}{\omega^p}, \tag{3.4}$$

where p varies from 1.6 to 3.0 (*Field and Brady*, 1997; *Ruderman*, 1997; *Hsiao and Millane*, 2005). A photograph taken underwater tends to be blurrier, so p will be higher. A photograph taken in the woods, where foliage produces high-frequency content, will have a smaller p. While the idea of taking $T \to 0$ *is* valid for numerical analysis applications that control the sampling spacing, we are not afforded the same freedom in most image-processing applications: we must therefore analyze the error kernel E over the entire frequency domain.

Recall we assume we only have access to the samples of f. If f was filtered by a good prefilter prior to sampling, the frequency content outside the Nyquist interval is close to zero. If not, whatever frequencies were outside the Nyquist interval have already been aliased back into it when the image was sampled. Therefore, rather than integrating on the real line as in (3.2), we focus on the Nyquist interval:

$$\|f - \tilde{f}_T\|_{L_2}^2 \approx \int_{-\frac{0.5}{T}}^{\frac{0.5}{T}} \left|\hat{f}(\omega)\right|^2 E(T\omega)\,\mathrm{d}\omega. \tag{3.5}$$

Since T is fixed, we may assume $T = 1$ with no loss of generality (see section 3.9 for proof).

We can now define our minimization problem:

$$\arg\min \int_{-0.5}^{0.5} \left|\hat{f}(\omega)\right|^2 E(\omega)\,\mathrm{d}\omega. \tag{3.6}$$

Assuming $\hat{f}$ known (to be detailed in section 3.4) and no prefiltering in the quasi-interpolation scheme ($\psi = \delta$ or $\hat{\psi} \equiv 1$), the degrees of freedom lie in the definitions of the digital filter $\boldsymbol{q}$ and the generator φ.

We explore three options for the form of digital filter $\boldsymbol{q}$, FIR, IFIR, and FIR-IFIR. Formally,

$$\text{FIR}: \ \boldsymbol{q} = [\ldots, 0, d_{\text{-}j}, \ldots, d_0, \ldots, d_j, 0, \ldots], \tag{3.7}$$

$$\text{IFIR}: \ \boldsymbol{q} = [\ldots, 0, e_{\text{-}k}, \ldots, e_0, \ldots, e_k, 0, \ldots]^{\text{-}1}, \quad \text{and} \tag{3.8}$$

$$\text{FIR-IFIR}: \ \boldsymbol{q} = \boldsymbol{d} * \boldsymbol{e}. \tag{3.9}$$

These formulations provide us with $2j+1$, $2k+1$, and $2(j+k+1)$ degrees of freedom, respectively.

To isolate the degrees of freedom in the generator in a meaningful way, we use the parametrization by *Blu et al.* (2003) in terms of its degree N, support W, regularity R, and approximation order L (for simplicity, we write $\varphi \in \{N, W, L, R\}$).

Theorem III.2. *Given $W \geqslant N$, $\varphi \in \{N, W, R, L\}$ if and only if there exists a unique set of coefficients $a_{k,\ell}$, $b_{k,\ell}$, and $c_{k,\ell}$ such that*

$$\begin{aligned}\varphi\big(x - \tfrac{W}{2}\big) = &\sum_{\ell=1}^{M} \sum_{k=0}^{N\text{-}L\text{-}\ell} a_{k,\ell} \left(\beta_{nc}^{L+k\text{-}1} * \gamma_{\ell}^{N\text{-}L\text{-}k}\right)(x) \\ &+ \sum_{\ell=0}^{M} \sum_{k=0}^{W\text{-}N+\ell\text{-}1} b_{k,\ell}\, \beta_{nc}^{N\text{-}\ell}(x-k) \\ &+ \sum_{k=0}^{W\text{-}L} \sum_{\ell=0}^{L\text{-}R\text{-}2} c_{k,\ell}\, \Delta^{*\ell}\, \beta_{nc}^{L\text{-}\ell\text{-}1}(x-k),\end{aligned} \tag{3.10}$$

where $M = N - \max(R+1, L)$.

Proof: See (*Blu et al.*, 2003).

$\blacksquare$

In the formulas above,

$$\beta^n_{nc}(x) = \beta^n\big(x - \tfrac{n+1}{2}\big), \tag{3.11}$$

is the non-centered B-spline, $\Delta^{*\ell}$ is the ℓ^{th}-order finite difference, and γ^n_ℓ are distributions (e.g., derivatives and shifts).

For example, setting $N = 1$, $W = 2$, $R = -1$ (meaning φ is bounded), and $L = 1$ in the decomposition theorem produces

$$\varphi(x) = b_{0,0}\beta^1(x) + c_{0,0}\beta^0\big(x + \tfrac{1}{2}\big) + c_{1,0}\beta^0\big(x - \tfrac{1}{2}\big) \tag{3.12}$$

This gives us 3 additional degrees of freedom, relative to the common choice of $\varphi = \beta^1(x)$ (*Condat et al.*, 2005; *Dalai et al.*, 2006), with which we minimize our objective function.

3.4 Optimization

We now state the minimization problem that will result in optimal quasi-interpolators. Before the objective function itself, we detail the constraints.

Degree and width of φ The degree N is the guiding parameter in our method. We set the width to $W = N + 1$ to match the run-time efficiency of generators such as B-splines and O-MOMS.

Regularity of φ The only restriction we impose is boundedness ($R = -1$). Several authors have observed that regularity is not fundamental for good approximation quality *Schaum* (1993); *Blu et al.* (2001). Our results confirm this. Applications requiring more regularity (e.g., for derivatives) can change this parameter in the optimization.

Approximation order of φ In stark contrast to previous work, we only require first-order approximation ($L = 1$). This means that frequency $\omega = 0$ (i.e., DC or the average input value) will be preserved, but nothing else. In analogy to the regularity constraint, our results show that these additional degrees of freedom are better left to the discretion of the optimizer.

These constraints determine the coefficients in (3.10) that are available for minimization. We encapsulate them into lists of coefficients $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$:

$$\mathbf{A} = \{a_{k,\ell}\}, \quad \mathbf{B} = \{b_{k,\ell}\}, \quad \mathbf{C} = \{c_{k,\ell}\}. \tag{3.13}$$

Symmetry of φ and $\boldsymbol{q}$ To guarantee linear phase, we require our quasi-interpolators $\boldsymbol{q} * \varphi$ to be symmetric. The condition imposes simple linear relationships between the coefficients $a_{k,\ell}$, $b_{k,\ell}$, and $c_{k,\ell}$, and sets $\boldsymbol{d}_i = \boldsymbol{d}_{\text{-}i}$, and $\boldsymbol{e}_i = \boldsymbol{e}_{\text{-}i}$, for all i.

Unit scale for φ and $\boldsymbol{q}$ There is a scale ambiguity within the remaining degrees of freedom. Scaling φ by s and $\boldsymbol{q}$ by $\frac{1}{s}$ leaves the quasi-interpolator $\boldsymbol{q} * \varphi$ unchanged. We therefore impose

$$\int_{-\infty}^{\infty} \varphi(x)\,\mathrm{d}x = 1, \quad \text{and} \quad \sum_{i\in\mathbb{Z}} \boldsymbol{d}_i = \sum_{i\in\mathbb{Z}} \boldsymbol{e}_i = 1. \tag{3.14}$$

Approximation order of the scheme We also require the *scheme as a whole* to have first order of approximation. The generator φ satisfies the restriction by construction, but a misguided choice of $\boldsymbol{q}$ could ruin it. The equivalent condition on the error kernel is

$$E(0) = 0. \tag{3.15}$$

See theorem II.17 for the proof.

Objective function Recall the spectrum of natural images tend to follow (3.4). Since we seek input-independent quasi-interpolators, we set p to the intermediate value of $p = 2$:

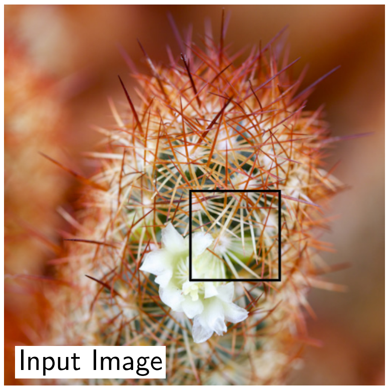
$$\left|\hat{f}(\omega)\right|^2 \approx \frac{1}{\omega^2}. \tag{3.16}$$

This choice has an extra advantage in our formulation. Since we are imposing $E(0) = 0$ and since $E'(0) = 0$ is automatically satisfied due to the symmetry of the error kernel, we have $E(\omega)$ proportional to ω^2 near the origin. This causes the integrand in (3.6) to converge to a finite value at the origin.

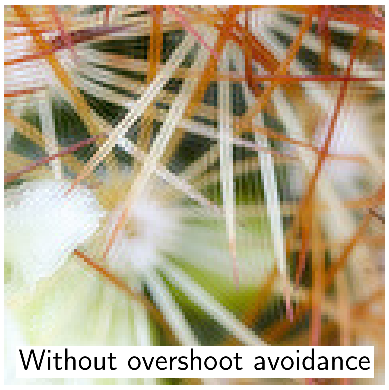
The optimization problem Given a degree N:

$$\begin{aligned}
\underset{\boldsymbol{q},\mathbf{A},\mathbf{B},\mathbf{C}}{\arg\min} \quad & F(d) := \int_0^d \frac{1}{\omega^2} E(\omega)\, \mathrm{d}\omega && (3.17)\\
\text{subject to} \quad & \varphi \in \{N, N+1, -1, 1\}, && (3.18)\\
& \varphi^\vee = \varphi, \quad \boldsymbol{q}^\vee = \boldsymbol{q}, && (3.19)\\
& \int \varphi(x)dx = 1, \quad \sum \boldsymbol{q}_k = 1, && (3.20)\\
& E(0) = 0. && (3.21)
\end{aligned}$$

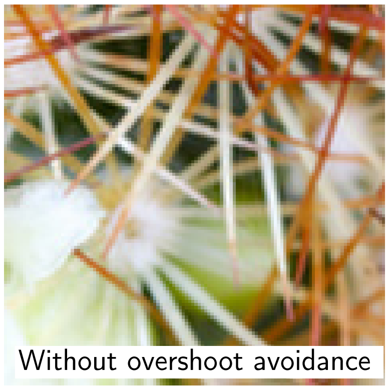
(We can restrict the integral to positive ω because of symmetry.)



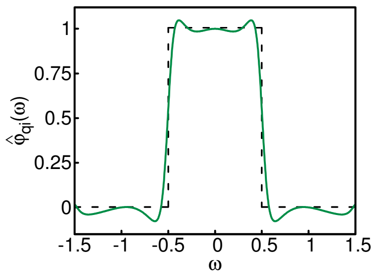
(a) PSNR = ∞, SSIM =1.0



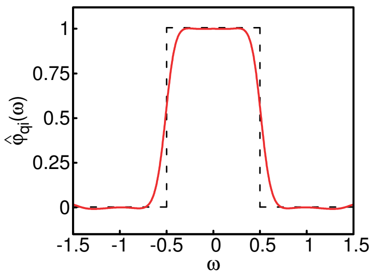
(b) PSNR = 34.662, SSIM = 0.995



(d) PSNR = 43.812, SSIM = 0.999



(c) Frequency response



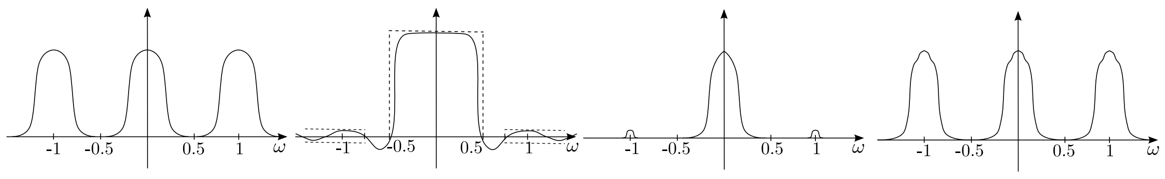
(e) Frequency response

Figure 3.5: Quadratic interpolation result for 20 compounded translations. Minimizing $F(0.5)$ leads to a quasi-interpolator φ_{qi} that overshoots high frequencies (b). This problem is avoided by minimizing $F(0.34)$ (d), where $d = 0.34$ is automatically obtained by a binary search. Plots in figures (c) and (e) show the frequency response of the associated quasi-interpolators compared with the frequency response of the ideal interpolator.

Controlling overshoot and aliasing The natural choice for the integration limit d in (3.17) is 0.5, since we only have access to samples of f. Unfortunately, this often results in quasi-interpolators with highly oscillating spectra, such as the one presented in figure 3.5c.

By minimizing (3.17) with $d = 0.5$ we are requiring the error kernel to be small near $\omega = 0.5$, say $E(0.5 - \varepsilon) \approx 0$. As shown in section 3.9, this implies

$$\begin{aligned} \hat{\varphi}_{\text{qi}}(0.5-\varepsilon) &\approx 1, \quad \hat{\varphi}_{\text{qi}}(0.5+\varepsilon) \approx 0, \\ \hat{\varphi}_{\text{qi}}(\text{-}0.5+\varepsilon) &\approx 1, \quad \hat{\varphi}_{\text{qi}}(\text{-}0.5-\varepsilon) \approx 0. \end{aligned} \tag{3.22}$$



(a) $DTFT([\![f * \psi^\vee]\!])(\omega)$ (b) $\hat{\varphi}_{\text{qi}}(\omega)$ (c) $DTFT([\![f * \psi^\vee]\!])(\omega) \cdot \hat{\varphi}_{\text{qi}}(\omega)$ (d) $DTFT([\![[\![f * \psi^\vee]\!] * \varphi_{\text{qi}}]\!])(\omega)$

Figure 3.6: By imposing $\hat{\varphi}_{\text{qi}}(\omega) \leqslant 1.0025$, $\forall\omega \in [-0.5, 0.5]$ and $|\hat{\varphi}_{\text{qi}}(\omega)| \leqslant 0.025$, $\forall\omega \in [-\infty, -0.75] \cup [0.75, \infty]$ we control overshoot and aliasing in the re-sampled image. The DTFT of the input $[\![f * \psi^\vee]\!]$ (a) is multiplied by $\hat{\varphi}_{\text{qi}}$ (b) and results in a spectrum with small out-band (aliasing) spectrum and non-amplified in-band spectrum (c). Resampling it leads to $[\![[\![f * \psi^\vee]\!] * \varphi_{\text{qi}}]\!]$ (whose DTFT is shown in figure d) with controlled frequency amplification.

Thus $E(0.5 - \varepsilon) \approx 0$ leads to $\hat{\varphi}_{\text{qi}}(\omega) = \hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega)$ that approximates a function with discontinuities near $\omega = \pm 0.5$. Since $\hat{\varphi}(\omega)$ cannot oscillate much (see (*Blu et al.*, 2003) for the expression), $\hat{\boldsymbol{q}}(\omega)$ is responsible for approximating the discontinuities near $\omega = \pm 0.5$. Since the filter has a finite support in the form of (2.4) or its reciprocal, this leads to the Gibbs phenomenon in $\hat{\boldsymbol{q}}(\omega)$, which is modulated by $\hat{\varphi}(\omega)$ and manifests itself as ringing in the reconstructed images (figure 3.5b).

To prevent this issue, we only consider quasi-interpolators that satisfy the following admissibility conditions:

$$\hat{\varphi}_{\text{qi}}(\omega) \leqslant 1.0025, \quad \forall\omega \in [-0.5, 0.5] \quad \text{and} \tag{3.23}$$

$$|\hat{\varphi}_{\text{qi}}(\omega)| \leqslant 0.025, \quad \forall\omega \in [-\infty, -0.75] \cup [0.75, \infty]\,. \tag{3.24}$$

Intuitively, condition (3.23) prevents overshoot and condition (3.24) prevents aliasing. The values 1.0025, 0.025 and 0.75 were empirically determined. To solve the optimization problem, we relax the objective function by performing a binary search for the largest value of $d \in [0, 0.5]$ in (3.17) that leads to an admissible quasi-interpolator.

Figure 3.6 illustrates the importance of conditions (3.23) and (3.24) by following the effects of each stage of the sampling pipeline, in the frequency domain. The DTFT of the input $[f * \psi^{\vee}]$ is modulated by the spectrum of the quasi-interpolator $\hat{\varphi}_{\mathrm{qi}}$. The resampling step then replicates this spectrum. Conditions (3.23) and (3.24) control magnification of both in-band and out-band spectra.

Note that condition (3.24) skips interval $(0.5, 0.75)$. In fact, for $N = 1$, even this relaxed condition is too restrictive. We therefore test only condition (3.23). Degrees $N = 2$ and 3 have larger parameter spaces, and we can find a value for d that satisfies both constraints.

The practical effect of the admissibility conditions can be seen in the example of figure 3.5. There, the quasi-interpolator that results from the optimization with $d = 0.5$ leads to overshoot in high frequencies (note ringing surrounding thorns). The binary search finds the value $d \approx 0.34$. The resulting quasi-interpolator is softer, but is still sharp enough. The overshooting is mostly gone.

Length and type of $\boldsymbol{q}$ We solve (3.17)–(3.21) using FIR, IFIR and FIR-IFIR digital filters. FIR filters led to the worst results, both w.r.t. the objective function (3.6) and our interpolation experiments. Since IFIR and FIR-IFIR formulations lead to similar results, we prefer the lower cost IFIR. The wider $\boldsymbol{q}$ is (i.e., the more degrees of freedom it offers), the lower objective function values are obtained. However, very little is gained for widths greater than 5. All our results assume a width 5 and an IFIR formulation for the digital filter.

3.5 Results and discussion

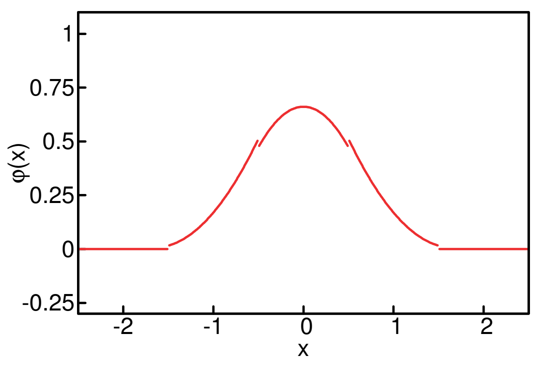
We have implemented this optimization framework in Mathematica, selecting the optimization method by *Nelder and Mead* (1965). This method is suitable for constrained non-linear problems, and worked best in practice with our objective function. To reduce the risk of finding poor local minima, we solve each optimization problem 40 times and select the best result.

The objective function is somewhat brittle, due to the integrand in (3.17) being unstable near the origin. We were careful to keep the error kernel in its simplest possible form to avoid numerical round-off errors. All calculations were performed with 20-digit precision. The Quasi-Monte Carlo method gave the most robust results for the numerical integration of (3.17).

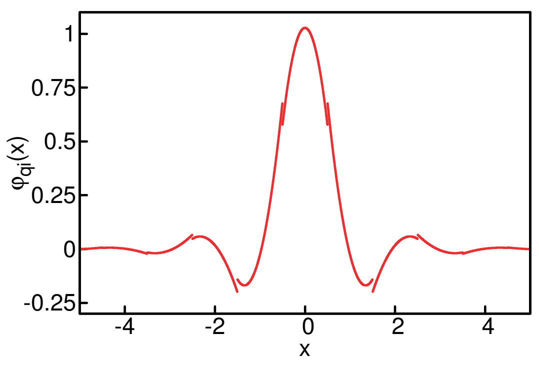
The values for all arguments in the solution to the optimization problem of degrees 1, 2, and 3 are given in section 3.8. For convenience, we also provide the source-code for the generators in the supplemental materials and the digital filter entries. We compare the practical performance of our reconstruction schemes against previous quasi-interpolators by performing a variety of experiments.

Figure 3.7 shows a plot of our quadratic generator φ (a) and the quasi-interpolator $\varphi_{\mathrm{qi}} = \boldsymbol{q} * \varphi$ (b). As has been observed in previous work, regularity is not fundamental for achieving good approximation quality. Like the local Lagrangian interpolators of *Schaum* (1993) and the OMOMS-2 of *Blu et al.* (2001), our quadratic quasi-interpolator is not even continuous. The figure also shows a comparison between the frequency response $\hat{\varphi}_{qi}$ of our quasi-interpolator with the state-of-the art in quadratics (c). It is clear our interpolator is sharper. Furthermore, the error kernel plots (d) show that our quasi-interpolator has a lower error overall in the Nyquist interval. Figure 3.8 shows the same analysis, this time for our cubic quasi-interpolator. Similar conclusions can be drawn. Please note that the improvements due to our new quasi-interpolators is more marked than the quality differences between the previous state-of-the-art.

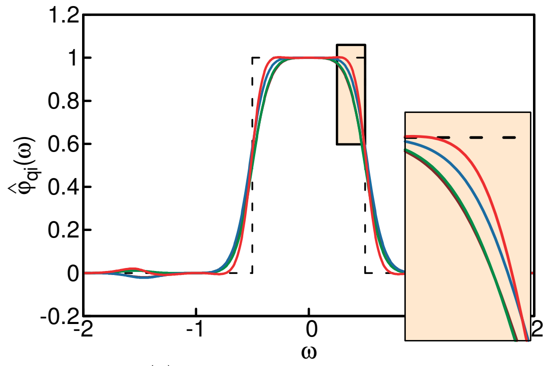
Figure 3.9 shows results for our linear quasi-interpolator. The tests add a random perturbation to each compounded translation offset in order to rule out the possibility of errors being cancelled by negative correlations. Our results are significantly sharper than those obtained by state-of-the art linear quasi-interpolator proposed by *Condat et al.* (2005). In fact, our results compare favourably even against the



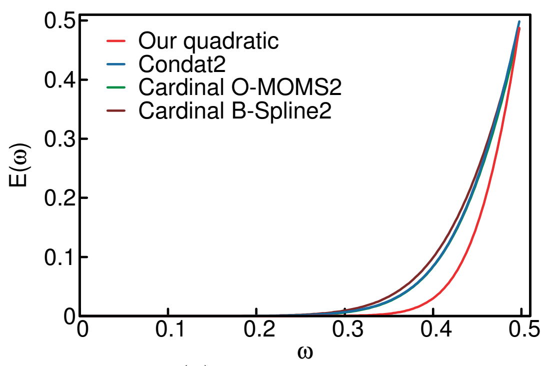
(a) Our generator



(b) The quasi-interpolator



(c) Frequency responses



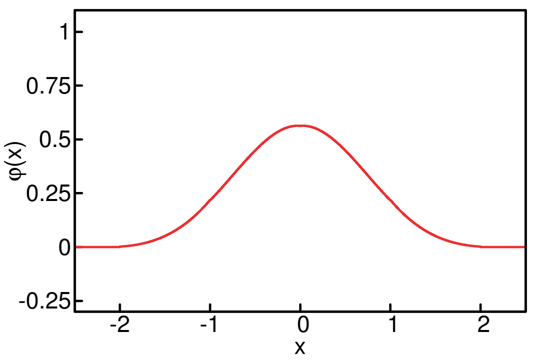
(d) Error kernels

Figure 3.7: Impulse responses of our quadratic generator (a) and quasi-interpolator (b). We compare the frequency response of our quasi-interpolator with the best quadratics (c), showing ours to be closer to the ideal interpolator. Plot in (d) shows that the error kernel associated to our method is smaller in most of the Nyquist interval.

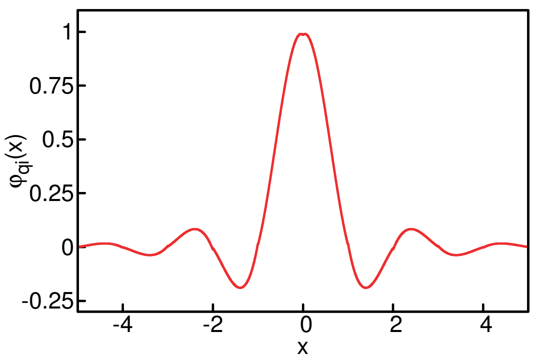
cardinal *quadratic* B-spline.

The example in figure 3.10 shows that our quadratic quasi-interpolator performs better than the one proposed by *Condat et al.* (2005). In fact, our quadratic compares favourably against the cardinal *cubic* O-MOMS, which is the state-of-the-art in cubic interpolation (*Thévenaz et al.*, 2000).

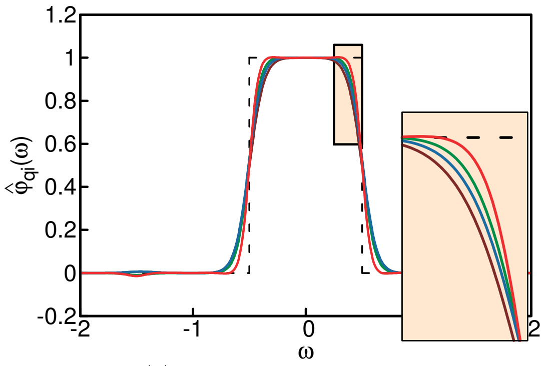
Figure 3.11 tests the performance of our cubic quasi-interpolator with a challenging task of rotating a high-frequency pattern consisting of parallel lines. Our result shows almost perfect reconstruction. The cubic quasi-interpolator proposed by *Blu and Unser* (1999a) (which uses a wider FIR-IFIR formulation) and, to a lesser extent, the quintic cardinal B-spline, show aliasing in the form of spurious slanted lines. This



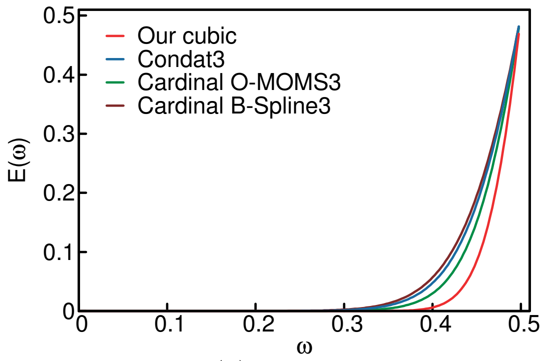
(a) Our generator



(b) The quasi-interpolator



(c) Frequency responses



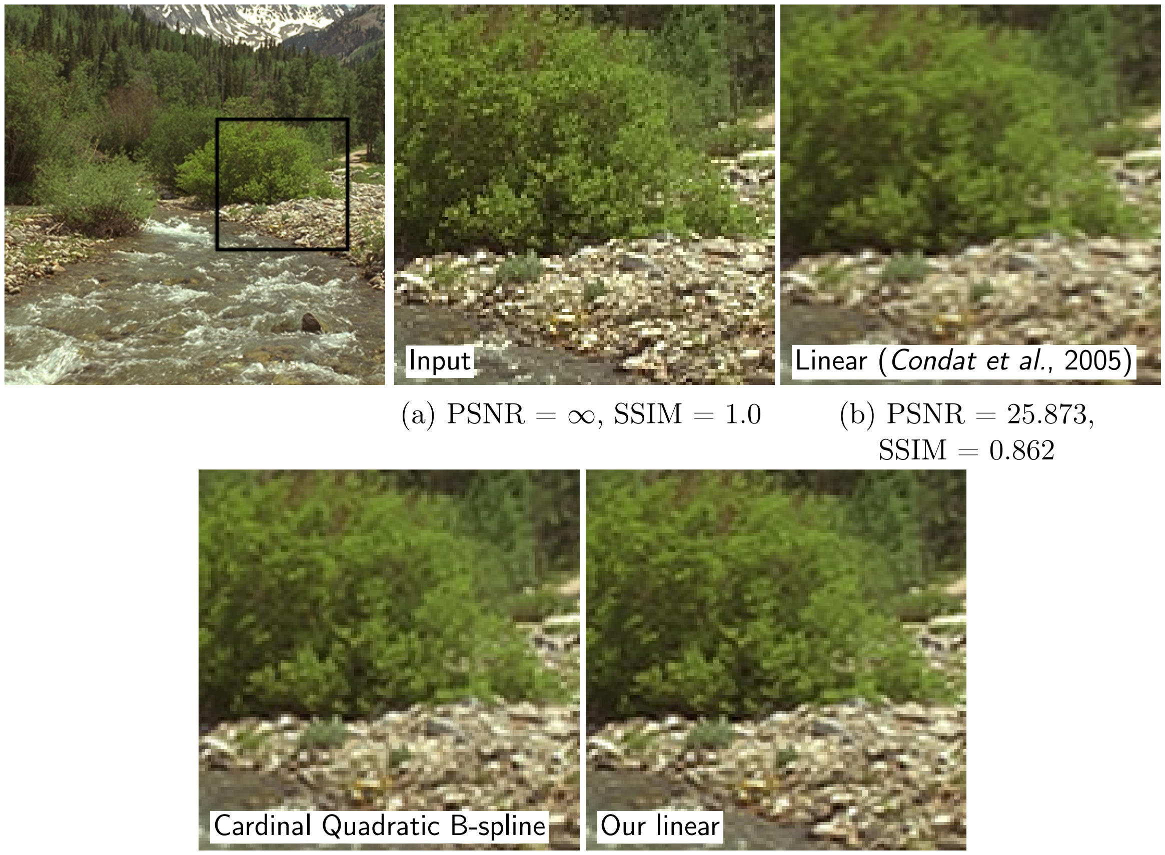
(d) Error kernels

Figure 3.8: Impulse responses of our cubic generator (a) and quasi-interpolator (b). Its frequency response compared with best cubics (c) and the associated error kernels are shown in (d).

final example helps emphasize one of the key points in our paper: the quintic cardinal B-spline has approximation order 6, and our cubic has only approximation order 1. Nevertheless, our cubic performs better.

To put all these results in context, we ran an additional experiment. We applied 90 randomized translations to the images in (*Kodak*, 2010) in such a way that after every 3 translations it goes back to the initial position. At these points, we can measure PSNR against the input. To obtain a single number, we average the PSNR results over the 24 input images. Results can be seen in figure 3.12. Our cubic quasi-interpolator performs best, even when compared to quintic quasi-interpolators. Our quadratic quasi-interpolator performed better than any other quadratic and cubic.

We recommend the use of our solutions for degrees 2 and 3, given their superior



(a) PSNR $= \infty$, SSIM $= 1.0$ (b) PSNR $= 25.873$, SSIM $= 0.862$

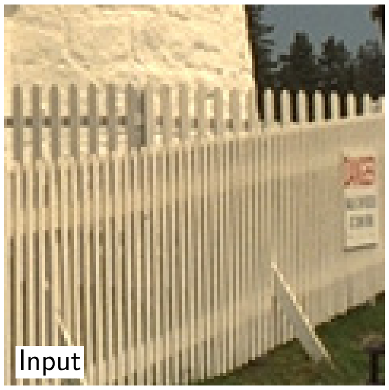
(c) PSNR $= 27.000$, SSIM $= 0.898$ (d) PSNR $= 27.642$, SSIM $= 0.913$

Figure 3.9: Result of 9 repeated (randomized) translations. Our linear quasi-interpolator (d) produces a result sharper than the one produced by one of the best linear quasi-interpolators (*Condat et al.*, 2005) (b). The output by our method is also slightly better than the one produced with the cardinal quadratic B-spline (c).

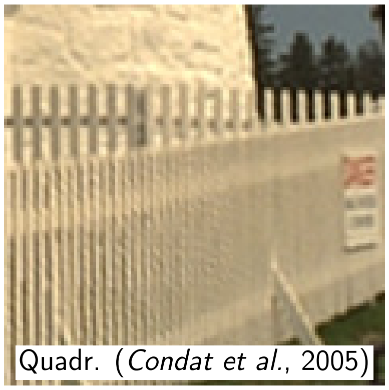
performance and moderate computational cost. Tasks requiring even more speed can use the degree 1 solution, which we take as a proof-of-concept. We refer the reader to the supplemental material for full resolution images of all these experiments. We also provide additional videos containing other interpolation sequences.

3.5.1 Limitations

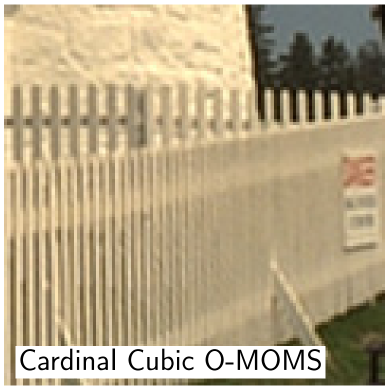
One limitation of our method can be seen in figure 3.13, which uses our linear quasi-interpolator (b). The figure shows the result of 4 compound translations by



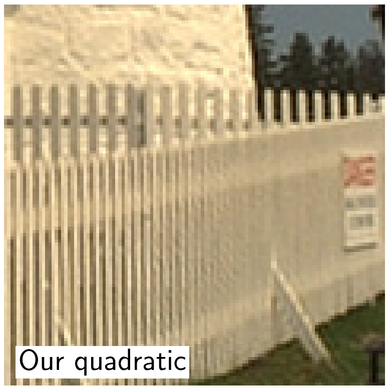
(a) PSNR = ∞, SSIM =1.0



(b) PSNR = 30.429, SSIM = 0.932



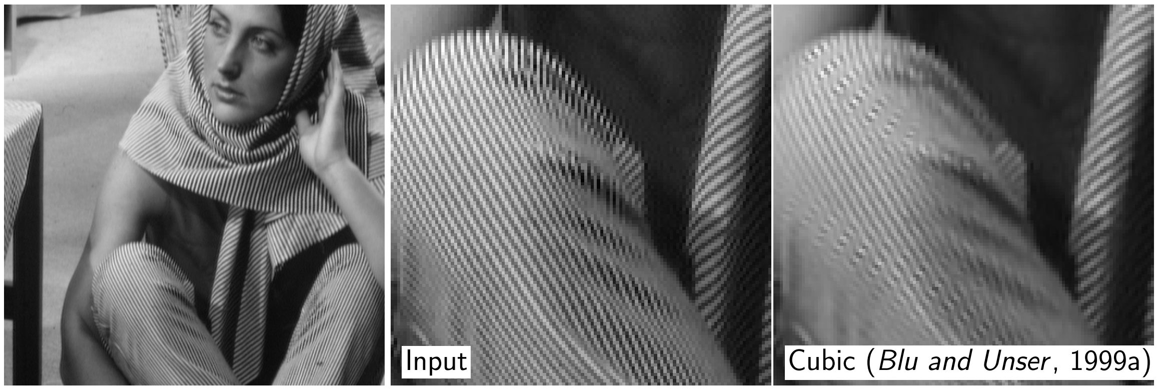
(c) PSNR = 32.221, SSIM = 0.951



(d) PSNR = 33.291, SSIM = 0.960

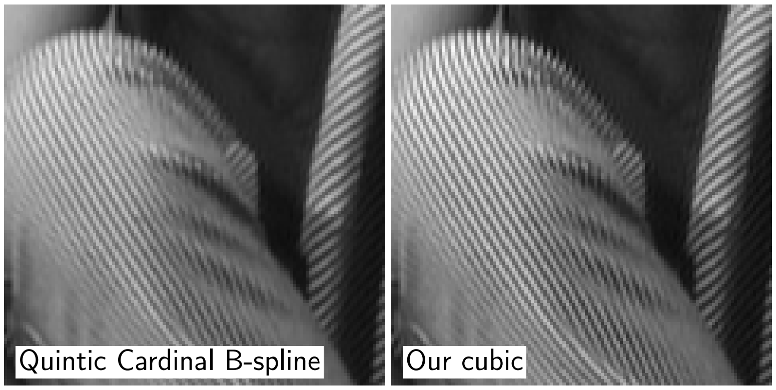
Figure 3.10: Comparison of 30 (randomized) rotations for different quasi-interpolators. The quadratic proposed by *Condat et al.* (2005) distorts the vertical aspect of the fence, while ours better preserves the geometry of the scene. Our result is competitive even if compared with the one produces by the cardinal cubic O-MOMS (considered the best cubic in the literature).

exactly half a pixel. It is clear that high-frequencies have been excessively magnified. This limitation is not specific to our approach (c). The plots in figure 3.13 explain the problem: for each translation τ, the shaded region illustrates the minimum and maximum possible frequency amplitude scaling. The worst behavior happens in the unfortunate case $\tau = 0.5$. This problem practically disappears when random translations are applied. As future work, we will incorporate new criteria in our optimization framework to reduce this effect.



(a) PSNR = ∞,
SSIM = 1.0

(b) PSNR = 13.013,
SSIM = 0.749



(c) PSNR = 13.047,
SSIM = 0.775

(d) PSNR = 13.044,
SSIM = 0.784

Figure 3.11: 40 compounded rotations. The result produced by one of the best cubic quasi-interpolators (*Blu and Unser*, 1999a) has aliasing of high frequencies. Using the cardinal quintic B-spline presents the same problem at a smaller magnitude. Our cubic almost completely removes the artefacts, while keeping the result sharp.

We have also noticed that the sharpness of our results comes at the cost additional mild ringing (for instance, see figure 3.1d). For hundreds of repeated translations, our linear and quadratic quasi-interpolators showed excessive ringing. In this extreme case, other methods presented either a similar behavior or excessive blurring.

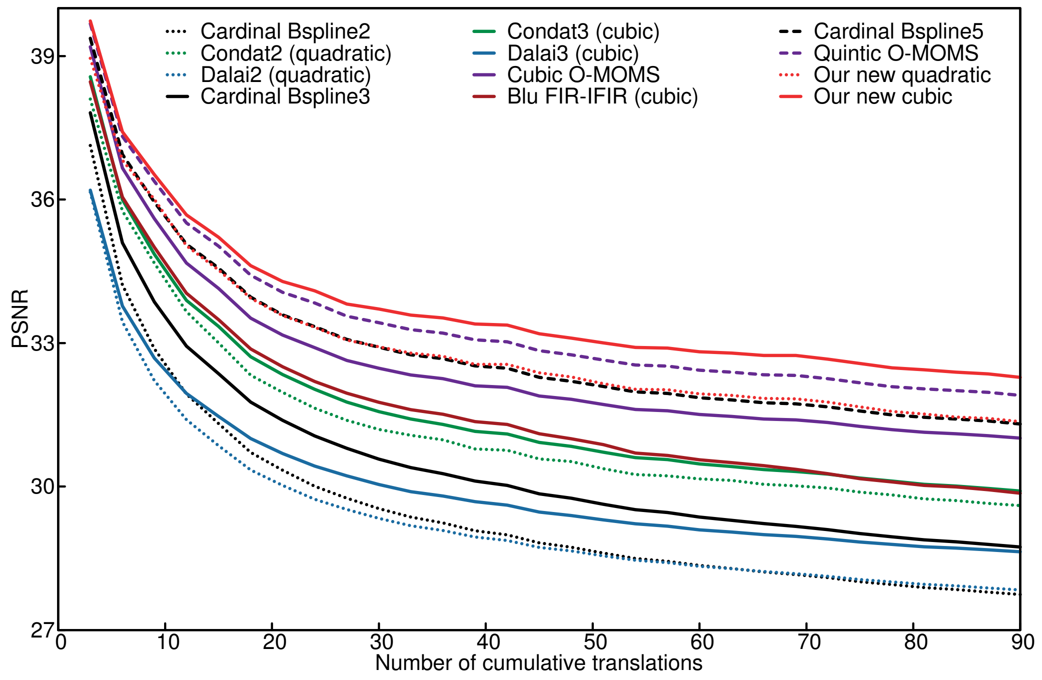


Figure 3.12: Average PSNR of applying 90 randomized translations to 24 input images. Translations were applied in a way that after every 3 translations, the image was back to its initial position and we could measure PSNR. The best quasi-interpolators in the literature were compared. Our new cubic quasi-interpolator (solid red) reaches the best quality, better than the *quintic* O-MOMS (dashed purple). Our new quadratic (dotted red) reaches higher quality than any other quadratic and cubic.

3.6 Performance

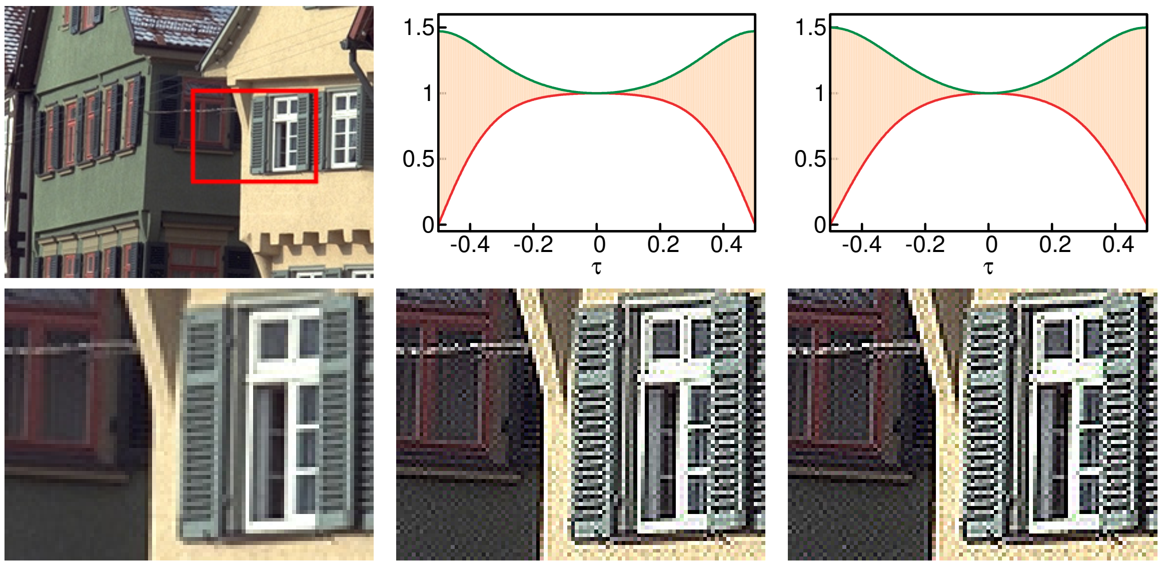
We need to obtain the reconstruction

$$\tilde{f} = \boldsymbol{c} *_T \varphi, \tag{3.25}$$

where

$$\boldsymbol{c} = [f]_T * \boldsymbol{q} \tag{3.26}$$

and $\boldsymbol{q}$ is an IFIR digital filter. To produce an output image, we need to sample this reconstruction.



(a) Input (b) Our linear (c) *Condat et al.* (2005)

Figure 3.13: Result of 4 repeated translations by exactly half pixel. Our linear quasi-interpolator (b) and the one proposed by *Condat et al.* (2005) magnify frequencies too much. Shaded regions in the plots show the range of frequency amplitude scaling for each translation τ. The worst case is $\tau = 0.5$.

We calculate the mixed convolution in (3.25) as described by *Nehab and Hoppe* (2014) (section 8.2), since this is the most efficient way we know.

The discrete convolution in (3.26) amounts to solving a linear system

$$\mathbf{A}\boldsymbol{c} = [\![f]\!]_T. \tag{3.27}$$

Since all our digital filters are symmetric and have length five they can be written as

$$\boldsymbol{q} = \big[\ldots, 0, r, q, p, q, r, 0, \ldots\big]^{-1}. \tag{3.28}$$

Assuming mirrored infinite extension for the images involved, the matrix $\mathbf{A}$ has the following form:

$$\mathbf{A} = \begin{pmatrix} p+q & q+r & r & 0 & & & & & & \\ q+r & p & q & r & 0 & & & & & \\ r & q & p & q & r & 0 & & & & \\ 0 & r & q & p & q & r & 0 & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & & \\ & & & 0 & r & q & p & q & r & 0 \\ & & & & 0 & r & q & p & q & r \\ & & & & & 0 & r & q & p & q+r \\ & & & & & & 0 & r & q+r & p+q \end{pmatrix}. \tag{3.29}$$

We factor $\mathbf{A}$ into $\mathbf{A} = \mathbf{L}\mathbf{U}$, where $\mathbf{U}$ is tridiagonal lower triangular and $\mathbf{U}$ is tridiagonal upper triangular and then solve the system (3.27) by backward and forward substitutions. We noticed that the entries of the diagonals of $\mathbf{L}$ converge very quickly and the entries of $\mathbf{U}$ are determined in terms of the entries of $\mathbf{L}$. Thus, we have to store and prescribe only the first entries of the diagonals of $\mathbf{L}$.

For the other methods shown in figures 3.14 and 3.15, the associated IFIR digital filters have only three entries. So we do what we have just described, but with $\mathbf{A}$ tridiagonal, $\mathbf{L}$ and $\mathbf{U}$ bidiagonal.

We report the timings of performing upsampling of a 1D image with 101 entries in figure 3.14 and downsampling of an image with 10001 entries in figure 3.15. We compare our cubic quasi-interpolator with other cubic ones, box and hat. We notice that our method has a slightly worse performance (around 10%) when compared to cubic Condat and cubic O-MOMS. This was expected since our IFIR digital filter has more entries and we think this additional cost pays off for the higher quality achieved by our method.

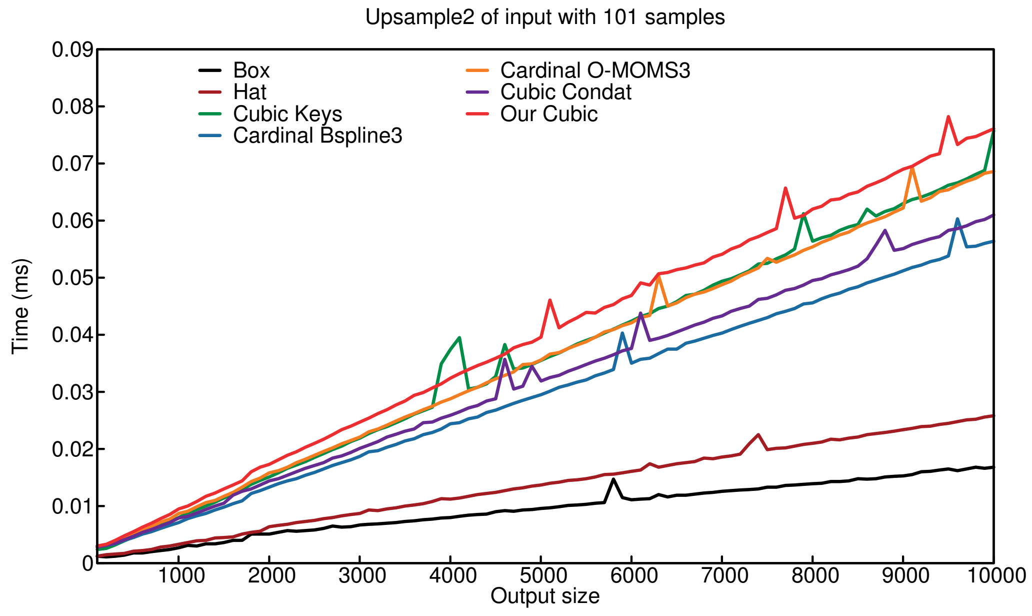


Figure 3.14: Upsampling a 1D input image with 101 samples with different methods. The performance of our cubic (solid red line) is around 10% worse.

3.7 Conclusion

We have presented a new class of quasi-interpolators for image processing that are optimal with respect to a non-asymptotic criterion. In contrast, previous strategies focused on making them optimal only around $\omega = 0$. Additionally, we used all available degrees of freedom in the approximation problem to reach higher quality.

An improvement to our optimization would be to consider a metric other than L_2. On one hand, it could lead to a more natural treatment for the overshoot problem, but it could also add additional difficulties to the optimization.

In this work we have considered a 1D formulation of the approximation problem, but applied it to images in a separable fashion. We believe that considering non-separable 2D quasi-interpolators will increase approximation quality, and we also consider this direction for future work.

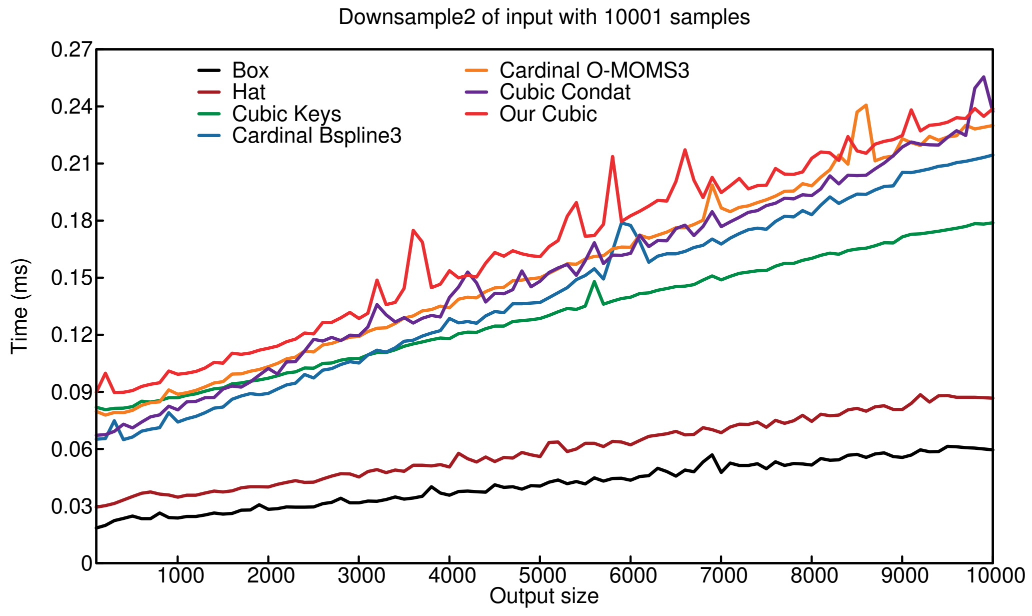


Figure 3.15: Downsampling a 1D input image with 10001 samples with different methods. Our cubic quasi-interpolator presents a slightly worse performance.

3.8 Quasi-Interpolators

Linear $d = 0.5$.

$$b_{0,0} = 0.79076352 \qquad c_{0,0} = c_{1,0} = 0.10461824,$$

$$e_0 = 0.77412669, \qquad e_1 = e_{\text{-}1} = 0.11566267,$$

$$e_2 = e_{\text{-}2} = -0.00272602$$

Quadratic $d \approx 0.34.$

$$
\begin{aligned}
a_{0,1} &= 0, & b_{0,0} &= 0.75627421,\\
b_{0,1} = b_{1,1} &= 0.11798097, & c_{0,0} = c_{2,0} &= 0.01588197,\\
c_{1,0} &= -0.02400002, & e_0 &= 0.65314970,\\
e_1 = e_{\text{-}1} &= 0.17889730, & e_2 = e_{\text{-}2} &= -0.00547216
\end{aligned}
$$

Cubic $d \approx 0.35.$

$$
\begin{aligned}
a_{0,1} &= 0.07922533, & a_{0,2} &= 0,\\
a_{1,1} = -2.25\, a_{0,1} &= -0.17825701, & b_{0,0} &= 0.53954836,\\
b_{0,1} &= 0.32092636, & b_{0,2} = b_{2,2} &= 0.02593862,\\
b_{1,1} = -1.5\, a_{0,1} + b_{0,1} &= 0.20208835, & b_{1,2} &= -0.01871558,\\
c_{0,0} = c_{3,0} &= 0.001940114, & c_{1,0} = c_{2,0} &= -0.00028665,\\
e_0 &= 0.56528428, & e_1 = e_{\text{-}1} &= 0.21523558,\\
e_2 = e_{\text{-}2} &= 0.00212228.
\end{aligned}
$$

3.9 Proofs

We can take $T = 1$ in (3.5) because, for any fixed $T > 0$, and assuming input spectra as (3.4):

$$\begin{aligned} \|f - \tilde{f}_T\|^2_{L_2} &\approx \int_{-0.5/T}^{0.5/T} \left|\hat{f}(\omega)\right|^2 E(T\omega)\,\mathrm{d}\omega \\ &= T \int_{-0.5}^{0.5} \left|f(\omega/T)\right|^2 E(\omega)\,\mathrm{d}\omega \\ &= T^{p+1} \int_{-0.5}^{0.5} 1/\omega^p\, E(\omega)\,\mathrm{d}\omega \\ &= T^{p+1} \int_{-0.5}^{0.5} \left|\hat{f}(\omega)\right|^2 E(\omega)\,\mathrm{d}\omega. \end{aligned} \tag{3.30}$$

To see why $E(0.5-\varepsilon) \approx 0$ implies $\hat{\varphi}_{\mathrm{qi}}(0.5-\varepsilon) \approx 1$, $\hat{\varphi}_{\mathrm{qi}}(0.5+\varepsilon) \approx 0$, $\hat{\varphi}_{\mathrm{qi}}(\text{-}0.5+\varepsilon) \approx 1$, $\hat{\varphi}_{\mathrm{qi}}(\text{-}0.5-\varepsilon) \approx 0$, recall $\hat{\psi} = 1$, and both $\hat{\boldsymbol{q}}$ and $\hat{\varphi}$ are real due to symmetry. The error kernel simplifies to:

$$\begin{aligned} E(\omega) &= 1 - \frac{\varphi(\omega)^2}{\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega)} + \widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega)\left(\hat{\boldsymbol{q}}(\omega)^2 - 2\frac{\hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega)}{\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega)} + \frac{\hat{\varphi}(\omega)^2}{\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega)^2}\right) \\ &= 1 - 2\hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega) + \hat{\boldsymbol{q}}(\omega)^2\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) \\ &= \left(1 - \hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega)\right)^2 - \hat{\boldsymbol{q}}(\omega)^2\hat{\varphi}(\omega)^2 + \hat{\boldsymbol{q}}(\omega)^2 \sum_n \hat{\varphi}(\omega+n)^2 \\ &= \left(1 - \hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega)\right)^2 + \sum_{n\neq 0} \hat{\boldsymbol{q}}(\omega+n)^2\hat{\varphi}(\omega+n)^2 \\ &= \left(1 - \hat{\varphi}_{\mathrm{qi}}(\omega)\right)^2 + \sum_{n\neq 0} \hat{\varphi}_{\mathrm{qi}}(\omega+n)^2. \end{aligned} \tag{3.31}$$

Above, we used the following equalities

$$\hat{\varphi}_{\mathrm{qi}}(\omega) = \hat{\boldsymbol{q}}(\omega)\hat{\varphi}(\omega) \tag{3.32}$$

$$\hat{\boldsymbol{q}}(\omega) = \hat{\boldsymbol{q}}(\omega + n), \forall n \in \mathbb{N}, \quad \text{and} \tag{3.33}$$

$$\widehat{\boldsymbol{a}_{\boldsymbol{\varphi}}}(\omega) = \sum_n \hat{\varphi}(\omega + n)^2. \tag{3.34}$$

The sum of non-negative terms in (3.31) shows us that

$$E(0.5-\varepsilon)\approx 0 \Rightarrow \hat{\varphi}_{\mathrm{qi}}(0.5-\varepsilon)\approx 1, \hat{\varphi}_{\mathrm{qi}}(\text{-}0.5-\varepsilon)\approx 0. \tag{3.35}$$

The symmetry of E implies $E(\text{-}0.5+\varepsilon)\approx 0$. From (3.31), we have

$$E(\text{-}0.5+\varepsilon)\approx 0 \Rightarrow \hat{\varphi}_{\mathrm{qi}}(\text{-}0.5+\varepsilon)\approx 1, \hat{\varphi}_{\mathrm{qi}}(0.5+\varepsilon)\approx 0. \tag{3.36}$$

CHAPTER IV

Foundations for the Tetrahedralization Algorithm

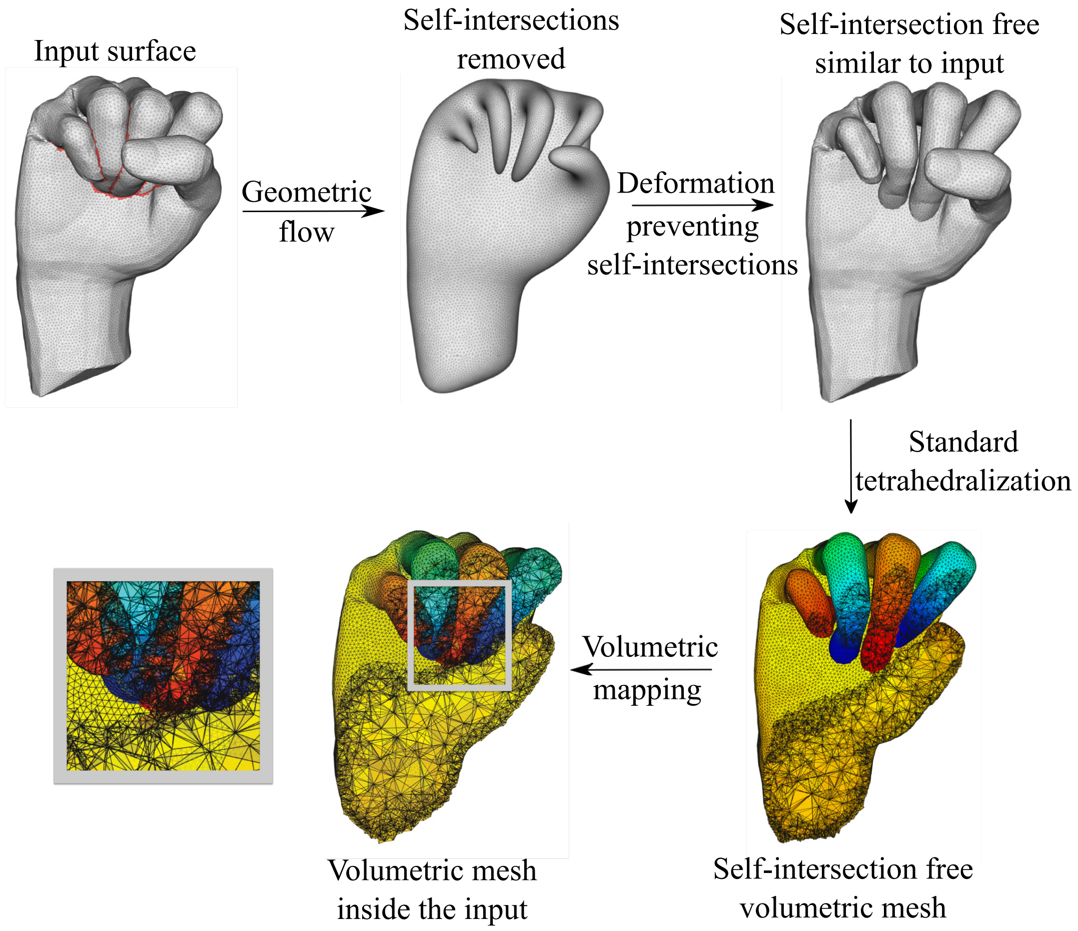

Input surface
Geometric flow
Self-intersections removed
Deformation preventing self-intersections
Self-intersection free similar to input
Standard tetrahedralization
Self-intersection free volumetric mesh
Volumetric mapping
Volumetric mesh inside the input

Figure 4.1: Overview of our new tetrahedralization algorithm. The input for it is a self-intersecting triangle mesh that cannot be tetrahedralized with previous algorithms. The first step is to apply a geometric flow until all self-intersections are removed. We then reverse the flow to get a surface similar to the input one, but free of self-intersections. This resulting surface can then be tetrahedralized with existing methods. To obtain an output tetrahedral mesh, we define a volumetric mapping with boundary conditions being the input surface.

4.1 Introduction

This chapter is devoted to explain the main concepts to understand our new tetrahedralization algorithm, to be detailed in the next chapter.

We introduce here a summarized description and motivation for it, so the reader can understand where each of the algorithms described in this chapter fit in our pipeline, and why is the problem itself relevant.

The input for our method is a self-intersecting closed orientable triangle mesh. These surfaces arise quite often as output of (surface-based) geometry processing algorithms, such as deformation, smoothing, subdivision and decimation (many examples are provided in the next chapter). These algorithms may create self-intersections, even if their input is a clean (closed, orientable) mesh. A self-intersecting model can also be a design choice, when an artist wants to model a self-intersecting solid that would naturally self-intersect and does not care about the volume that is inside it (for example, see *input* in figure 4.1).

Current tetrahedralization methods and available software do not handle self-intersecting input, because all of them assume *watertight* input, i.e., surfaces embedded in $\mathbb{R}^3$ whose underlying space is the same as the boundary of the closure of a 3-manifold in $\mathbb{R}^3$. We refer the reader to (*Shewchuk*, 2012) for a complete survey on existing tetrahedralization methods.

To the best of our knowledge, our method is the first to consistently mesh the interior of self-intersecting triangle meshes, i.e., we output a tetrahedral mesh whose boundary is the input triangle mesh and that overlaps itself respecting the overlaps in the input.

A description for our method is provided in figure 4.1. We start by applying a geometric flow to the input surface to remove its self-intersections. This flow is described in section 4.2. We then restore the shape of the original mesh by minimizing a deformation energy, while preventing self-intersections. The deformation energy we use is described in section 4.3. Given this self-intersection free surface similar to the input, we tetrahedralize its interior using (*Si*, 2003). The last step consists of mapping this resulting tetrahedral mesh to the interior of the input surface. We do it in a locally injective way to prevent inverted elements. The method we use to obtain locally injective mappings is described in section 4.4.

4.2 Geometric Flows

We present in this section the geometric flows we considered for the task of removing self-intersections from the input mesh.

4.2.1 Mean-Curvature Flow

This flow has been long studied both under theoretical and applied perspectives. It has the effect of pushing a point on a surface towards the average position of its neighbors, smoothing out the geometry surface.

Let M be a two-dimensional manifold, let $\Phi_t : M \to \mathbb{R}^3$ be a smooth family of immersions, and let $g_t(\cdot,\cdot)$ be the metric induced by the immersion at time t. We say that Φ_t is a solution to the *mean-curvature flow* if

$$\frac{\partial \Phi_t}{\partial t} = \Delta_t \Phi_t \tag{4.1}$$

where Δ_t is th Laplace-Beltrami operator defined with respect to the metric g_t.

We can implement it using a semi-implicit finite-element discretization (please see details in (*Kazhdan et al.*, 2012)). This formulation relates the vector of positions at the next time step $\mathbf{x}(t+\delta)$ to the current positions $\mathbf{x}(t)$ by the linear system

$$(D^t - \delta L^t)\mathbf{x}(t+\delta) = D^t\mathbf{x}(t). \tag{4.2}$$

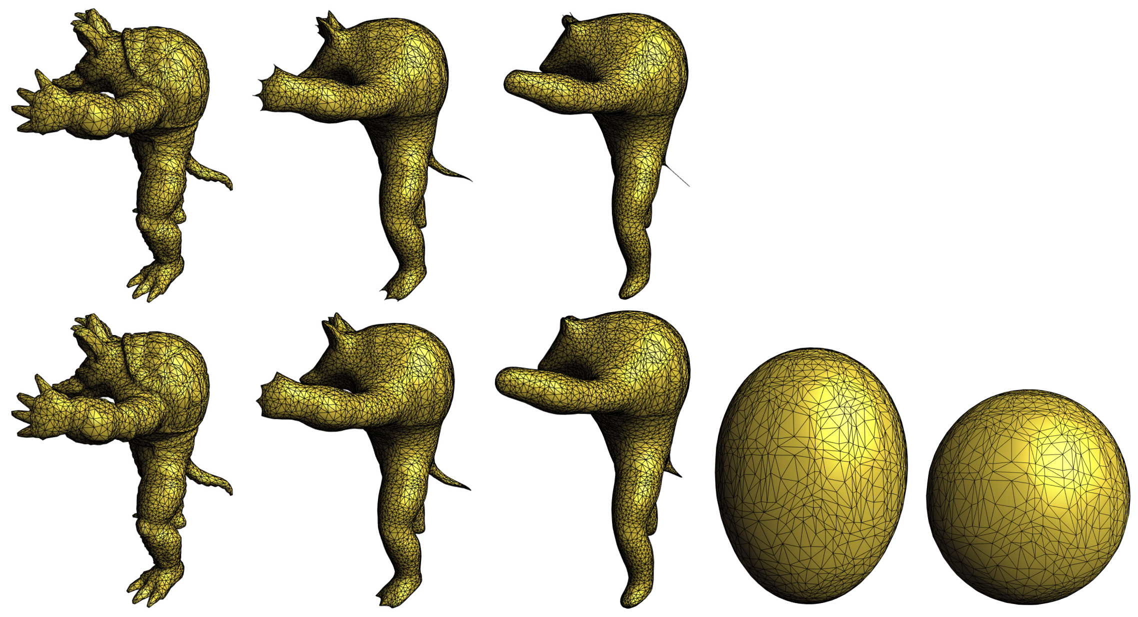


Figure 4.2: Top: Mean curvature flow (MCF) using time step $\delta = 10^{-4}$. We show the input surface, the result after 4 steps and 8 steps. This last surface presents neck pinching singularities on the tale and ears, causing singularities in the associated linear system and preventing the flow from progressing. Bottom: conformalized mean curvature flow (cMCF) using the same time step $\delta = 10^{-4}$. We show the input surface, the result after 4 steps, 8 steps, 300 steps and 1000 steps. This modification of the traditional mean curvature flow avoids instabilities and converges to the sphere for this example.

The entries of D^t and L^t depend on the triangle mesh at time t and are given by

$$D_{ij} = \begin{cases} \dfrac{area(T^1_{ij}) + area(T^2_{ij})}{12}, & \text{if} \quad j \in N(i) \\ \displaystyle\sum_{k \in N(i)} D_{ik}, & \text{if} \quad j = i \end{cases} \tag{4.3}$$

and

$$L_{ij} = \begin{cases} \dfrac{\cot(\beta^1_{ij}) + \cot(\beta^2_{ij})}{2}, & \text{if} \quad j \in N(i) \\ -\displaystyle\sum_{k \in N(i)} L_{ik}, & \text{if} \quad j = i \end{cases} \tag{4.4}$$

where $N(i)$ are the indices of the vertices adjacent to vertex i, T^1_{ij} and T^2_{ij} are the two triangles sharing edge (i,j), and β_{ij}^1 and β_{ij}^2 are the two angles opposite to edge (i,j). D is called *mass matrix* and L is called *stiffness matrix* or *cotangent matrix*.

We show in figure 4.2 (top) the result of applying this discretization of the flow to the *Armadillo* mesh. After each time step, we re-normalize the resulting surface to have unit area, otherwise it would shrink towards a point. After some iterations, the mesh develops neck pinching singularities on the tale and the ears. This prevents the flow from progressing, since the pinches correspond to zero area triangles and the matrix in (4.2) becomes singular. We discuss the causes of this problem in the next subsection.

4.2.2 Conformalized Mean-Cruvature Flow

Kazhdan et al. (2012) noticed that the (local) anisotropic scaling caused by the evolution of the flow makes the entries of L^t (4.4) to blow up, and this leads to singularities of the linear system (4.2). To overcome this problem, they suggest to solve a modified version of the linear system:

$$(D^t - \delta L^0)\mathbf{x}(t+\delta) = D^t\mathbf{x}(t). \tag{4.5}$$

The only change is that the cotangent matrix L is fixed to be the one calculated based on the initial (input) mesh.

The authors show that this modified linear system corresponds to the semi-implicit finite element discretization of the following flow:

$$\frac{\partial \Phi_t}{\partial t} = \sqrt{|g_t^{-1}g_0|}\Delta_{g_0}\Phi_t \tag{4.6}$$

This modified flow is called *conformalized mean-curvature flow* (cMCF). It has an important convergence result that will be useful for us:

Theorem IV.1. *If cMCF converges, than it converges to a map onto the sphere if*

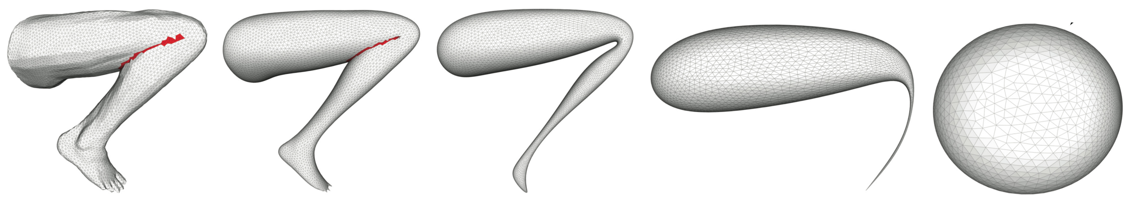


Figure 4.3: Conformalized mean curvature flow applied to remove self-intersections (shown in red). Intersections are removed much earlier before reaching the limit surface.

and only if the limit map is conformal.

Proof: The only proposition in (*Kazhdan et al.*, 2012).

■

The authors present a number of examples for genus zero surfaces showing that the limit map is conformal, but the theoretical question is still open. If this is always true, then cMCF always converges to the sphere for genus zero input.

The example in figure 4.2 (bottom) illustrates both practical and theoretical advantages of this new flow. In the practical side, we can see that the singularities that arise for the standard mean curvature flow are not developed, and the flow can take many time steps without having numerical problems. After 1000 steps it reaches the (sphere) limit surface.

The convergence to the sphere is of fundamental importance to our application. We have a theoretical guarantee that all self-intersections present in the input surface will be removed by the flow. In the worst case, this will happen at the limit state. But many of our experiments show that intersections are removed after a few steps of the flow, making it an efficient option for this task. Figure 4.3 shows an example of the conformalized mean-curvature flow removing intersections.

4.2.3 Conformal Willmore Flow

The mean-curvature flow in (4.1) has an alternative interpretation as being the gradient descent step for the minimization of the *membrane energy*

$$E_A(\Phi) = \int_M dA, \tag{4.7}$$

where dA denotes the area element. In other words, mean-curvature flow progresses towards area-minimizing surfaces. The reason why our results do not shrink is that we renormalize the resulting surfaces at each step.

Another well-known energy for surface fairing purposes is the *Willmore energy*. It is defined as

$$E_W(\Phi) = \int_M H^2 dA, \tag{4.8}$$

where H is the pointwise mean-curvature of the surface. Its gradient descent formulation is not as simple as for the membrane energy due to non-linearities involved (*Crane et al.*, 2013a).

Crane et al. (2013a) (please see details in their paper) noticed that (4.8) can be rewritten in terms of the *mean-curvature half density* μ as

$$E_W(\mu) = \int_M \mu^2 = \|\mu\|^2. \tag{4.9}$$

This change of variables turns the gradient descent formulation into

$$\dot{\mu} = -2\mu \tag{4.10}$$

They then apply forward Euler to (4.10) and recover positions in terms of half-curvature density as described in (*Crane et al.*, 2011). Moreover, the way they recover

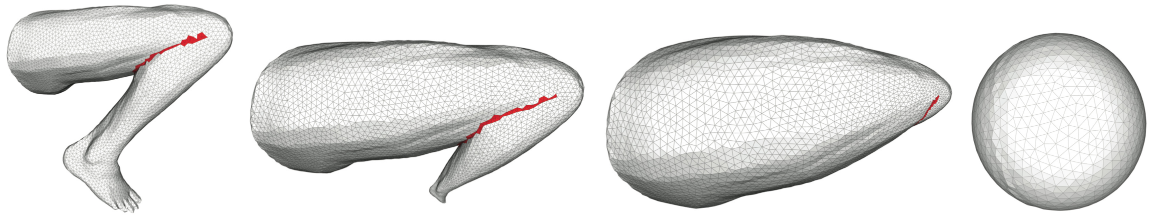


Figure 4.4: Conformal Willmore flow takes too many steps to remove self-intersection (shown in red). The intersections are only removed near the sphere for this example.

positions from curvatures results in a conformal transformation from the surface from the previous step.

Even using a forward discretization for (4.10) the authors illustrate the remarkable stability of their method with many examples. Unfortunately for our application this flow takes very long to remove self-intersections (see figure 4.4). Another issue is that it does not have a convergence proof analogous to Theorem IV.1.

4.3 Deformation Energies

We opted for using the conformalized mean-curvature flow to remove self-intersections from the input mesh due to its stability and theoretical guarantees. After applying the flow until all self-intersections are removed we could (at least in theory) tetrahedralize its interior with standard methods.

But in the practical setting, a problem appears (see figure 5.7 for an example): the flow shrinks elongated parts of the mesh towards its main body. These shrunk regions lead to triangle areas that are too small for the tetrahedralization method to handle.

Motivated by this problem we proposed a reverse flow to restore the triangle areas on the mesh, while preventing self-intersections (to be able to tetrahedralize it in the end). In the next section we present the energy we minimize and the constraints are presented in the next chapter.

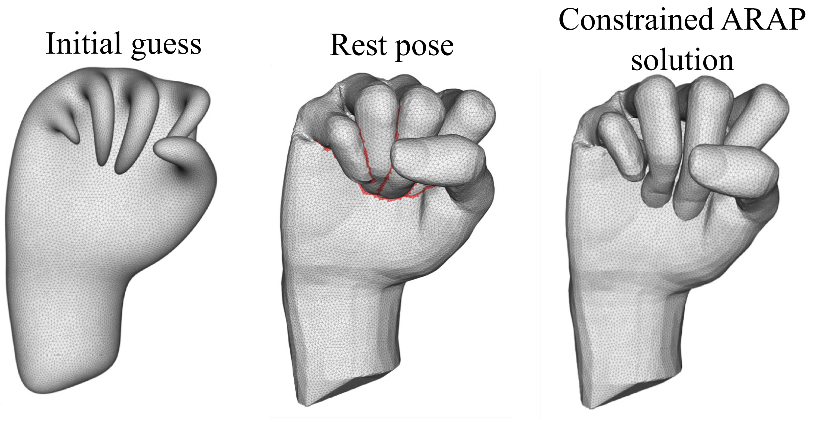


Figure 4.5: We restore the shape of the triangles minimising ARAP energy subject to non-penetration constraints. The surface resulting from the flow is treated as initial guess and the input mesh is treated as rest pose.

4.3.1 As-Rigid-As-Possible Deformation

We show in figure 4.5 the setting for the ARAP minimization. A reference mesh (center) contains all triangles with good shape (a.k.a. *rest pose*). In our case, this mesh corresponds to the input before the flow was applied. We denote all vertices of the rest pose by $\mathbf{p}$. We want to find a configuration of vertices $\mathbf{p}'$ that has triangles shapes similar to $\mathbf{p}$.

One way of doing this is to find $\mathbf{p}'$ that is a locally rigid transformation of $\mathbf{p}$, i.e., each vertex in $\mathbf{p}'$ is a composition of rotation and translation from the corresponding vertex in $\mathbf{p}$. Shearing and scaling are penalized. The *as-rigid-as-possible (ARAP) energy* provides a way of quantifying rigidity between $\mathbf{p}$ and $\mathbf{p}'$:

$$E_{ARAP}(\mathbf{p}, \mathbf{p}') = \sum_{j \in N(i)} w_{ij} \|(\mathbf{p}'_i - \mathbf{p}'_j) - \mathbf{R}_i(\mathbf{p}_i - \mathbf{p}_j)\|^2 \tag{4.11}$$

Above $\mathbf{R}_i$ are unknown rotations and $\mathbf{p}'_i$, $\mathbf{p}'_j$ are unknown positions, $N(i)$ denotes the neighboring vertices of vertex i and w_{ij} are weights (see (*Sorkine and Alexa*, 2007) for details).

Assuming $\mathbf{p}'_i$, $\mathbf{p}'_j$ known, we follow the development by *Sorkine and Alexa* (2007) to obtain optimal rotations $\mathbf{R}_i$. For each vertex i we define the covariance matrix

$$\mathbf{S}_i = \sum_{j \in N(i)} w_{ij} (\mathbf{p}_i - \mathbf{p}_j)(\mathbf{p}'_i - \mathbf{p}'_j)^T \tag{4.12}$$

Sorkine and Alexa (2007) prove that if the SVD factorization of $\mathbf{S}_i$ is given by

$$\mathbf{S}_i = \mathbf{U}_i \Sigma_i \mathbf{V}_i^T \tag{4.13}$$

then the optimal rotations are given by

$$\mathbf{R}_i = \mathbf{V}_i \mathbf{U}_i^T. \tag{4.14}$$

This optimal rotation calculation is called *local step* of the ARAP minimization.

If the rotations $\mathbf{R}_i$ are known, finding the optimal configuration of the vertices $\mathbf{p}'$ is just a quadratic minimization problem that is equivalent to solving the linear system

$$\mathbf{L}\mathbf{p}' = \mathbf{b} \tag{4.15}$$

where $\mathbf{L}$ is the cotangent matrix and $\mathbf{b}_i = \sum_{j \in N(i)} \frac{w_{ij}}{2} (\mathbf{R}_i + \mathbf{R}_j)(\mathbf{p}_i - \mathbf{p}_j)$. This calculation of optimal positions is called *global step*.

To minimize the non-linear energy (4.11), *Sorkine and Alexa* (2007) propose to alternate between local and global steps. Given an initial guess for positions $\mathbf{p}_0$ (the mesh on the left in figure 4.5) and a rest pose $\mathbf{p}$, optimal rotations are determined by (4.14). Once this optimal rotations are calculated, then optimal positions are determined by (4.15). This alternation between local and global steps is performed until convergence is reached.

Although formulated in a discretized setting, it is possible to show that (4.11) has a close relation to a continuous energy presented by *Chao et al.* (2010) in section 4.2.

4.3.2 Other energies

We can restate the ARAP energy in an alternative way so it can be generalized to other *elements* such as tetrahedra (in the previous section we presented the ARAP energy for triangular meshes only).

Let d be the dimension of the problem (2 or 3) and let $\mathbf{F}_j \in \mathbb{R}^{d\times d}$ be the *deformation gradient*, which is defined as the mapping that takes the rest-pose vertices of an element j to their deformed positions. A (similar) version of the ARAP energy can be written as

$$E_{ARAP}(\mathbf{p}') = \sum_{j\in\mathcal{E}} \lambda_j(\mathbf{p}) \|\mathbf{F}_j - \mathbf{R}_j\|_F^2 \tag{4.16}$$

where $\lambda_j(\mathbf{p})$ are areas or volumes of the rest pose elements, $\mathbf{R}_j$ are the optimal rotations, $\mathcal{E}$ is the set of all elements in the mesh and $\|A\|_F = \sqrt{\sum_i \sum_j |a_{ij}|^2}$ is the Frobenius norm.

With this same notation, other energies can be defined. A simpler way of quantifying deformations is given by the *Dirichlet Energy*:

$$E_{DIR}(\mathbf{p}') = \sum_{j\in\mathcal{E}} \lambda_j(\mathbf{p}) \|\mathbf{F}_j - \mathbf{I}\|_F^2. \tag{4.17}$$

The Dirichlet energy penalizes rotations. An alternative energy that is rotation-invariant is the *Green's strain energy*, given by:

$$E_{GS}(\mathbf{p}') = \sum_{j\in\mathcal{E}} \lambda_j(\mathbf{p}) \|\mathbf{F}_j^T \mathbf{F}_j - \mathbf{I}\|_F^2 \tag{4.18}$$

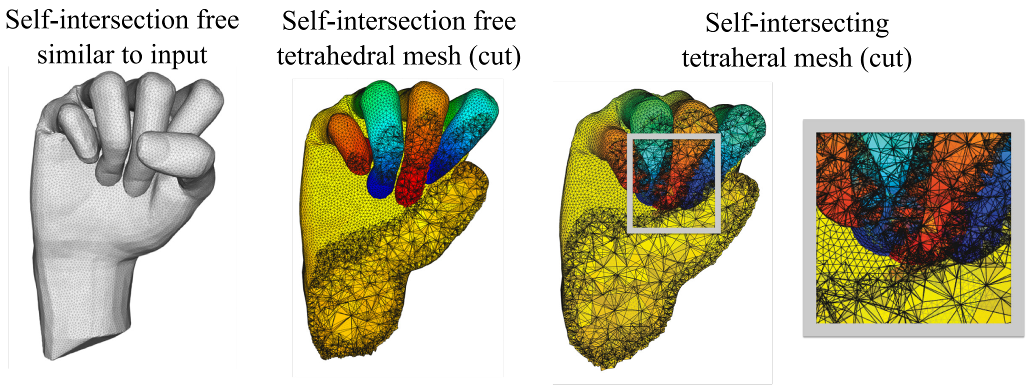

Self-intersection free similar to input
Self-intersection free tetrahedral mesh (cut)
Self-intersecting tetraheral mesh (cut)

Figure 4.6: After minimizing the ARAP energy subject to non-penetration constraints we obtain a surface mesh similar to the input but without self-intersections. This mesh can be tetrahedralized with standard algorithm (left-center). The last step consists of mapping this volumetric mesh to inside the initial surface, leading to the self-overlapping volumetric mesh on the right. To prevent inverted elements, we use *locally injective mappings* (*Schüller et al.*, 2013) for this last step.

4.4 Locally Injective Mappings

After restoring the shapes of the triangles in the surface mesh, we obtain a mesh similar to the input one but free of self-intersections. This mesh can be tetrahedralized with standard algorithms such as (*Si*, 2003), leading to the volumetric mesh in figure 4.6 (left-center). We want to map this tetrahedral mesh to the interior of the input mesh ("Input surface" in figure 4.1).

Let E be one of the energies presented in the previous section, $\mathbf{d}$ be the boundary conditions (positions of the input mesh) and $\mathbf{v}$ the (unknown) vertices of the final tetrahedral mesh. One way to minimize E imposing the boundary conditions is by solving the unconstrained problem

$$\arg\min_{\mathbf{v}} E(\mathbf{v}) + \alpha \|\mathbf{C}\mathbf{v} - \mathbf{d}\|^2, \tag{4.19}$$

where $\alpha \geqslant 0$ is a parameter specifying the weight of the boundary conditions and $\mathbf{C}$ is a matrix that selects the boundary vertices from $\mathbf{v}$.

If we define the solution of (4.19) as our final mesh it would contain inverted elements. This unconstrained energy does not punish element inversion. Inverted elements are undesired in many applications and in some of them they are not allowed.

To solve this problem we apply the method proposed by *Schüller et al.* (2013): let $\lambda_j(\mathbf{v})$ be the volume of tetrahedron j. We define a constraint function as

$$c_j(\mathbf{v}) = \lambda_j(\mathbf{v}) - \varepsilon, \tag{4.20}$$

where $\varepsilon > 0$ is a small constant that takes into account numerical inaccuracies.

A barrier function $\phi : \mathbb{R} \to \mathbb{R}$ is a function such that

$$\lim_{x \to 0} \phi(x) = \infty, \text{ and } \phi(x) = \infty, \text{ for } x \leqslant 0. \tag{4.21}$$

To enforce non-inverted elements (as hard constraints) the authors solve the following unconstrained problem:

$$\arg\min_{\mathbf{v}} E(\mathbf{v}) + \alpha \|\mathbf{C}\mathbf{v} - \mathbf{d}\|^2 + \beta \sum_{j \in \mathcal{E}} \phi_j(c_j(\mathbf{v})), \tag{4.22}$$

where $\beta > 0$ is a scalar specifying the barriers strength, $\mathcal{E}$ is the set of all tetrahedra in the mesh and ϕ_j are barriers defined for each element.

Numerical issues aside, solving (4.22) guarantees a tetrahedral mesh with no inverted elements. To solve (4.22) efficiently and to consider the complicated numerics in the problem, *Schüller et al.* (2013) present a new solver specific to this problem. We refer to their work for details.

Their work also contains many results illustrating that their formulation is effective. Specifically, they show a tetrahedral mesh with no inverted elements in their figure 13. Another example is the tet-mesh we present in figure 4.6. Many more examples are provided in the next chapter.

CHAPTER V

Consistent Volumetric Discretizations Inside Self-Intersecting Surfaces

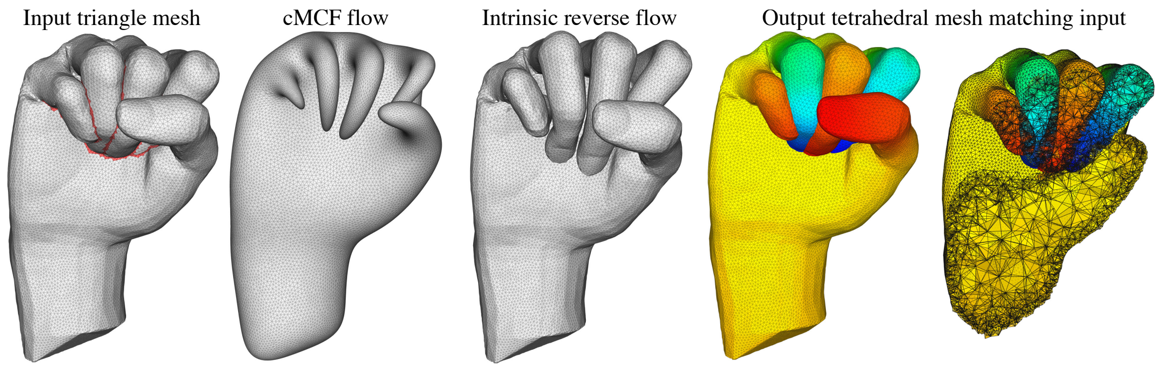


Figure 5.1: The triangle mesh of the *Hand* forms a closed surface, but contains nearly 2000 intersecting triangle pairs. Our method flows the surface according to conformalized mean-curvature flow (cMCF) until all self-intersections are removed. Then we *reverse* the flow so that shape intrinsics are restored but self-intersections are avoided. Finally we can tet-mesh inside this surface and map the mesh so that it matches the original surface. We may then solve PDEs, such as this biharmonic function.

5.1 Introduction

Recent years have shown leaping advancements in *surface-based* shape processing, in particular polygonal mesh processing (*Botsch et al.*, 2010). Volumetric shape processing, on the other hand, lags behind. One significant obstacle is the inability to

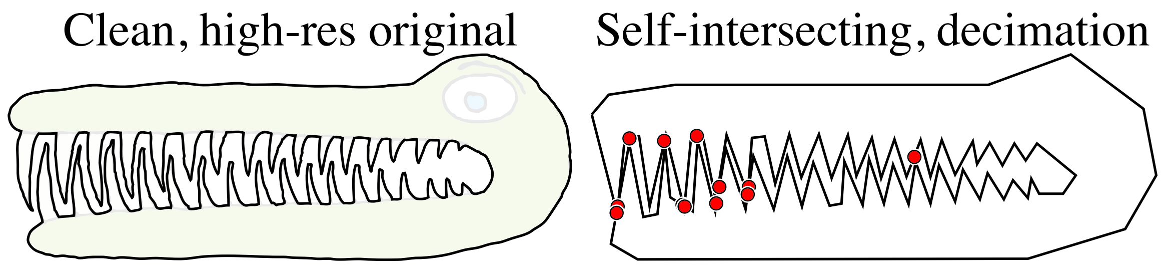


Figure 5.2: Even simple geometric operations like boundary decimation can introduce self-intersections (red dots).

convert boundary representations of solid shapes into explicit volumetric representations at any given stage of the geometry processing pipeline. Robust tools for creating tetrahedral meshes from watertight input surfaces do exist, e.g. *Si* (2003). However, the presence of self-intersections in the surface mesh invalidates the otherwise clean (closed, orientable) input to volume meshing algorithms. Unfortunately, most—if not nearly all—steps in the *surface-based* geometry processing pipeline (such as decimation, smoothing, subdivision, remeshing, surface-based deformations) may invalidate watertightness by creating self-intersections (*Harmon et al.*, 2011) (see Figure 5.2). As a consequence, attempts at further volumetric processing reveal artifacts resulting from ignoring or deleting self-intersecting regions (see figure 5.10), and therefore geometry processing remains limited to the surface.

Volumetric processing has a lot of valuable advantages. While surface meshes are appropriate representations for some shapes, such as thin shells (e.g. an automobile fender), many interesting shapes are solids. For a solid shape, like a deformable human character, we may sometimes get away with a surface-only representation thanks to its intrinsic, though indirect, relationship to the underlying volume and because we often will only render the surface. However, many processing tasks perform drastically differently when treating a solid shape as a surface rather than a volume, e.g. the bending of scanned clay statuette (see Figure 10 in (*Botsch et al.*, 2007)), rendering shapes made of translucent material, like an amber jewel (*Li et al.*, 2012), registering two poses of a dancing human (*Litman et al.*, 2012), or even simple shape smoothing (see figure 5.3). Notably, volumetric representations facilitate volume preservation and internal geodesic distance computation. Ubiquitous techniques like finite element analysis and solving PDEs typically require an *explicit* representation of a shape's volume: most commonly, a tetrahedral mesh.

We propose a method to construct a tetrahedral mesh for self-intersecting input. Instead of gluing overlapping regions together, our output volume mesh overlaps itself consistently with the self-intersections in the input surface (see figure 5.1). This enables correct geodesic information necessary for shape-aware volumetric processing at any stage in the geometry processing pipeline.

We begin with a key observation: For sphere-topology surfaces, conformalized mean-curvature flow (cMCF) converges to the unit sphere (*Kazhdan et al.*, 2012) and removes all intersections. Given an input surface, we follow this flow until all self-intersections are removed; typically long before reaching the sphere (see figure 5.5). Meshing techniques like constrained Delaunay tessellation (CDT) should in theory work on this resulting surface. We could try to mesh its interior and then try to map this mesh back inside the original surface as an exercise in volumetric parameterization. However, exponential scaling of the triangles during cMCF introduces numerical issues for existing CDT software. This large distortion also makes the subsequent volumetric parameterization difficult or impossible, even with state-of-the-art methods (*Schüller et al.*, 2013).

We solve this problem in two steps. We first reverse the flow in an intrinsic way, while maintaining absence of self-intersections as an invariant. For each step backward in the flow, we minimize a surface-distortion energy subject to safe contact constraints. Because cMCF is smooth (even conformal in the limit), this may be interpreted as an intersection-free, surface simulation *regularized* by the flow. When we have returned to time zero, our surface is similar to the original input surface

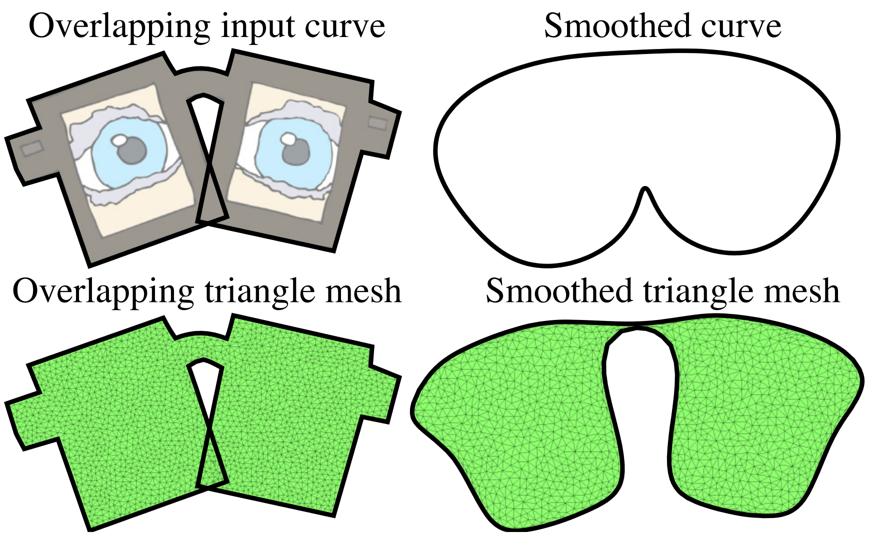


Figure 5.3: Geometric operations perform drastically differently on boundary versus region representations. Compare the results of Laplacian smoothing for these eyeglasses as a curve and as a region.

intrinsically, but self-intersection free (see figure 5.6). Now, we may safely apply existing CDT methods. As a final step, we map the surface of this volume to the original, self-intersecting input surface and propagate the map to the interior. We show the success of our method for applications including solving PDEs, volumetric elastic deformation and simulation, automatic skinning weight definition and geodesic distance computation.

5.2 Related Work

Volumetric discretizations of *watertight* shapes have greatly improved in the past years (*Shewchuk*, 2012). State-of-the-art methods provide guarantees on element quality and have rich feature sets like the ability to specify spatially-varying density fields or exactly conforming to a given piecewise-linear surface mesh (CGAL; *Si*, 2003; *Labelle and Shewchuk*, 2007; *Geuzaine and Remacle*, 2009). However, they all assume that the input is a representation (implicit function, triangle mesh) of a watertight surface[1]. We bootstrap these methods in order to discretize volumes of *self-intersecting* solids, which of course have self-intersecting surfaces. We heavily employ the CDT and mesh refinement routines of the award-winning TetGen software (*Si*, 2003). By leveraging locally injective volumetric parameterization, we prevent inverted elements in our output, so we may further post-process our output by other mesh refinement techniques to achieve gradations or even higher quality (*Shewchuk*, 2012).

Some surface repair techniques eliminate self-intersections, outputting a watertight mesh which may then be meshed. But these repairs either delete (*Shen et al.*, 2004; *Attene*, 2010) or fuse intersecting pieces (*Jacobson et al.*, 2013). Our method accommodates self-intersections without any modification to the original surface—local or otherwise (see Figure 5.10).

Naive methods are generally not an option. Having meshed the entire convex hull, one could segment based on the winding number (analogous to the "nonzero-rule" of SVG and OpenGL, (*Foley et al.*, 1990; *Jacobson et al.*, 2013)). This not only leads to incorrectly deleting or joining entire regions, but also easily results in non-manifold output. It is also tempting to consider decomposing the input into intersection-free pieces, meshing each independently and reconnecting them. Constructing, let alone combinatorially reconnecting, such a decomposition is not obvious for complicated or multiple overlaps. Further, one must ensure discretization coherency across cuts. *Luo et al.* (2012) consider planar cuts to decompose shapes for 3D printing, but even assuming self-intersection free input they show that managing these requires care. Our method avoids combinatorial decisions.

Instead of decomposing or modifying the input surface, we find a new embedding for it via conformalized mean-curvature flow (cMCF), which removes self-intersections (*Kazhdan et al.*, 2012). Several common flows converge to the sphere, e.g. the Will-

[1]Formally, a watertight surface is "a 2-[manifold] embedded in $\mathbb{R}^3$ whose underyling space is same [sic] as the boundary of the closure of a 3-manifold in $\mathbb{R}^3$" (*Dey and Goswami*, 2003).

more flow, volume-preserving mean-curvature flow, heat diffusion flow, etc. (*Willmore*, 2000). Unlike mean-curvature flow, many of these flows are known to create new self-intersections even in their absence in the input (*Mayer and Simonett*, 2000, 2003), making them unsuitable for our purposes. We also enjoy the simplicity, performance and robustness of cMCF for triangle mesh discretizations.

We optimize a surface-based, elastic energy with dynamics to reverse the flow while preventing self-intersections with safe contact constraints and repulsion forces at the vertex level. *Harmon et al.* (2011) similarly prevent self-intersections during interactive deformation, speeding up computation by grouping collision responses. Alternatively, we could employ this method, but grouping runs the risk of locking early on during the reverse flow. Further, our simulation conveniently does not require expensive and complicated gradient computation for the deformation energy. There is a large amount of literature in physically-based simulation on robust and efficient handling of collisions (*Harmon*, 2010). Given our intersection-free state we could treat the original surface as a rest-state and run any off-the-shelf surface simulation with safe collision handling (e.g. (*Bridson et al.*, 2002)). However, this quickly results in locking (see Figure 5.7). Our use of the forward flow surfaces at intermediary time steps avoids this. Another option would be to avoid the flow altogether and attempt to untangle the self-intersections (*Baraff et al.*, 2003). Tailored to open cloth surfaces, this heuristic appears to succeed for small overlaps, but purposefully assumes no knowledge of a possible intersection-free state (e.g. a previous frame in their simulation). It relies instead on heuristic global topological analysis of the input.

A few works have defined similar problems in $\mathbb{R}^2$: discretizing or charting the area inside an overlapping curve in the plane. Unfortunately their solutions are slow and do not extend to $\mathbb{R}^3$ (*Shor and Van Wyk*, 1989; *Eppstein and Mumford*, 2009; *Mukherjee et al.*, 2011). It is very difficult to have a definition of valid input that is not based on the existence of valid output, or in other words, defining what valid input is such that

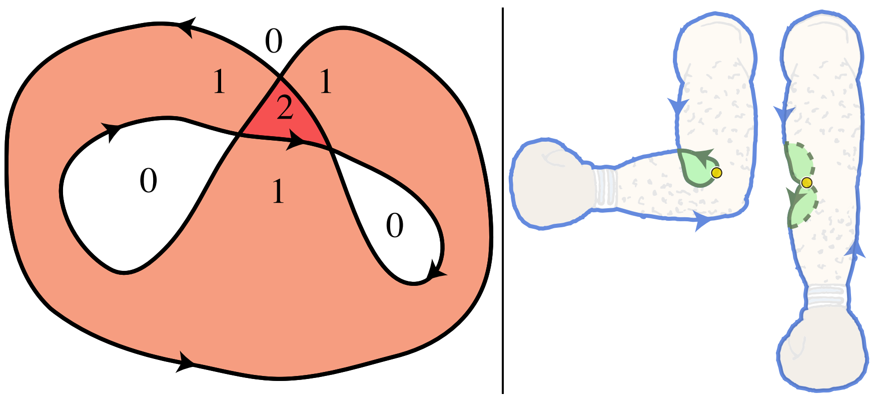


Figure 5.4: Left (with winding numbers indicated): contrary to *Mukherjee et al.* (2011), this curve is not a "self-crossing loop". Middle: This *elbow case* is also not a self-crossing loop, and so it is not considered by any existing 2D method. Our method succeeds by allowing the mapping to not be locally injective on the boundary (at the yellow dot), visualized in a hypothetical untangled state (right).

validity is easy to verify. The observations in Section 2.4 of (*Mukherjee et al.*, 2011) make use of the *winding number*, the signed number of times a curve wraps around a point. Unfortunately, these observations are necessary but not sufficient, since they would accept the infeasible curve in Figure 5.4 (left). This curve is not even a valid "self-crossing loop", i.e. the boundary curve of a locally-injectively deformed circular disc (*Shor and Van Wyk*, 1989). Interestingly, the existence-of-output definition of self-crossing loops used in (*Shor and Van Wyk*, 1989) and all following works does not include the *elbow case* (Figure 3b of (*Shor and Van Wyk*, 1989)), which arises frequently during surface deformations (see Figure 5.4, right). Our formulation is more general and handles this case by allowing a zero-measure subset of the boundary where the computed map is not locally injective. A similar situation in 3D is shown in Figure 5.9.

5.3 Problem description

Let our input be $\mathcal{M}$, a closed, orientable $(d-1)$-manifold embedded in $\mathbb{R}^d$, with possible self-intersections. From now on we assume $d = 3$, but our problem and

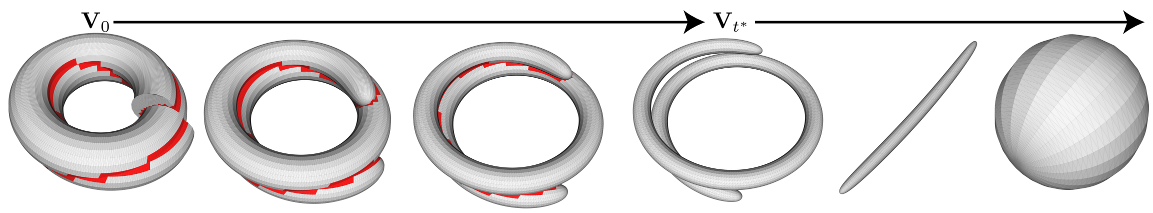


Figure 5.5: We evolve a self-intersecting surface (intersections are shown in red) with the conformalized Mean Curvature Flow. After some time t^* the surface reaches a self-intersection free state. The limit is a conformal mapping to the unit sphere.

solution generalize for $d \geq 2$. Our goal is to chart the volume *inside* $\mathcal{M}$. That is, we wish to find a continuous map $\Omega : \mathcal{D} \rightarrow \mathbb{R}^3$ where $\mathcal{D}$ is an abstract 3-manifold with boundary and our mapping Ω meets the following requirements:

$$\Omega(\partial\mathcal{D}) = \mathcal{M}, \tag{5.1}$$

$$\Omega(\mathcal{D} \setminus \partial\mathcal{D}) \quad \text{is differentiable}, \tag{5.2}$$

$$\left|J_\Omega(\mathbf{p})\right| > 0 \quad \forall \mathbf{p} \in \mathcal{D} \setminus \partial\mathcal{D}, \tag{5.3}$$

where $\left|J_\Omega(\mathbf{p})\right|$ is the determinant of the Jacobian matrix of the map Ω evaluated at point $\mathbf{p}$. The first requirement states that Ω should map the boundary of $\mathcal{D}$ to the input surface $\mathcal{M}$. The second and third requirements ensure local injectivity on the interior. We do not require local injectivity on the boundary. This allows isolated "hinge points" on $\partial\mathcal{D}$ to appear, necessary for handling *elbow cases* (see Figure 5.4). With the local injectivity requirement on the boundary our definition would be equivalent to "self-crossing loops" (*Shor and Van Wyk*, 1989). In practice, these requirements imply that our output tetrahedral mesh conforms to the input boundary, has no flipped (negative signed volume) tetrahedra, and has proper connectivity.

Forward flow. As in (*Kazhdan et al.*, 2012), we define the conformalized mean-curvature flow (cMCF) $\Phi_t : \mathcal{M} \rightarrow \mathbb{R}^3$ to be a smooth family of immersions, each the

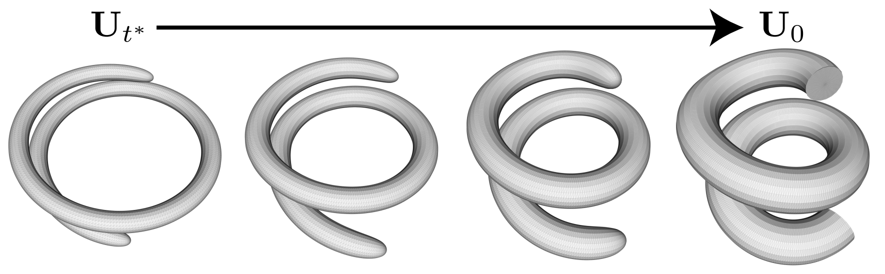


Figure 5.6: We evolve the intersection-free surface obtained with cMCF to a one that is similar to the input surface intrinsically, but without self-intersections.

solution to the partial differential equation:

$$\frac{\partial \Phi_t}{\partial t} = \sqrt{\left|g_t^{-1} g_0\right|} \Delta_{g_0} \Phi_t, \tag{5.4}$$

where $g(\cdot,\cdot)_t$ is the metric induced by the immersion at time t, and Δ_{g_0} is the Laplace-Beltrami operator defined with respect to the original metric g_0 of $\mathcal{M}$.

Kazhdan et al. (2012) prove that "if cMCF converges, than [sic] it converges to a map on the sphere if and only if the limit map is conformal". This is of special importance to us because the mapping to the sphere will have removed all self-intersections. They observed this convergence in all their tests with sphere-topology examples, and we confirm this empirically, too. In general, self-intersections disappear long before reaching the sphere. Let t^* and Φ_{t^*} be the time at which this occurs and the corresponding immersion, respectively (see Figure 5.5).

Reverse flow. In the discrete setting, the exponential scaling in Φ_t introduces numerical issues that prevent us from directly mapping the volumetric closure of $\Phi_{t^*}(\mathcal{M})$ back to $\mathcal{M}$. We alleviate this by finding an *intrinsic* reverse flow. Let $\Psi_t : \mathcal{M} \to \mathbb{R}^3$ be a family of immersions defined as the optimum of the nonlinear, constrained optimization problem:

$$\begin{aligned} \arg\min_{\Psi_t} \quad & E_{\text{surf}}(g_t, \tilde{g}_t), && (5.5) \\ \text{subject to:} \quad & \Psi_t \text{ is injective} && (5.6) \end{aligned}$$

where $\tilde{g}(\cdot,\cdot)_t$ is the metric induced by Ψ_t and $E_{\text{surf}}(g_t, \tilde{g}_t)$ measures the *similarity* of the metrics g_t and $\tilde{g}_t$. In this way, our optimization finds a non-self-intersecting immersion Ψ_t which is as close as possible to Φ_t (see Figure 5.6). Since Ψ_t is defined w.r.t. discontinuous constraints, we cannot write that it is a *smooth* family of immersions w.r.t. t, but this is not an issue for us as we are only concerned with the quality of Ψ_0.

We can immediately notice that if cMCF converges, the feasible set of solutions is non-empty: the sphere and thus also Ψ_{t^*} are intersection-free. In the ideal world, E_{surf} would measure the similarity of the *volumes* within Φ_t and Ψ_t, but of course without the unknown discretization of the volume within Φ_t this is elusive in practice. Hence, we choose E_{surf} to be a cumulative measure of local surface rigidity, namely the surface-based elastic energy discussed in (*Chao et al.*, 2010).

For our purposes, we are only concerned with the final reversed flow Ψ_0, but in practice we compute Ψ_t at the same samples in time as Φ_t. These intermediary solutions will be essential as feasible, initial guesses to the subsequent steps back in time $(t - \delta)$ until reaching time 0.

Volumetric parameterization. Now we no longer need to treat $\mathcal{D}$ as an abstract domain: let $\mathcal{D}$ be the closure of $\Psi_0(\mathcal{M})$. The problem of finding a suitable volumetric mapping Ω reduces to a volumetric parameterization problem with a fixed boundary.

We can write this volumetric parameterization Ω as the optimum of:

$$\begin{aligned} \underset{\Omega}{\arg\min} \quad & E_{\text{vol}}(\Omega), && (5.7) \\ \text{subject to:} \quad & \left|J_\Omega(\mathbf{p})\right| > 0 \quad \forall \mathbf{p} \in \mathcal{D} \setminus \partial\mathcal{D}, && (5.8) \\ & \Omega(\partial\mathcal{D}) = \mathcal{M}, && (5.9) \end{aligned}$$

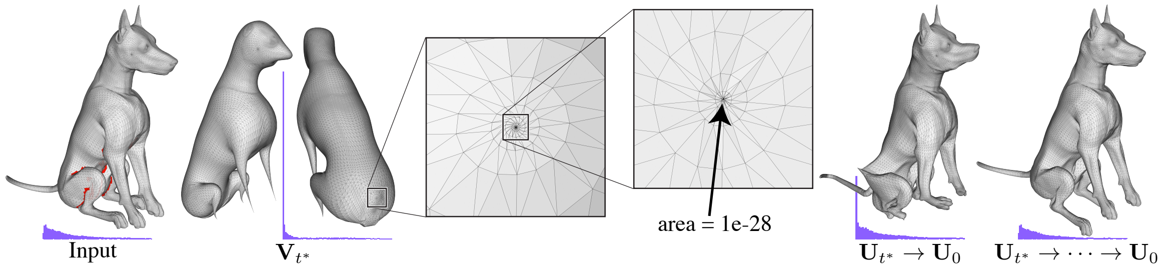
where E_{vol} is an arbitrary non-negative energy. In fact, since we only care about the map Ω insofar as we care about the constraints (5.8-5.9), it is helpful conceptually to consider simply $E_{\text{vol}}(\Omega) = 0$. This is in contrast to the typical, variational parameterization or deformation problems, where energies are carefully crafted to minimize distortion or satisfy problem-specific needs.

5.4 Discretization

We have described our solution in the continuous case, and now we must discretize it in order to mesh self-intersecting input surfaces. Let us restrict our input shape in $\mathbb{R}^3$ to be a closed, orientable, triangle mesh described by a list of n vertices $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, $\mathbf{v}_i \in \mathbb{R}^3$ and a list of m triangle facets $\mathbf{F} = \{f_1, f_2, \ldots, f_m\}$ where $f_i \in \{1, 2, \ldots, n\}^3$. Our goal is then to find a set of tetrahedral elements $\mathbf{E} \subset \{1, \ldots, k\}^4$ defined over a set of vertices $\mathbf{V_E} \supseteq \mathbf{V}$ which represent the overlapping volume of $(\mathbf{V}, \mathbf{F})$. In general, $k > n$ since we may add Steiner points.

The discrete analogs to the requirements (5.1-5.3) are:

1. all triangles in $\mathbf{F}$ appear as faces of boundary tets in $\mathbf{E}$,
2. the signed volume of each tet in $\mathbf{E}$ is positive, and
3. $\mathbf{E}$ forms a combinatorial 3-manifold with boundary.



area = 1e-28
Input
$\mathbf{V}_{t^*}$
$\mathbf{U}_{t^*} \rightarrow \mathbf{U}_0$
$\mathbf{U}_{t^*} \rightarrow \cdots \rightarrow \mathbf{U}_0$

Figure 5.7: cMCF removes intersections on the *Dog* producing $\mathbf{V}_{t^*}$, but also introduces enormous scaling: flowed tail shown in nested insets. Attempting to optimize directly back to the original metric ($\mathbf{U}_{t^*} \rightarrow \mathbf{U}_0$) finds a self-intersection free immersion, but with heavy distortion. Our reverse flow using intermediary steps ($\mathbf{U}_{t^*} \rightarrow \cdots \rightarrow \mathbf{U}_0$) instead finds a self-intersection free and low distortion solution. Purple histogram overlays show the distribution of triangle areas.

Forward flow. Discretizing the cMCF Φ_t follows exactly as described in (*Kazhdan et al.*, 2012). We compute the cotangent Laplacian of the original mesh $(\mathbf{V}, \mathbf{F})$ and then for each discrete step δ forward in time we update the mass matrix according to vertex positions $\mathbf{V}_t$ of the current immersion Φ_t. In addition to updating the flow, at each time step we detect if any self-intersections remain. We do this by computing all triangle-triangle intersections using the exact predicates (but inexact construction) kernel in CGAL. We, of course, stop early as soon as an intersection is found (see Figure 5.5). To efficiently compute the intersections we use the box intersection implementation provided by CGAL that exploits spatial hierarchies.

Reverse flow. We follow forward in time by discrete steps δ ($\sim 10^{-4}$) until no self-intersections remain, resulting in $\mathbf{V}_{t^*}$ (see Figure 5.5). Then, we flow *in reverse.* Starting with $t = t^*$ and initializing $\mathbf{U}_{t^*} \leftarrow \mathbf{V}_{t^*}$, we minimize the surface-based elastic energy, which treats $\mathbf{V}_{t-\delta}$ as the rest-state, using $\mathbf{U}_t$ as the initial guess of the unknown positions $\mathbf{U}_{t-\delta}$ (see Figure 5.6).

We implement this in a fashion similar to the dynamics method of *Chao et al.* (2010) with three notable differences. Instead of a volumetric energy we use a surface-based as-rigid-as-possible (ARAP) energy, in particular the "spokes-and-rims" energy described in their Section 4.2. Instead of the Newton solver proposed by Chao et al., we use a "local-global" solver as described by *Sorkine and Alexa* (2007). This is simpler to implement and avoids expensive Hessian computations. Finally, we need absolutely safe collision detection and response. To handle this we detect all intersecting triangles for each time step in the simulation, again using CGAL. All vertices of each offending triangle are fixed to their previous positions, and the solution for that time step is resolved recursively until no intersections remain. For all vertices of each intersecting pair of triangles we also accumulate repulsion forces to be used for the next time step. These force vectors are the difference between the barycenters of the triangles times a scalar weighting term (for all results in this paper we have defined this weight as 1000, but good results are obtained for the range $[1, 1000]$). The accumulated forces for each vertex are defined as the external forces only for the next time step.

We opt for dynamics, rather than worry about fixing enough vertices in $\mathbf{U}_t$ to remove the translational and rotational degrees of freedom in the under-constrained ARAP energy. Adding dynamics to an ARAP energy optimization is simple. Suppose our unknown positions at *simulation* time i are $\mathbf{U}_t^i$, then we begin with Netwon's second law:

$$\mathbf{f}_{\text{ext}} + \mathbf{f}_{\text{int}} = \mathbf{M}\mathbf{a}^i \tag{5.10}$$

$$\mathbf{f}_{\text{ext}} + \nabla E_{\text{ARAP}}\left(\mathbf{U}_t^i\right) = \mathbf{M}\mathbf{a}^i \tag{5.11}$$

where $\mathbf{f}_{\text{ext}} \in \mathbb{R}^{n\times 3}$ and $\mathbf{f}_{\text{int}} \in \mathbb{R}^{n\times 3}$ are the external and internal forces respectively, $\mathbf{M} \in \mathbb{R}^3$ is the (diagonalized) mass matrix, and the unknown accelerations are discretized with finite differences in *simulation* time $\mathbf{a}^i = \left(\mathbf{U}_t^i - 2\mathbf{U}_t^{i-1} + \mathbf{U}_t^{i-2}\right)/\epsilon^2$, where ϵ is the *simulation* time step (not to be confused with the flow time step δ). We immediately treat the gradient of our ARAP energy E_{ARAP} as an internal force. The anti-derivative of Equation (5.11) w.r.t. ${\mathbf{U}_t^i}^{\mathsf{T}}$ results in a time dependent energy:

$$E_{\text{dyn}}\left(\mathbf{U}_t^i, \mathbf{U}_t^{i-1}, \mathbf{U}_t^{i-2}\right) = \frac{\epsilon}{2}{\mathbf{a}^i}^{\mathsf{T}}\mathbf{M}\mathbf{a}^i - {\mathbf{U}_t^i}^{\mathsf{T}}\mathbf{f}_{\text{ext}} + E_{\text{ARAP}}\left(\mathbf{U}_t^i\right).$$

Notice that in terms of the implementation of a local-global solver, the addition of dynamics only affects the global Poisson solve, and only the right-hand side changes as simulation time i advances. Thus a Cholesky decomposition may still be prefactored. The local step (SVDs for best-fit rotations) is not affected. More details may be found in (*Jacobson et al.*, 2012). Intuitively, the dynamic simulation corresponds to introducing an energy term at each iteration that gently pulls the current guess toward the previous guess. This creates drag, but also regularizes the Poisson solve at the "global" step and enables the aforementioned repulsion forces. Finally, after each simulation is completed, we register $\mathbf{U}_t$ to $\mathbf{V}_t$ with a globally optimal rigid transformation (*Sorkine*, 2009).

We repeat this simulation for each discrete flow time t, resulting in a sequence of self-intersection free immersions $\{\mathbf{U}_{t^*}, \ldots \mathbf{U}_t, \ldots, \mathbf{U}_0\}$ (see Figure 5.6). Most importantly, $\mathbf{U}_0$ is intrinsically similar to our original mesh geometry $\mathbf{V}_0$, but free of self-intersections. We mesh the interior of $(\mathbf{U}_0, \mathbf{F})$ using TetGen (*Si*, 2003), setting parameters to achieve slightly graded elements, but with sufficient circumradius-to-edge ratio (~ 2). This produces a tet-mesh with elements $\mathbf{E}$ and vertex positions $\mathbf{U}_\mathbf{E} \supseteq \mathbf{U}_0$.

Though only auxillary, the intermediary steps $\mathbf{U}_t$ are essential as initial guesses for each step backward in the flow. Immediately optimizing for $\mathbf{U}_0$ treating $\mathbf{V}_{t^*}$ as an initial guess results in many collisions early on, locking the surface in an unsatisfactory shape (see Figure 5.7).

Volumetric parameterization. The ill-posed energy optimization in Equation (5.7) could be discretized into a piecewise-linear mapping $\Omega : (\mathbf{U}_\mathbf{E}, \mathbf{E}) \rightarrow \mathbb{R}^3$ in a variety of ways. We examine two choices. First we could consider treating the local injectivity constraint in Equation (5.8) as a *weak* constraint, translating it into an energy term which punishes flipped elements. One available energy is the elastic energy for tetrahedral meshes described in (*Chao et al.*, 2010). We can minimize this energy (subject to the boundary constraints implied by Equation (5.9)) with the same local-global solver used earlier, only now the energy is volumetric, defined with $(\mathbf{U_E}, \mathbf{E})$ as the rest state.

This discretized energy punishes flipped elements, but only with finite energy[2]. For most applications, flipped elements are undesired, leading to inaccuracies. For some applications like physically based simulation or mesh refinement, flipped elements are a deal-breaker. While mesh untangling methods such as (*Freitag and Plassmann*, 2000; *Knupp*, 2001) could be employed, they are slow and ignore our extra information of a self-intersection free state $(\mathbf{U_E}, \mathbf{E})$. Therefore, we consider another option: treating the constraints Equation (5.8) as *hard* constraints.

We employ a solver for exactly this problem: finding locally injective maps (*Schüller et al.*, 2013). Rather than use an arbitrary constant energy for E_{vol}, we notice that, while we do not care about the distortion of the mapping (we can always refine as a post process), choosing an energy that punishes tetrahedra with degenerating unsigned volumes assists (*Schüller et al.*, 2013) in finding a locally injective map. To this end, we employ Green's strain energy (refer to (*Schüller et al.*, 2013) for the definition).

Additionally, since we do not care about the distortion of the internal mapping, we may further assist (*Schüller et al.*, 2013) by refining our tet mesh during optimization. For the volume parameterization and deformation problems considered by *Schüller et al.* (2013), this is in general not possible or desired. In our case, we notice that this greatly improves the chances of finding a feasible solution. We refine

[2]In terms of the derivation in (*Chao et al.*, 2010), this can be chalked up to discretization error as the continuous energy should be infinite.

every 100 iterations using TetGen's coarsen-then-refine option. This first removes existing internal Steiner points and then adds new points, ensuring the same quality as discussed earlier and avoiding an explosion in the mesh size. Refining the current solution to a new mesh $(\mathbf{U}'_\mathbf{E}, \mathbf{E}')$ effectively redefines the domain of our mapping, $\Omega : (\mathbf{U}'_\mathbf{E}, \mathbf{E}') \rightarrow \mathbb{R}^3$, but by abuse of notation we are still finding a solution to our original problem.

The problem in $\mathbb{R}^2$ is far more studied, and many options for locally injective mappings exist. While the method of *Schüller et al.* (2013) outperforms other methods such as (*Lipman*, 2012), (*Xu et al.*, 2011) present an even faster solution, guaranteed to detect feasibility. Unfortunately, while extending the *implementation* of *Xu et al.* (2011) to $\mathbb{R}^3$ is straightforward, there is no proof (yet) that the guarantees extend as well. In our experiments, we found that for simple examples extending (*Xu et al.*, 2011) to $\mathbb{R}^3$ succeeds, but for more complicated inputs numerical issues arose before convergence could be reached or infeasibility could be determined.

5.5 Experiments and results

We report statistics in Table 5.1. Timings were obtained on an iMac Intel Core i7 3.4GHz computer with 16GB memory. We implement our method primarily as a serial program in MATLAB. Self-intersections are determined using the CGAL C++ library, and meshing is performed with Tetgen. Neither are a bottle-neck. The locally injective mapping method of *Schüller et al.* (2013) is also implemented as a serial subroutine written in C++.

We use (*Schüller et al.*, 2013) for all of our examples, except the *Dog* in Figure 5.7 for which it does not find a solution. Finding locally injective mappings is a difficult and unsolved problem, but as new methods appear in these areas, our approach will immediate see benefits. Switching to ARAP for this example is a possibility though doing so produces 654 flipped tetrahedra out of 147588. We also tried ARAP on other

Input model			Computation time			Output
Name	$\|\mathbf{F}\|$	#s.i.	Φ_{t^*}	Ψ_0	Ω	$\|\mathrm{E}\|$
Decimated	500	7	0.1	1	0.8	694
Leg	13230	239	0.4	125	30	23968
Cheese	15944	368	0.4	340	1	24447
Male	30788	1133	0.9	200	223	78647
Hand	44000	1924	1.6	435	1922	86000
Dog	50576	1042	2.2	523	-	-
Polygirl	65800	679	0.6	140	1642	122118

Table 5.1: Statistics for the various examples. $|\mathbf{F}|$ is the number of facets in the input 3D surface and #s.i. the number of intersecting pairs of facets. We report timings for each stage of our algorithm in seconds: (Φ_{t^*}) computing the cMCF flow until self-intersections are removed, (Ψ_0) computing reverse flow preventing self-intersections, (Ω) computing volumetric map. The number of elements in the output tet mesh is $|\mathrm{E}|$.

examples where (*Schüller et al.*, 2013) succeeds: for the *Hand* in Figure 5.1, ARAP flips 325 out of 86000. Both the *Dog* and the *Hand* highlight the intrinsic quality of our final reverse flow surface.

The *Dog* model in Figure 5.7 contains multiple self-overlapping parts, all resolved by cMCF. However, flowing with cMCF comes at a cost: the entire tail has shrunk around a small point on the *Dog*'s behind. The smallest triangle area is far too small for TetGen to deal with. Instead, our reverse flow restores the intrinsic shape of the dog. This surface, and a tet-mesh generated in it, is ready to use for physically based simulation applications.

In Figure 5.1, the *Hand* is also restored well by our reverse flow. As desired, when mapped to match the input surface, the generated tetrahedral mesh has overlapping tetrahedra on the fingers: this is made obvious when solving the biharmonic equation with alternating Dirichlet boundary conditions on the finger tips.

Geodesic distance computation is a cornerstone of geometry processing. In Figure 5.8, we compare *surface* geodesic distances computed on a self-intersecting input with *volumetric* geodesic distances computed on our consistently overlapping output

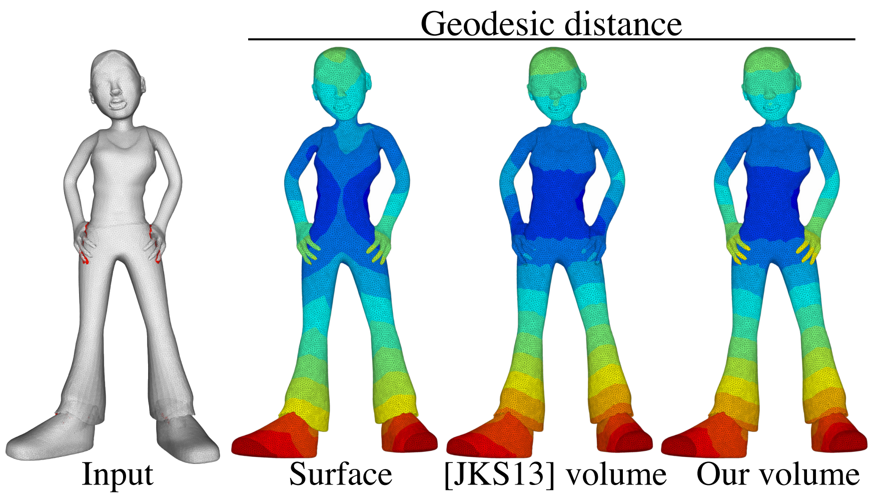


Figure 5.8: *Polygirl*'s hands intersect her waist (left). Geodesic distance to a point on her back computed on the surface correctly separates the hands, but measures around the waist instead of through it. The positive winding number mesh of *Jacobson et al.* (2013) [JKS13] has the opposite situation. Using our output volume mesh, both are correct.

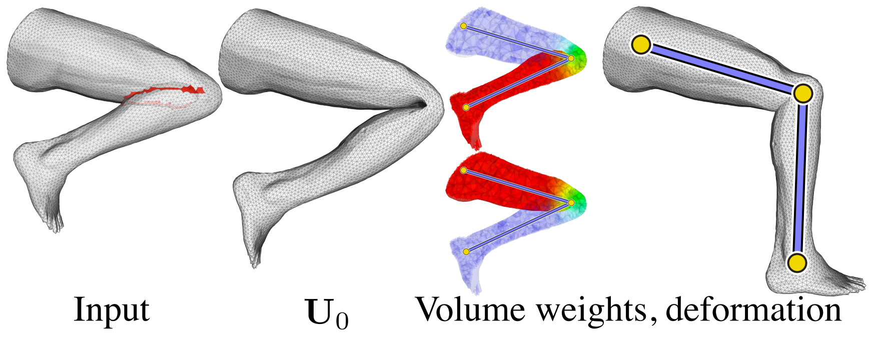


Figure 5.9: Our method enables automatic skinning weight computation with (*Jacobson et al.*, 2011).

tet mesh and the fused mesh of *Jacobson et al.* (2013). Our mesh reveals the semantically distant waist and hands, without compromising the volumetric distance through the body.

The *Leg* in Figure 5.9 overlaps itself non-trivially in a 3D analog to the elbow case in Figure 5.4. Our method resolves this and volumetric skinning weights for a manually defined skeleton are generated using Bounded Biharmonic Weights (BBW) (*Jacobson et al.*, 2011). Each weight function correctly controls the corresponding portion of the *Leg*, despite the spacial overlap. Deforming the mesh with linear blend

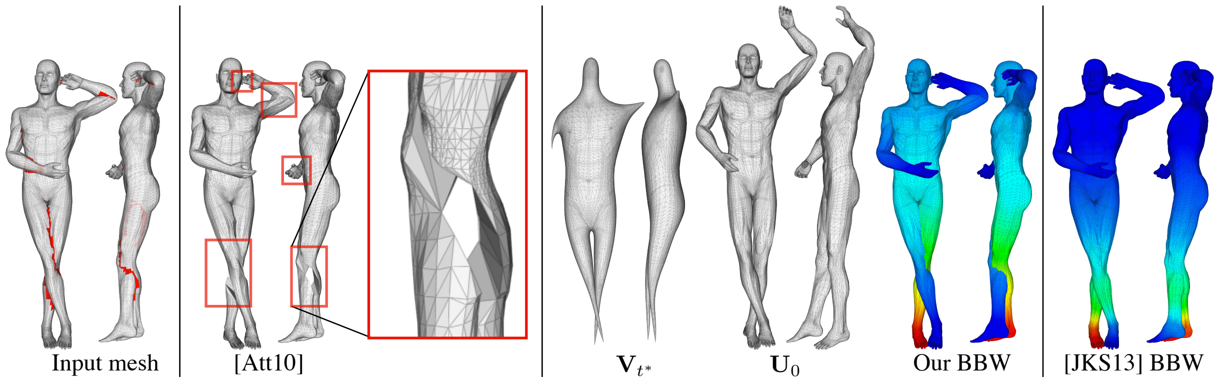


Figure 5.10: Left to right: the limbs of *the Male* intersect each other and the body (front and sideview). MeshFix of *Attene* (2010) [Att10] successfully removes these intersections, but local modifications are too aggressive. The cMCF flow removes intersections, resulting in $\mathbf{V}_{t^*}$, and our reverse flow restores the original shape intrinsics $\mathbf{U}_0$. We use *Schüller et al.* (2013) to map this to the interior of the input surface and compute BBW weights for point handles at each extremity (left foot visualized). Meshing according to positive winding number (*Jacobson et al.*, 2013) [JKS13] glues the limbs together, producing unsatisfactory weights.

skinning reveals this as the *Leg* extends without artifacts.

Mesh repairing algorithms can be used to resolve intersections from a surface mesh, such that it can be subsequently tetrahedralized. We show in Figure 5.10 the watertight output of MeshFix (*Attene*, 2010). Although a tet-mesh could be generated, much of the surface of the *Male* has been deleted or altered (see inside red rectangles). Instead, our method resolves self-intersection without modifying the input's connectivity. The result of our reverse flow is meshed and the mesh is mapped back to the original surface, where BBWs may be computed for each extremity. An alternative, is to consider the interior as defined by the positive winding number (*Jacobson et al.*, 2013). This glues semantically distant regions together resulting in poor BBWs.

Reduced elastic simulations (*Jacobson et al.*, 2012) are computed for these two tet-mesh and BBW results and compared in Figure 5.11. Notably the limbs in our mesh separate freely, while those of *Jacobson et al.* (2013) are awkwardly stuck together.

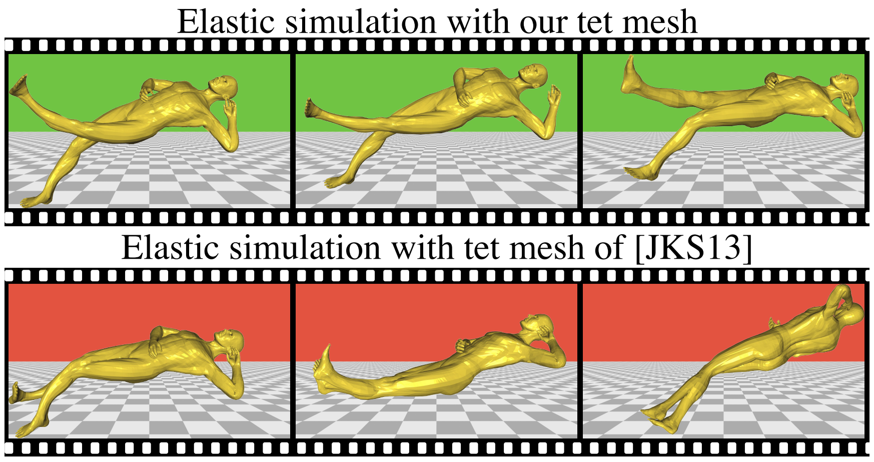

Elastic simulation with our tet mesh
Elastic simulation with tet mesh of [JKS13]

Figure 5.11: Our output tetmesh is consistent with the self-intersections of the original shape. This allows limbs to move freely during elastic simulation of *the Male* (top). Meshing according to positive winding number glues semantically distant regions together *Jacobson et al.* (2013) [JKS13], causing the legs to stick together and the arms to stick to the belly and head (bottom).

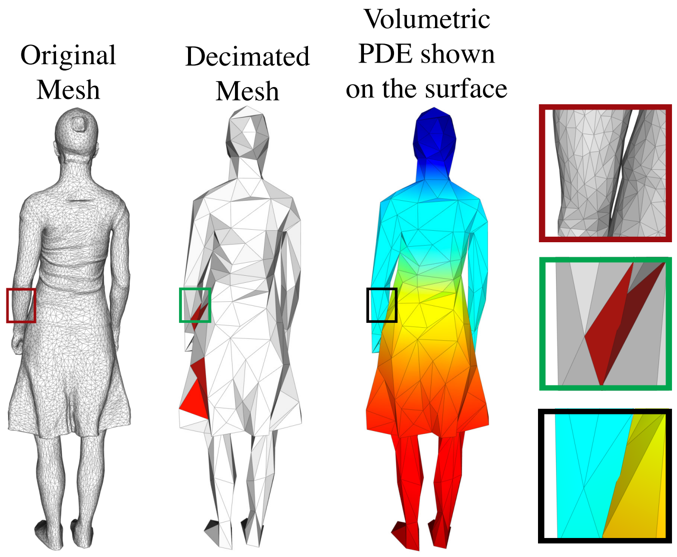


Figure 5.12: A dense mesh is decimated, and the result presents self-intersections. Our method successfully defines a tet-mesh for its interior, on which we can solve the bi-harmonic PDE.

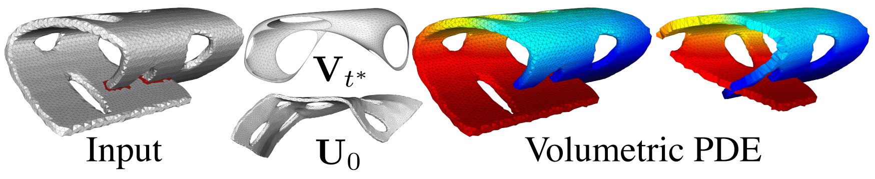


Figure 5.13: We observe that for some cases—such as this thick slice of *Cheese*—self-intersections are also removed from high genus shapes before cMCF converges.

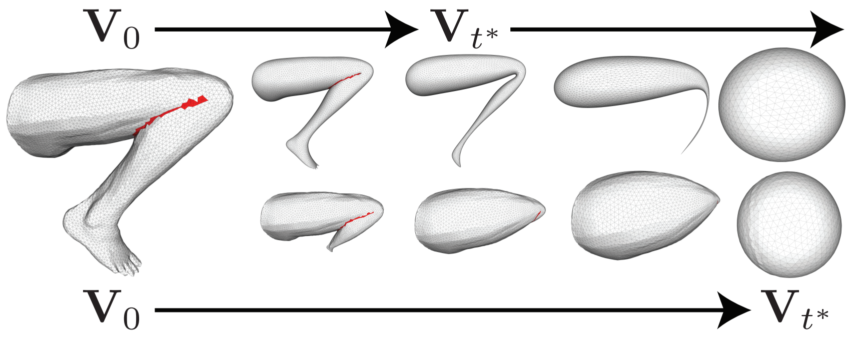


Figure 5.14: Conformalized mean curvature flow (top) removes self-intersections much earlier than conformal Willmore flow (bottom).

To perform complex volumetric physical simulations, it is common to first decimate surface meshes to a very coarse level and then tetrahedralize their interior. We show in Figure 5.12 that decimation can introduce self-intersections and that our method is able to define a tet-mesh for the interior of the decimated surface.

5.5.1 Limitations and future work

We would like to improve the computational performance of our reverse flow dynamics in future work, perhaps by employing a subspace reduction method such as (*Jacobson et al.*, 2012), though the incorporation of safe contact response is not obvious.

cMCF and other flows such as Willmore, do not, in general, converge to self-intersection free surfaces. Thus we have no guarantee that our method will work for high-genus shapes. However, we found that self-intersections are often removed by cMCF early on, even for high genus shapes (see Figure 5.13). This is in contrast

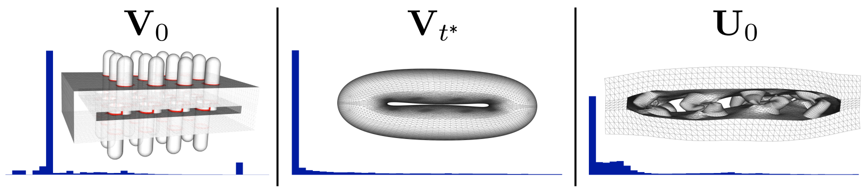


Figure 5.15: The input surface $\mathbf{V}_0$ is taken to a self-intersection-free state $\mathbf{V}_{t^*}$, but the reverse flow cannot restore the original shape due to the lock caused by the external torus on the bristles. Histograms below each surface show the triangle area distribution and confirm that the result of the reverse flow was not able to restore the original areas.

to the conformal Willmore flow (*Crane et al.*, 2013b). Our experiments show that this method removes all intersections at a much later state (see Figure 5.14). Using (*Crane et al.*, 2013b) as the forward flow in our pipeline would make it difficult for the reverse flow to restore the shape. Although it seems that this new flow works better for high-genus surfaces, there is no guarantee of convergence to a self-intersection free shape. Investigating other flows or modifications of existing flows for removing and *preventing* self-intersections is an interesting direction for future work.

Although we use the state-of-the-art locally-injective mapping method of *Schüller et al.* (2013), the optimization may sometimes get stuck, unable to satisfy the boundary conditions. Their method treats local injectivity as hard constraints using a barrier method, but the positional constraints on the boundary are only enforced with quadratic penalty terms. We do not report the result for the *Dog* model since (*Schüller et al.*, 2013) was not able to satisfy all the soft constraints.

In some cases the reverse flow does not succeed in restoring the original shape. In Figure 5.15 we show an example where the bristles inside a torus need more space to restore their original geometry and get locked by the torus. Histograms of triangle areas (bottom) show that the geometry of the result of the reverse flow is not similar to the input shape.

Finally, even some sphere-topology shapes will not work with our method. Shapes implying regions of negative winding number (see Figure 5.16) are ill posed as they do

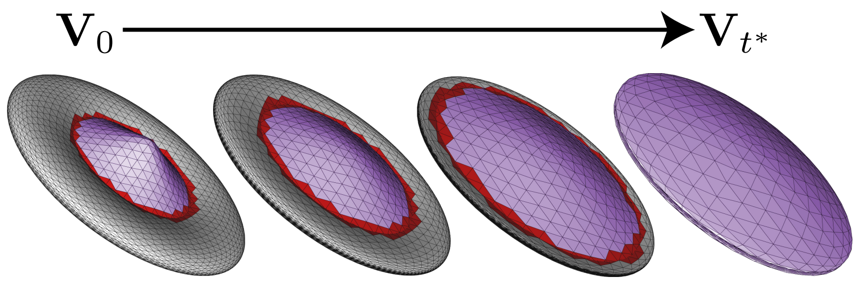


Figure 5.16: This *Flying Saucer* shape is created by pulling one side through the other (back-faces are purple). The region in the middle has negative winding number: meshing it is ill-posed, so this is not valid input for our method. Interestingly, cMCF does remove all self-intersections, but flips the entire shape inside-out in the process.

not define a clear "interior." Our method can reject these immediately, but it remains to prove the sufficient conditions for invalid input to our problem.

5.6 Conclusion

Our discretized formulation proves to be a powerful tool for consistently meshing, previously *unmeshable* models. Our reverse flow takes maximal advantage of the cMCF, which is computed anyway to remove self-intersections. This complements modern tet-meshing software and state-of-the-art bijective parameterization, forming a useful tool to assist in a variety of geometry processing tasks. We hope that by providing a method to recover volumetric discretizations for self-intersecting surfaces of solid shapes we will encourage volumetric processing at every applicable stage in the geometry processing pipeline.

CHAPTER VI

Conclusion and Future Work

In this thesis we presented new algorithms for two distinct fields of visual computing: image processing and geometry processing. We also presented the theory behind both algorithms. This theoretical background involved many areas of mathematics, including functional analysis, differential geometry, optimization, numerical analysis and others.

The first method proposed new digital filters and generators for the sampling and reconstruction modern pipeline. Our experiments showed that our approach is superior when compared to previous methods for a similar computational cost.

The second part of the thesis presented the first method to tetrahedralize the interior of self-intersecting triangle meshes. We illustrated how useful this method can be with many examples from practical tasks such as physical simulation, deformation and geodesic distance computations.

The background for both methods is going to be useful for proposing other new algorithms in visual computing. In fact, we have already been working on them and show some preliminary results on the sequel.

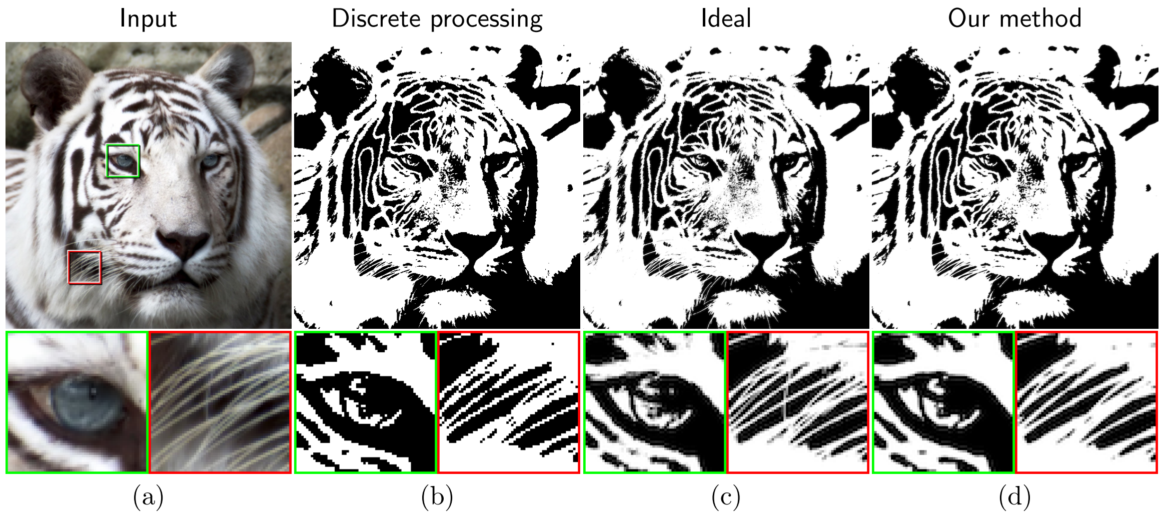


Figure 6.1: A continuous function f (a scene) is prefiltered and sampled, resulting in an image shown in (a), on which we wish to apply, for example, a threholding operation T. (b) When the transformation is applied directly to image samples, aliasing artifacts are visible. (c) Ideally, we should have prefiltered $T(f)$, but f may no longer be available. (d) Our continuous image-processing algorithm first reconstructs the image, then transforms it in the continuous domain, before finally prefiltering the result to generate the output. Artifacts are eliminated.

6.1 Approximation theory for real-time image processing

We start with an input image f that has been prefiltered with a good anti-aliasing prefilter ξ before being sampled ((a) in figure 6.1). In the notation introduced in chapter II, this input is given by

$$\boldsymbol{i} = [\![f * \xi^\vee]\!]. \tag{6.1}$$

We now want to apply some image operation T to this image, such as contrast enhancement, thresholding, bilateral filtering, laplacian edge-enhancement, etc. Most of these operations are highly nonlinear, even discontinuous in many cases.

Due to simplicity current image processing software define the output of applying

T to $\boldsymbol{i}$ as

$$[\ldots, T(i_{-2}), T(i_{-1}), T(i_0), T(i_1), T(i_2), \ldots], \tag{6.2}$$

i.e., simply evaluate T at the samples of f. This results in aliasing artefacts such as the jagged edges shown in figure 6.1 (b).

Our method consists of defining a continuous reconstructions $\tilde{f}$ using the techniques presented in chapter II, applying the operator T to it, prefiltering the result with some good prefilter ρ and then sampling to obtain the output. Thus, the result of our method is given by

$$\boldsymbol{[}T(\tilde{f}) * \rho^{\vee}\boldsymbol{]}. \tag{6.3}$$

We show one of our results in figure 6.1 (d).

Of course this process is more expensive than the usual approach but we leverage new methods for function reconstruction (with B-splines) in the GPU. This allows us to produce high-quality results at real time.

6.2 Physical simulation for the generation of cage meshes

Given a fine mesh we want to automatically calculate a cage mesh to be used in multigrid computations and deformation. A cage is a mesh that encloses the input one.

Figure 6.2 illustrate our 2D prototype. We first calculate a simplification of the input mesh (blue mesh in figure 6.2 (left)) to be an initial guess for the output cage. The input mesh (dashed in figure 6.2 (left)) is shrunk using the conformalized mean-curvature flow presented in section 4.2.1, but now without renormalizing at each step. Once the fine mesh no longer intersects the coarse one, we stop shrinking. The result

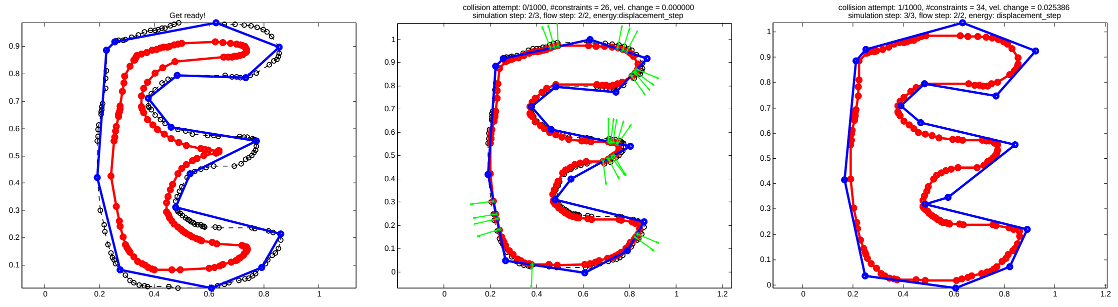


Figure 6.2: The 2D prototype of our method for automatic cage generation. Left: an input mesh is first shrunk until it no longer intersects its coarse simplified version. Center: the fine mesh is grown back to its initial position. When it hits the coarse mesh, collision normals are defined and the coarse mesh moves away responding to these collision. Left: final blue mesh is the output cage.

of this shrinking is the red mesh in figure 6.2 (left).

We then step back in the flow and every time a collision between red and blue meshes happens, we define a collision normal and an energy to push the blue mesh away (figure 6.2 (center)). The physical simulation guarantees that the coarse (blue) mesh always encloses the fine (red) mesh. When the fine mesh is back to its initial position, the blue mesh is a cage for it.

BIBLIOGRAPHY

BIBLIOGRAPHY

Aldroubi, A., and M. Unser (1994a), Sampling procedures in function spaces and asymptotic equivalence with Shannon's sampling theory, *Numerical Functional Analysis and Optimization*, *15*(1–2), 1–21.

Aldroubi, A., and M. Unser (1994b), Sampling procedures in function spaces and asymptotic equivalence with Shannon's sampling theory, *Numerical Functional Analysis and Optimization*, *15*(1-2), 1–21.

Attene, M. (2010), A lightweight approach to repairing digitized polygon meshes, *The Visual Computer*, *26*(11), 1393–1406.

Baraff, D., A. Witkin, and M. Kass (2003), Untangling cloth, *22*(3), 862–870.

Blu, T., and M. Unser (1999a), Quantitative Fourier analysis of approximation techniques: Part I—Interpolators and projectors, *47*(10), 2783–2795.

Blu, T., and M. Unser (1999b), Approximation error for quasi-interpolators and (multi-)wavelet expansions, *Applied and Computational Harmonic Analysis*, *6*(2), 219–251.

Blu, T., P. Thénavaz, and M. Unser (1999), Generalized interpolation: Higher quality at no additional cost, pp. 667–671.

Blu, T., P. Thévenaz, and M. Unser (2001), MOMS: Maximal-order interpolation of minimal support, *10*(7), 1069–1080.

Blu, T., P. Thévenaz, and M. Unser (2003), Complete parametrization of piecewise-polynomial interpolation kernels, *12*(11), 1297–1309.

Blu, T., P. Thévenaz, and M. Unser (2004), Linear interpolation revitalized, *13*(5), 710–719.

Botsch, M., M. Pauly, M. Wicke, and M. Gross (2007), Adaptive space deformations based on rigid cells, *26*(3), 339–347.

Botsch, M., L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy (2010), *Polygon Mesh Processing*, AK Peters.

Bridson, R., R. P. Fedkiw, and J. Anderson (2002), Robust treatment of collisions, contact, and friction for cloth animation, *21*(3), 594–603.

CGAL (), CGAL, Computational Geometry Algorithms Library, http://www.cgal.org.

Chao, I., U. Pinkall, P. Sanan, and P. Schröder (2010), A simple geometric model for elastic deformations, *29*(4), 38:1–38:6.

Chui, C., and H. Diamond (1990), A characterization of multivariate quasi-interpolation formulas and its applications., *Numer. Math.*, *57*(2), 105–121.

Condat, L., T. Blu, and M. Unser (2005), Beyond interpolation: optimal reconstruction by quasi-interpolation, pp. 33–36.

Crane, K., U. Pinkall, and P. Schröder (2011), Spin transformations of discrete surfaces, *30*(4).

Crane, K., U. Pinkall, and P. Schröder (2013a), Robust fairing via conformal curvature flow, p. to appear.

Crane, K., U. Pinkall, and P. Schröder (2013b), Robust fairing via conformal curvature flow, *ACM Trans. Graph.*, *32*.

Dalai, M., R. Leonardi, and P. Migliorati (2006), Efficient digital pre-filtering for least-squares linear approximation, in *Visual Content Processing and Representation*, *Lecture Notes in Computer Science*, vol. 3893, pp. 161–169.

de Boor, C. (1990), in *Computation of Curves and Surfaces*, edited by M. Gasca and C. A. Micchelli, chap. Quasiinterpolants and the approximation power of multivariate splines, Kluwer Academic.

Dey, T. K., and S. Goswami (2003), Tight cocone: a water-tight surface reconstructor.

Dodgson, N. A. (1997), Quadratic interpolation for image resampling, *6*(9), 1322–1326.

El-Khamy, S. E., M. M. Hadhoud, M. I. Dessouky, B. M. Salam, and F. E. Abd El-Samie (2005), An adaptive cubic convolution image interpolation approach, *14*(3), 235–258.

Eppstein, D., and E. Mumford (2009), Self-overlapping curves revisited, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Field, D. J., and N. Brady (1997), Visual sensitivity, blur and the sources of variability in the amplitude spectra of natural scenes, *Vision Res.*, *37*(23), 3367–3383.

Foley, J. D., A. van Dam, S. K. Feiner, and J. F. Hughes (1990), *Computer graphics: principles and practice (2nd ed.)*, 965 pp., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Freitag, L. A., and P. Plassmann (2000), Local optimization-based simplicial mesh untangling and improvement, *International Journal for Numerical Methods in Engineering*, *49*(1-2), 109–125.

German, I. (1997), Short kernel fifth-order interpolation, *45*(5), 1355–1359.

Geuzaine, C., and J. F. Remacle (2009), GMSH: A 3-D finite element mesh generator with built-in pre- and post-processing , *Numerical Methods in Engineering*.

Harmon, D. (2010), Robust, efficient, and accurate contact algorithms, Ph.D. thesis, Columbia University.

Harmon, D., D. Panozzo, O. Sorkine, and D. Zorin (2011), Interference aware geometric modeling, *30*(6).

Heckbert, P. S. (1986), Filtering by repeated integration, *1986*, *20*(4), 315–321.

Hsiao, W. H., and R. Millane (2005), Effects of occlusion, edges, and scaling on the power spectra of natural images, *Journal of the Optical Society of America A*, *22*(9), 1789–1797.

Jacobson, A., I. Baran, J. Popović, and O. Sorkine (2011), Bounded biharmonic weights for real-time deformation, *30*(4).

Jacobson, A., I. Baran, L. Kavan, J. Popović, and O. Sorkine (2012), Fast automatic skinning transformations, *31*(4).

Jacobson, A., L. Kavan, and O. Sorkine-Hornung (2013), Robust inside-outside segmentation using generalized winding numbers, *32*(4), to appear.

Jia, R., and J. Lei (1993), Approximation by multiinteger translates of functions having global support, *Journal of Approximation Theory*, *72*(1), 2 – 23.

Kazhdan, M., J. Solomon, and M. Ben-Chen (2012), Can mean-curvature flow be made non-singular?

Keys, R. G. (1981), Cubic convolution interpolation for digital image processing, *29*(6), 1153–1160.

Knupp, P. (2001), Hexahedral and tetrahedral mesh untangling, *Engineering with Computers*, *17*(3), 261–268.

Kodak (2010), True colour kodak images.

Kopf, J., A. Shamir, and P. Peers (2013), Content-adaptive image downscaling, *2013*, *32*(6), 173.

Labelle, F., and J. R. Shewchuk (2007), Isosurface stuffing: fast tetrahedral meshes with good dihedral angles, *26*(3).

Li, D., X. Sun, Z. Ren, S. Lin, Y. Tong, B. Guo, and K. Zhou (2012), Transcut: Interactive rendering of translucent cutouts, *19*(3).

Lipman, Y. (2012), Bounded distortion mapping spaces for triangular meshes, *31*(4).

Litman, R., A. Bronstein, and M. Bronstein (2012), Stable volumetric features in deformable shapes, *Computers & Graphics*, *36*(5).

Luo, L., I. Baran, S. Rusinkiewicz, and W. Matusik (2012), Chopper: partitioning models into 3D-printable parts, *31*(6).

Mayer, U. F., and G. Simonett (2000), Self-intersections for the surface diffusion and the volume-preserving mean curvature flow, in *Mean Curvature Flow. Differential and Integral Equations*, pp. 1189–1199.

Mayer, U. F., and G. Simonett (2003), Self-intersections for willmore flow, in *Evolution Equations: Applications to Physics, Industry, Life Sciences and Economics*, pp. 341–348.

Meijering, E. H. W. (2002), A chronology of interpolation: From ancient astronomy to modern signal processing, *Proceedings of the IEEE*, *90*(3), 319–342.

Meijering, E. H. W., W. J. Niessen, and M. A. Viergever (2001), Quantitative evaluation of convolution-based methods for medical image interpolation, *Medical Image Analysis*, *5*(2), 111–126.

Mitchell, D. P., and A. N. Netravali (1988), Reconstruction filters in computer graphics, *1988*, *22*(4), 221–228.

Möller, T., R. Machiraju, K. Mueller, and R. Yagel (1997), Evaluation and design of filters using a Taylor series expansion, *3*(2), 184–199.

Mukherjee, U., M. Gopi, and J. Rossignac (2011), Immersion and embedding of self-crossing loops, in *Proc. SBIM.*

Nehab, D., and H. Hoppe (2014), A fresh look at generalized sampling, *Foundations and Trends in Computer Graphics and Vision*, *8*(1), 1–84.

Nehab, D., A. Maximo, R. S. Lima, and H. Hoppe (2011), GPU-efficient recursive filtering and summed-area tables, *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2011)*, *30*(6), 176.

Nelder, J. A., and R. Mead (1965), A simplex method for function minimization, *The Computer Journal*, *7*(4), 308–313.

Park, S. K., and R. A. Schowengerdt (1983), Image reconstruction by parametric cubic convolution, *23*(3), 258–272.

Ruderman, D. L. (1997), Origins of scaling in natural images, *Vision Res.*, *37*(23), 3385–3398.

Sacht, L., and D. Nehab (2014), Optimized quasi-interpolators for image reconstruction, *Tech. Rep. A5739/2014*, IMPA.

Sacht, L., A. Jacobson, D. Panozzo, C. Schüller, and O. Sorkine-Hornung (2013), Consistent volumetric discretizations inside self-intersecting surfaces, *Computer Graphics Forum (proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing)*, *32*(5), 147–156.

Schaum, A. (1993), Theory and design of local interpolators, *55*(6), 464–481.

Schüller, C., L. Kavan, D. Panozzo, and O. Sorkine-Hornung (2013), Locally injective mappings.

Shannon, C. E. (1949), Communication in the presence of noise, *Proceedings of the Institute of Radio Engineers*, *37*(1), 10–21.

Shen, C., J. F. O'Brien, and J. R. Shewchuk (2004), Interpolating and approximating implicit surfaces from polygon soup, *ACM Trans. Graph.*, *23*(3), 896–904.

Shewchuk, J. (2012), Unstructured Mesh Generation, *Combinatorial Scientific Computing*, *12*, 257.

Shor, P. W., and C. J. Van Wyk (1989), Detecting and decomposing self-overlapping curves, in *Proc. Symp. Computational Geometry.*

Si, H. (2003), TetGen: A 3D Delaunay tetrahedral mesh generator, http://tetgen.berlios.de.

Sigg, C., and M. Hadwiger (2005), Fast third-order texture filtering, in *GPU Gems 2*, edited by M. Pharr, chap. 20, pp. 313–329, Addison Wesley Professional.

Sorkine, O. (2009), Least-squares rigid motion using SVD, *Tech. rep.*, Courant Institute of Mathematical Sciences, New York University.

Sorkine, O., and M. Alexa (2007), As-rigid-as-possible surface modeling, pp. 109–116.

Strang, G. (1971), The finite element method and approximation theory, in *SYNSPADE Proceedings*, edited by A. Press, pp. 547–584.

Strang, G., and G. Fix (1971), A Fourier analysis of the finite element variational method, in *Constructive Aspects of Functional Analysis*, *C.I.M.E. Summer Schools, 2011*, vol. 57, edited by G. Geymonat, pp. 793–840, Springer Berlin Heidelberg.

Strang, G., and G. Fix (1973), A fourier analysis of the finite element variational method, *Constructive Aspects of Functional Analysis*, pp. 795–840.

Thévenaz, P., T. Blu, and M. Unser (2000), Interpolation revisited, *19*(17), 739–758.

Unser, M. (1996), Approximation power of biorthogonal wavelet expansions, *44*(3), 519–527.

Unser, M. (1999), Splines: A perfect fit for signal and image processing, *IEEE Signal Processing Magazine*, *16*(6), 22–38.

Unser, M. (2000), Sampling—50 years after Shannon, *Proceedings of the IEEE*, *88*(4), 569–587.

Unser, M., and A. Aldroubi (1994), A general sampling theory for nonideal acquisition devices, *IEEE Transactions on Signal Processing*, *42*(11), 2915–2925.

Unser, M., and I. Daubechies (1997), On the approximation power of convolution-based least squares versus interpolation, *Trans. Sig. Proc.*, *45*(7), 1697–1711.

Unser, M., A. Aldroubi, and M. Eden (1991), Fast B-spline transforms for continuous image representation and interpolation, *13*(3), 277–285.

Unser, M., A. Aldroubi, and M. Eden (1993), B-spline signal processing: Part II—efficient design and applications, *41*(2), 834–848.

Wang, Z., A. Bovik, H. Sheikh, and E. Simoncelli (2004), Image quality assessment: From error visibility to structural similarity, *13*(4), 600–612.

Willmore, T. (2000), Surfaces in conformal geometry, *Annals of Global Analysis and Geometry*, *18*(3-4).

Xu, Y., R. Chen, C. Gotsman, and L. Liu (2011), Embedding a triangular graph within a given boundary, *Comput. Aided Geom. Des.*, *28*(6).

Zhang, X., and B. A. Wandell (1996), A spatial extension to CIELAB for digital color image reproduction, in *Society for Information Display Symposium Technical Digest*, vol. 27, pp. 731–734.