Poisson Surface Reconstruction on non-Cartesian Lattices

***Abstract*—In this work, we cast the well-known Poisson surface reconstruction algorithm into a more general setting, we reformulate it in terms of shift invariant spaces (spaces that are spanned by lattice translates of an admissible generating function). The treatment is general, but our specific interest is in reconstructing a surface of a solid model from an oriented point-set within function spaces defined over the body centred cubic lattice. We also propose a general framework for approximating the solution to Poisson?s equation in a hypercube with zero Dirichlet boundary conditions, and a new variational scheme to re-sample the initial oriented point-set onto a regular grid. Once the points have been re-sampled onto a grid, the rest of the pipeline is purely based on digital filtering. We also analyze the error incurred via the digital filtering approximation methodology and propose a Fourier domain error kernel that can be used to design solution filters that fully exploit the approximation capabilities of the target space. Finally, we show that a host of Poisson-like methods fail to take advantage of the full approximation spaces over which they are defined.**

I. Introduction

Reconstructing a solid model from a scattered set of points is a common and well studied problem in computer graphics. Interest in this problem is rooted in the desire to convert real life data-sets – data acquired from range scanners and now increasingly from commodity hardware like the Kinect – to 3D polygonal models. The idea also has a applications in model re-meshing and mesh hole filling, when one has a poorly formatted mesh and would like a better representation of it.

Given an oriented point-set, a common approach is to find an implicit function whose iso-contour corresponds to an approximation of the original surface. In this work we take a "signal processing" approach to this problem – we cast the problem into a general shift-invariant setting. More explicitly, we reconstruct the desired implicit function within a function space spanned by lattice translates of a single generating function. The extension to this class of function spaces allows us to consider anchoring our basis functions at non-Cartesian lattice sites (Section III-B). Our approach is general and applicable to any valid lattice and generator, but our implementation is narrowed to applicable function spaces over the *Body Centred Cubic* (BCC) and *Cartesian* (CC) lattices.

The advantage of moving to the BCC lattice is two fold. First, it's well known that the BCC lattice generates the optimal sampling pattern in $\mathbb{R}^3$. The argument for optimality is simple; the sampling of a function with respect to the BCC lattice is equivalent to a periodization of it's frequency spectrum about the BCC's dual lattice, the *Face Centered Cubic* (FCC) lattice. Optimality follows from the observation that the FCC lattice is the optimal sphere packing lattice in three dimensions, that is, it packs frequency replica as tightly as possible in the Fourier domain, resulting in less *pre-aliasing* from sampling. For isotropically band-limited functions, this tighter packing of frequency content allows for about 30% more information to be captured when compared to samplings generated from a Cartesian lattice. The second advantage to moving to the BCC lattice is that there exist reconstruction filters on the BCC lattice that outperform commonly used filters of equivalent order on the CC lattice in terms of both speed and accuracy [1] (in software implementations) – these filters are also more compact than their typical CC counterparts. In the context of our surface reconstruction scheme, this gives rise to regularization matrices that are more sparse on the BCC lattice and tend to provide faster convergence when optimizing the initial point set (Section **??**.)

Our surface reconstruction methodology follows the general Poisson approach, but with a few twists. First, we construct a smoothed approximation to the gradient field of a model's indicator function. The divergence of the gradient field is estimated and subsequently fed into a Poisson solver that outputs the coefficients needed to approximate the smoothed indicator function in a target shift invariant space. This Poisson solver is tailored to cater to the approximation power provided by each space. However, our approach is novel in two notable ways. Firstly, we employ a variational scheme to obtain a lattice-based approximation of the gradient field. This scheme depends only on the generating kernel of the target space and optimizes a functional that incorporates interpolation and smoothness constraints. Secondly, we utilize the theory behind shift-invariant spaces to seek discretizations of the divergence and Laplacian operators that are tailored to exploit the full approximation capabilities of the target space.

Additionally, we consider the possibility of approximating the smoothed gradient field representation within function spaces that are spanned by shifted versions of a single generating kernel. In the context of gradient estimation, introducing a shift in the direction of the derivative has shown to improve overall gradient approximation fidelity in terms of gradient orientation and magnitude, as well as displaying the ability to capture higher frequency details that are smoothed out in non-shifted schemes [2]. Since the divergence operator is intimately connected to the gradient operator, these results are of interest to us. Using an appropriate discretization of the derivative/divergence operator is an important step in recovering the indicator function of the initial model. We also derive specific error bounds, based on an error kernel formulation, for approximations of linear operators in arbitrary shift invariant spaces.

To summarize, our contributions are as follows:

- We reformulate the Poisson surface reconstruction ap-

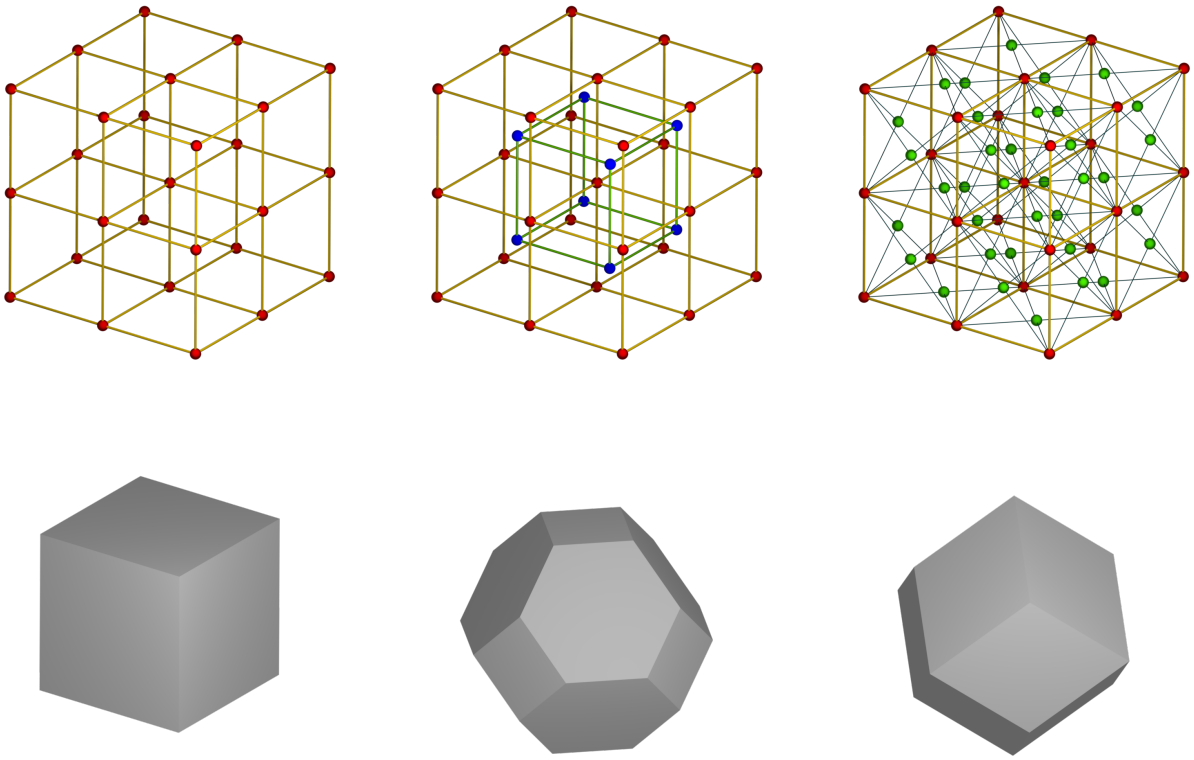


Fig. 1. Sampling lattices (top) with with their corresponding Voronoi regions (bottom). Left to right are the CC, BCC and FCC lattices respectively.

proach using general shift-invariant spaces as the target approximation space
- Specifically, we investigate the reconstruction of surfaces in both a sub-optimal (Cartesian) and an optimal (BCC) box-spline function space
- We present a new variational resampling scheme that resamples the original scattered oriented points' normals onto any regular lattice. This resampling scheme depends on the generating kernel, but is general enough to be extended to shifted generating kernels as this has been shown to improve gradient estimation in shift invariant spaces.
- We present an extension of the error kernel of Blu and Unser to general linear operators and show how to design filters that respect the full approximation order of the target approximation space.
- While our focus is on comparing the surface reconstruction algorithm on both the CC and BCC lattices, we also provide qualitative and quantitative comparisons between the presented technique (on both the Cartesian and BCC lattices) and similar methods.

II. Related Work

Surface reconstruction is still a very active area of research in computer graphics and it's beyond the scope of this work to touch on all the material related to the subject. We therefore very briefly review some relevant contributions – those seeking a more in-depth review may look to a recent survey of the field [3]. We focus mainly on some of the more related and recent developments.

Many surface reconstruction algorithms are so-called "*implicit*" reconstruction algorithms. They attempt to reconstruct an implicit function, $f : \mathbb{R}^3 \mapsto \mathbb{R}$, whose level-set, for some iso-value $s \in \mathbb{R}$, $f(\mathbf{x}) = s$ represents an approximation to the original surface. Other non-implicit techniques generally infer meshes through some other combinatorial means or set predefined logic/rules.

For example, there exist many Delaunay style algorithms in which a triangulation is constructed using a subset of the original point-set [4]. The Power Crust [5] and Cocone [6] algorithms are two other well-known techniques that do not construct implicit functional representations. These techniques often aim to exactly interpolate the input point set, and in the presence of noise tend to over-fit the surface. When the input data are noisy implicit approaches often perform better than non-implicit algorithms, as implicit algorithms inherently attempt to fit locally smooth basis functions to the data, averaging out any noise. Reconstructing either a distance function or an indicator function is common to these approaches.

One of the most well known implicit algorithms is the

algorithm of by Hoppe et al. [7]. In this work a set of tangent planes is constructed to create an approximation to a surface's distance function. A more recent work, is the Smoothed Signed Distance Field reconstruction technique [8], in which an oriented point set imposes constraints on the reconstructed function, its gradient, and its Hessian. These constraints are equvalent to requiring that the reconstructed function be an approximation to the distance field of the original model. Other techniques attempt to reconstruct a signed distance function within a function space spanned by radial basis functions [9]. Recently, there has also been research that incorporates point normals into the radial basis reconstruction framework [10]. It's also possible to construct an implicit function as a normalize sum of compactly supported basis functions defined over an oct-tree, essentially "smearing" the point-set around space and constructing an indicator "shell" around them [11].

Another approach is to find an indicator function that approximates the "inside-outside" function of original model. An early technique [12] transforms the smoothed gradient to the Fourier domain, applies an appropriate operator, then reconstructs it in the spatial domain. This approach often deals well with incomplete, missing, or otherwise noisy data and has the guarantee of creating a water tight surface. However, it involves reconstructing the indicator on a regular grid, which becomes very memory intensive as the grid is refined.

Poisson surface reconstruction techniques make the observation that the Fourier method is equivalently stated as a spatial Poisson problem [13], [14]. Moreover, they seek to rectify the memory limitations of that work, while still maintaining its benefits. However, current techniques based on this idea have been shown to over-smooth the initial point-set, and therefore the resulting surface. [15].

It is also possible to reconstruct the indicator of a point set within a wavelet basis [16]. In this work, an approximate indicator function of the model is projected onto a compact wavelet basis. This leads to a simple and efficient algorithm even when the data sets are massive. However the method's speed comes at a price, as the resulting surfaces are often non-smooth, and display many "jagged" artifacts (although a form of smoothing is applied to the resulting indicator function in an attempt to rectify this.)

Outside of the context of surface reconstruction there has been considerable evidence that shows non-Cartesian approaches to volume rendering [1], [17], [18] and fluid flow [19] outperform Cartesian approaches in both speed and memory consumption. Additionally, finite difference Laplacian stencils on non-Cartesian grids lead to reduced numerical dispersion compared to their Cartesian counterparts (see e.g. [20], [21] and references therein). It is therefore natural to question whether these advantages can be exploited within the context of surface reconstruction.

Our approach aims to solve the Poisson surface reconstruction problem in the setting of general shift-invariant spaces. A crucial step in this regard is the accurate estimation of the divergence of the gradient field from the oriented point-set. Radial basis functions (RBFs) are generally used in the context of the reconstruction of scattered data, however, RBF techniques are often computationally expensive, requiring the solution of large unstable systems of equations (for exact interpolation.) A proposed solution to this problem is to "*re-sample*" the input data onto a regular shift-invariant space. This allows one to use efficient compact reconstruction kernels and data processing techniques [22], [23]. Recently, an extension of this idea to box spline spaces [24] proposes a variational reconstruction framework in which the BCC lattice consistently outperforms the Cartesian lattice. Our method takes a similar variational approach and seeks to construct a smoothed lattice-based approximation of the gradient of the indicator function (Section IV-A). Our employed constraints force the gradient to be close to zero away from the surface, while maintaining a certain degree of smoothness in the resulting approximation. This is preferable in comparison to other works [12]–[14] in which each sample is trilinearly distributed about either a grid or an oct-tree respectively, and is known to over-smooth data.

Having obtained a high-quality lattice-based approximation of the gradient field, we invoke the theory behind shift-invariant spaces and accurately estimate the divergence by applying derivative filters that are designed to harness the full approximation capabilities of the target space (Section IV-B).

Once, the smoothed gradient's divergence has been calculated, recovering the indicator function is a simple matter of taking the inverse of the Laplacian over domain (i.e. solving the Poisson system). To do this, we take an approach that is also inspired by filtering methodologies in signal processing. In particular, we're interested in an approximate solution to Poisson?s equation that lies in a chosen shift-invariant space. Our solution methodology is reminiscent of the finite element method [**?**]. However, we do not impose any boundary conditions on the generating kernels. Rather, we satisfy homogeneous Dirichlet boundary conditions implicitly by requiring that the coefficient sequence be odd (Section III). The solution is obtained by applying a digital filter to point samples. We approach the problem from first principles and derive the tools necessary to design solution filters for a chosen shift-invariant space (Section VI). [[XXX I still need to fix-up section refs/refs here.]]

III. Preliminaries

A. Poisson Surface Reconstruction

We denote the indicator function of a model M as

$$\chi_M(\mathbf{x}) := \begin{cases} 1, & \text{if } \mathbf{x} \text{ is in the interior of the model } M \\ \frac{1}{2}, & \text{if } \mathbf{x} \text{ is on the model's surface} \\ 0, & \text{if } \mathbf{x} \text{ is exterior to } M \end{cases}.$$

The key observation of the Poisson surface reconstruction formulation, is that if one obtains a function $\vec{V}(\mathbf{x})$ with $\vec{V}(\mathbf{x}) \approx \nabla\chi_M$, that approximates the smoothed gradient of the indicator function, finding an approximation to the indicator function reduces to finding a $\tilde{\chi}$ such that

$$\Delta\tilde{\chi} = \nabla \cdot \vec{V}. \tag{1}$$

We define the set of input surface points to be $P := \{(\mathbf{p}_1, \mathbf{n}_1), (\mathbf{p}_2, \mathbf{n}_2) \dots (\mathbf{p}_M, \mathbf{n}_M) : (\mathbf{p}_i, \mathbf{n}_i) \in \mathbb{R}^3 \times \mathbb{R}^3\}$ where $\mathbf{p}_i$ and $\mathbf{n}_i$ are the position and normal vector of the i^{th} sample

respectively. Orientation can be succinctly represented with only two values, however, it is convenient to keep it as the normal at a point, because the length of the normal can be used to encode the sampling density of the point-set at a point. For the following discussion, we also need to introduce the notation $\mathcal{P}_{\mathbf{s}}$, which denotes a patch of the surface model about the sample location $\mathbf{s}$.

Inspired by the approach of Kazhdan et al. [13] the approximation $\vec{V}(\mathbf{x})$ is defined to be

$$\begin{aligned} \vec{V}(\mathbf{x}) &:= \sum_{(\mathbf{s},\mathbf{n})\in P} |\mathcal{P}_s|\, F(\mathbf{x}-\mathbf{s})\cdot\mathbf{n} \\ &\approx \sum_{(\mathbf{s},\mathbf{n})\in P} \int_{\mathcal{P}_s} F(\mathbf{x}-\mathbf{p})\nabla\chi(\mathbf{p})\mathrm{d}\mathbf{p}. \end{aligned} \tag{2}$$

Thus, $\vec{V}(\mathbf{x})$ approximates the gradient of the indicator smoothed by some filter F, broken up over surface patches of the model. We assume the point sampling to be uniform, thus the area of each patch $|\mathcal{P}_s|$ should be a constant, and our resulting approximation will be scaled by an unknown factor. We return to this in Section VI-A. The choice of F places computational constraints on the resulting algorithm.

F is often explicitly chosen to be a compactly supported function. This simplifies and reduces the computational cost of the method when used in conjunction with an oct-tree based representation of $\vec{V}(\mathbf{x})$ [13]. However, the disadvantage of this choice is that F may over smooth data in areas of high curvature [15], much in the same way that a representation of a sampled signal may fail to take advantage of the full approximation power of the function space in which it is represented if it is not properly prefiltered. Theoretically, one could simply reconstruct the surface at higher resolutions, allowing the basis functions to become more "fine", but increasing the total amount of memory used. On a regular grid, this is undesirable. Another approach is to impose a type of "interpolation" constraint on the original data [14] Section **??**.

Ideally, the filter F should correspond to some globally supported RBF. However, as the support of F grows, so does the cost of evaluating it at lattice sites. Moreover, the added benefit of a globally supported RBF is partially lost as the subsequent steps are entirely lattice-based and do not take the RBF into consideration. Our approach circumvents the choice of F entirely by posing the gradient approximation problem as a variational problem that, in addition to imposing the interpolation constraint, also imposes constraints on the "compactness" and "smoothness" of the gradient approximation. The only kernel we need to consider is the one that generates our target shift-invariant space, i.e. the space used to recover the smoothed indicator function. This type of variational reconstruction can be seen as an approximation to radial basis approaches [22], and is well-suited to our Poisson formulation in shift-invariant spaces.

Before presenting our approach (Section IV), we review the necessary background behind shift-invariant spaces and its connection to irregular sampling.

B. Sampling Lattices

In the univariate case, there is only one way to uniformly sample a signal: take equally spaced samples along the independent axis. However, when sampling multivariate signals the situation is not as straightforward. A common approach is to inherit the ideas from the univariate case and sample the function at regular intervals in each axis – a Cartesian sampling – while another might be to place samples at the centres of spheres arranged in the densest possible sphere packing (otherwise known as an FCC sampling). If $\mathbf{L}$ is a unimodular matrix, the set $\mathcal{L} := \{\mathbf{L}\cdot\mathbf{n} : \mathbf{n}\in\mathbb{Z}^3\}$ represents all possible sampling lattices. $\mathbf{L}$ is often called the *generating matrix* for a lattice, and we'll assume that $\mathbf{L}$ is normalized so that $|\det\mathbf{L}| = 1$. The lattice's dual, denoted by $\mathcal{L}^\circ$, is the lattice generated by $\mathbf{L}^{-T}$. Finally, to control the sampling rate of a function a uniform scaling parameter h is introduced – lattices scaled by this parameter will be denoted by $\mathcal{L}_h$. Typically, for the problem of surface reconstruction, our input data are not uniformly distributed in space, or even about the surface of the model, but the concept of a sampling lattice is fundamental in forming the definition of our target approximation spaces (Section III-C).

The Cartesian lattice is given by the generating matrix $\mathbf{L}_C := \mathbf{I}_3$, where $\mathbf{I}_3$ is the 3×3 identity matrix. There are many factors that contribute to the ubiquity of the Cartesian lattice in practice. Possibly one of its most attractive features is its separability. Owing to this, the implementation of generating functions that span function spaces on the Cartesian lattice may be realized by a tensor product extension of existing univariate techniques. Filtering techniques are also easy to implement, as the Fast Fourier Transform is separable and may be applied independently to each axis of the data. However, in terms of sampling efficiency, the BCC lattice is the optimal lattice in $\mathbb{R}^3$, while the Cartesian is sub-optimal.

The Body-Centered Cubic (BCC) lattice is generated by the matrix

$$\mathbf{L}_B := \frac{1}{\sqrt[3]{4}}\begin{bmatrix} -1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix}.$$

However, the BCC lattice is non-separable, and requires special care in data retrieval and indexing. Figure 1 shows these lattices and their Voronoi regions, including the dual of the BCC lattice, the FCC lattice (the Cartesian lattice is dual to itself.)

C. Shift Invariant Spaces

As our target reconstruction spaces, we consider the shift-invariant spaces defined by

$$\mathbb{V}(\varphi,\mathcal{L}_h) := \left\{ g(\mathbf{x}) := \sum_{\mathbf{m}\in\mathbb{Z}^3} c[\mathbf{m}]\varphi(\frac{\mathbf{x}}{h} - \mathbf{L}\mathbf{m}) : c\in\ell_2 \right\},$$

where φ is the generating function for the space, h is a scaling parameter that controls the sampling rate or granularity, and $\mathbf{L}$ is the generating matrix for a given sampling lattice. Representing a function f in a shift invariant space boils down to finding an appropriate coefficient $c[\mathbf{m}]$ vector that brings g as close as possible to the true underlying function f.

Typically, as memory is a limited quantity, we are only interested in a finite number of points on a lattice. We denote the set of such points as $\{\mathbf{s}_1, \mathbf{s}_2, \ \dots \ \mathbf{s}_N\} \subset \mathbf{L}\mathbb{Z}^3$. Corresponding to each lattice site is a coefficient c_k; we collect all such coefficients in the column vector $\mathbf{c} := [c_1 \ c_2 \ \dots \ c_N]^T$. We also use the short hand notation $\varphi_k(\mathbf{x}) := \varphi(\frac{\mathbf{x}}{h} - \mathbf{s}_k)$; from this perspective, our function space is now

$$\left\{ \sum_{k=0}^{N} c_k \varphi_k(\mathbf{x}) \right\} \subset \mathbb{V}(\varphi, \mathcal{L}_h). \tag{3}$$

Further, we denote $\mathbf{P}$ to be the $M \times N$ matrix whose elements are given by sampling the basis functions anchored at each lattice site, or $P_{i,j} := \varphi_j(\mathbf{x}_i)$. Clearly, if our data are sampled at the lattice sites ($\mathbf{x}_i = \mathbf{s}_i$), then the problem of finding an interpolative scheme reduces to finding $\mathbf{c}$ such that $\mathbf{Pc} = \mathbf{f}$, which only involves inverting the square matrix $\mathbf{P}$. However, our sample locations are almost certain to not coincide with lattice sites and we must consider alternative schemes for finding $\mathbf{c}$, which we'll come back to in Section III-H

D. Error Analysis

To facilitate the classical analysis of signals in a shift-invariant space, we need a few notions first. The first tool we need is the Fourier transform of a multivariate function $f(\mathbf{x})$, which we define as $\hat{f}(\boldsymbol{\omega}) := \int_{\mathbb{R}^3} f(\boldsymbol{x}) \exp(-2\pi\imath\boldsymbol{\omega} \cdot \boldsymbol{x}) \mathrm{d}\boldsymbol{x}$. The discrete time Fourier transform (DTFT) of a sequence $c[\cdot]$ associated with a lattice $\mathcal{L}$ is defined as $\hat{C}(\boldsymbol{\omega}) := \sum_{\boldsymbol{m} \in \mathbb{Z}^3} c[\boldsymbol{m}] \exp(-2\pi\imath\boldsymbol{\omega} \cdot (\mathbf{L}\boldsymbol{n}))$. Recall that $\hat{C}(\boldsymbol{\omega})$ is periodic with respect to the dual lattice $\mathcal{L}^\circ$, i.e. $\hat{C}(\boldsymbol{\omega} + \mathbf{L}^{-\mathsf{T}}\boldsymbol{m}) = \hat{C}(\boldsymbol{\omega}), \forall \boldsymbol{m} \in \mathbb{Z}^3$. The notation '$\leftrightarrow$' is subsequently employed to denote transform pairs, i.e. $f(\boldsymbol{x}) \leftrightarrow \hat{f}(\boldsymbol{\omega})$ (Fourier transform pairs) and $c[\boldsymbol{m}] \leftrightarrow \hat{C}(\boldsymbol{\omega})$ (DTFT pairs).

The inner product between two real valued functions f_1 and f_2 belonging to $L_2(\mathbb{R}^3)$ is denoted as $\langle f, g \rangle := \int_{\mathbb{R}^3} f(\boldsymbol{x}) g(\boldsymbol{x}) d\boldsymbol{x}$. Likewise, the inner product between two functions f_1 and f_2 that belong to $L_2(\mathcal{T}^s)$ is denoted as $\langle f_1, f_2 \rangle_{\mathcal{T}^s} := 2^{-s} \int_{\mathcal{T}^s} f_1(\boldsymbol{x}) f_2(\boldsymbol{x}) d\boldsymbol{x}$. The norms induced by these inner products are denoted as $\|\cdot\|_{L_2(\mathbb{R}^s)}$ and $\|\cdot\|_{L_2(\mathcal{T}^s)}$ respectively.

In order to ensure that any function $g \in \mathbb{V}(\varphi, \mathcal{L}_h)$ has a unique representation in terms of a finite energy coefficient sequence $c[\cdot]$, the lattice translates $\{\varphi(\cdot - \mathbf{L}\boldsymbol{n})\}_{\boldsymbol{n} \in \mathbb{Z}^s}$ must form a Riesz basis [**?**], i.e. there exist positive constants a and b such that $\forall c[\cdot] \in l_2(\mathbb{Z}^s)$,

$$a\|c\|^2_{l_2(\mathbb{Z}^s)} \leq \Big\| \sum_{\boldsymbol{n} \in \mathbb{Z}^s} c[\boldsymbol{n}] \varphi(\boldsymbol{x} - \mathbf{L}\boldsymbol{n}) \Big\|^2_{L_2(\mathbb{R}^s)} \leq b\|c\|^2_{l_2(\mathbb{Z}^s)}.$$

E. The Error Kernel

The error $\|f - f_{\mathsf{app}}\|_{L_2(\mathbb{R}^s)}$ between f and its approximation $f_{\mathsf{app}} \in \mathbb{V}(\varphi, \mathcal{L}_h)$ can conveniently be quantified via an error kernel in the Fourier domain. It is defined as

$$E(\boldsymbol{\omega}) := \underbrace{1 - \frac{\left|\hat{\varphi}(\boldsymbol{\omega})\right|^2}{\hat{A}_\varphi(\boldsymbol{\omega})}}_{E_{\mathsf{min}}(\boldsymbol{\omega})} + \underbrace{\hat{A}_\varphi(\boldsymbol{\omega}) \left|\hat{\bar{\varphi}}(\boldsymbol{\omega}) - \hat{\dot{\varphi}}(\boldsymbol{\omega})\right|^2}_{E_{\mathsf{res}}(\boldsymbol{\omega})}. \tag{4}$$

For a bandlimited function f whose spectrum is entirely contained within the Voronoi cell of $\mathcal{L}_h^\circ$ centered around $\boldsymbol{\omega} = 0$, we have

$$\|f - f_{\mathsf{app}}\|_{L_2(\mathbb{R}^s)} = \int_{\mathbb{R}^s} \left|\hat{f}(\boldsymbol{\omega})\right|^2 E(h\boldsymbol{\omega}) d\boldsymbol{\omega}, \tag{5}$$

whereas when $f \in W_2^k(\mathbb{R}^s)$ with $k > \frac{1}{2}$, the integral in (5) gives the L_2-error averaged over all possible shifts of f. Even though this kernel was initially proposed in the univariate setting by Blu and Unser [**?**], multivariate non-Cartesian extensions have also appeared in the literature [**?**], [**?**], [**?**], [**?**], [**?**]. Note that the minimum-error approximation scenario occurs when f is orthogonally projected to $\mathbb{V}(\varphi, \mathcal{L}_h)$, or in other words $\bar{\varphi}$ is chosen to be $\mathring{\varphi}$. polynomial spaces,

F. Approximation Error

If the generating function that spans the space satisfies the *Strang-Fix* conditions of order k, and f belongs to the Sobolev space of at least order k, then it can be shown that the minimum approximation error (in the least-squares sense) decays asymptotically as $O(h^k)$, and the space is said to have approximation order k. We often only have access to f through a discrete amount of samples, $\mathbf{f} := [f_1 \ f_2 \ \dots \ f_M]^T$ located at sample points $S := \{\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_M\}$ respectively. When this is the case, the minimum-error approximation is unattainable. However, interpolative and quasi-interpolative methods [25], [26], which can be seen as oblique projections onto $\mathbb{V}(\varphi, \mathcal{L}_h)$, maintain the same order of accuracy.

For sufficiently smooth functions f that are uniformly sampled and properly prefiltered, the trilinear and linear reconstruction filters (Cartesian and BCC respectively) assure second order convergence, while the tricubic and Quintic assure fourth order. Another nicety of these spaces is that data processing techniques — derivative approximation for example — can often be represented by a convolution of the coefficient vector with either an *Infinite Impulse Response* (IIR) or *Finite Impulse Response* (FIR) filter.

G. Box Splines

Box splines are compact piece-wise polynomial functions $M_{\mathbf{X}} : \mathbb{R}^s \to \mathbb{R}$. The matrix $\mathbf{X}$ denotes a collection of direction vectors, $\mathbf{X} := [\xi_1 \ \xi_2 \dots \xi_n]$. When $n = s$, the function $M_{\mathbf{X}}(\mathbf{x})$ is defined to be $1/|\det \mathbf{X}|$ inside the polytope formed by the Minkowski sum of the direction vectors of $\mathbf{X}$, and zero otherwise (that is, $1/|\det \mathbf{X}|$ when $\mathbf{x} \in \{\xi_1 t_1 + \ldots + \xi_s t_s : t_i \in [0, 1)\}$). When $n > s$, the box spline is recursively defined as

$$M_{\mathbf{X}}(\mathbf{x}) := \int_0^1 M_{\mathbf{X}/\xi_n}(\mathbf{x} - t\xi_n) \mathrm{d}t, \tag{6}$$

where $\mathbf{X}/\xi_n$ is the matrix obtained by removing the column vector ξ_n from $\mathbf{X}$. Box splines are a natural fit for our purposes, and are chosen to span our function spaces. One motivating factor that encourages the use of box splines on the BCC lattice over the Cartesian lattice is that the support of the box splines on the BCC lattice contains less lattice sites as opposed to their Cartesian counterparts. At the same time, they maintain the same order of accuracy and smoothness.

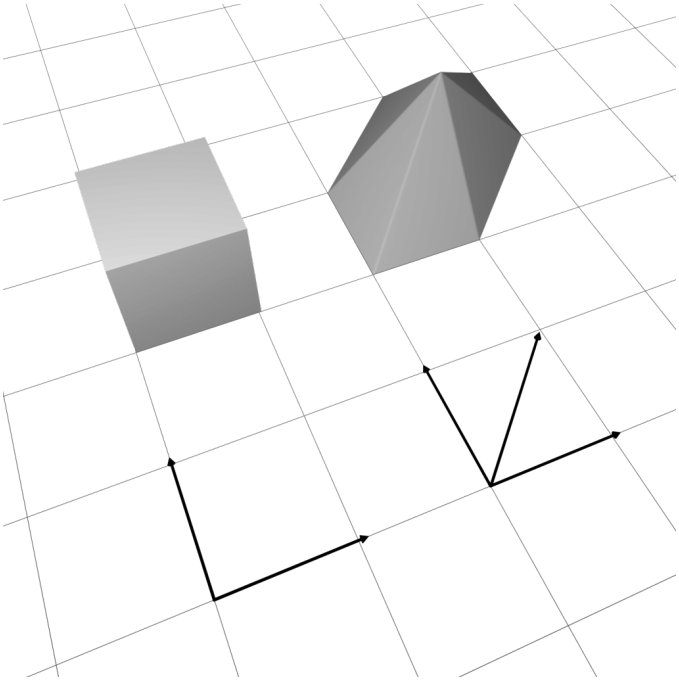


Fig. 2. An example of box splines in 2D. On the left is the box spline comprised of the direction vectors ξ_1, ξ_2, (the case n=s) which is the indicator function of the Minkowski sum of the vectors. The box spline on the right, includes a third vector ξ_3, which can be thought of as "smearing" the first spline along the direction ξ_3.

This implies that one should expect roughly the same quality reconstruction, with less memory accesses per reconstructed value.

The evaluation of a box spline, in general, can be achieved by evaluating a simple recursive expression [27]. However, for many purposes, this form of evaluation is too slow. There has been a line of research [17] aimed towards obtaining fast piecewise polynomial representations of the box splines on the BCC lattice, and showing their advantage in speed and accuracy over splines on the Cartesian lattice when the data to be interpolated are sufficiently smooth.

H. Irregular Sampling

In the work by Kazdhan et al. [12] the range data, P, are splatted via a linear filter onto a regular Cartesian grid. This tends to over-smooth the resulting reconstruction, as it allows the surface to deviate from the original sample locations. Ideally, we would like to find a function that not only interpolates the input data, but provides the representation with the smallest possible "shell" about the initial point set (i.e., the fewest non zero coefficients) while still interpolating the data and eliciting a unique solution.

Thus, we recast the problem of finding a representation of g within our function space as an optimization problem. There are similar works that impose variational constraints on random point samples for images [22], as well as multi-resolution techniques for volumetric data [23] and an extension to box spline spaces [24]. In the same vein, we seek a function $g \in \mathbb{V}(\varphi, \mathcal{L}_h)$ that attempts to interpolate the input data, and minimize a functional E. Specifically, we seek a g that minimizes

$$\sum_{i=1}^{M} (g(\mathbf{x}_i) - f_i)^2 + E(g). \tag{7}$$

It is important to introduce a bit of notation before proceeding, $\mathbf{D}_\mathbf{v} f := \mathbf{v} \cdot \nabla f$ defines the directional derivative of f in the direction $\mathbf{v}$. The Beppo-Levi inner product of order n is defined as $\langle f, g \rangle_{BL^n} := \sum_{|\mathbf{p}|=n} (n!/\mathbf{p}!) \langle \mathbf{D}^\mathbf{p} f, \mathbf{D}^\mathbf{p} g \rangle$, where $\mathbf{p}$ is an integer vector such that $|\mathbf{p}| := p_1 + p_2 + p_3$. We also use the short hand $\mathbf{D}^\mathbf{p} f := \partial^{|\mathbf{p}|} f / (\partial^{p_1} x_1 \partial^{p_2} x_2 \partial^{p_3} x_3)$. The second order Beppo-Levi norm of f is defined as $||f||^2_{BL^2} := \langle f, f \rangle_{BL^2}$.

The second order Beppo-Levi norm can be thought to measure the "smoothness" of a function. However, constraining the smoothness of our approximation alone is not sufficient to elicit the desired solution in our case. We propose that the smoothed normal field should also minimize the inner product of the function with itself. This has the physical interpretation of forcing the solution to be close to zero everywhere except near the input points. Our proposed cost functional is then $E(g) := \lambda_1 ||g|| + \lambda_2 ||g||_{BL^2}$. Here, λ_1 controls the amount to which the function is penalized for large coefficient values. When combined with the interpolation constraint, this imposes the behavior of forcing the function to be zero away from the surface. The parameter λ_2 controls the smoothness of the resulting normal field.

Finding g that minimizes equation (7) involves rewriting the minimization problem in terms of $\mathbf{c}$ [24]. Following some straightforward algebraic manipulations (see [24] for details), the coefficient vector can be obtained by taking $\mathbf{c} = (\mathbf{P}^T\mathbf{P} + \lambda_1\mathbf{G} + \lambda_2\mathbf{S})^{-1}\mathbf{P}^T\mathbf{f}$, where $G_{i,j} := \langle \varphi_i, \varphi_j \rangle$, $S_{i,j} := \langle \varphi_i, \varphi_j \rangle_{BL^2}$, and $P_{i,j} := \varphi_j(\mathbf{x}_i)$ as before. Since φ is known in closed form in our test cases, the entries of both $\mathbf{G}$ and $\mathbf{S}$ can be evaluated analytically. The above vector equation can be solved using the Conjugate Gradient method.

IV. Approach

A high level description of our approach is outlined in Figure 3. From the oriented surface samples, we construct the approximation $\vec{V}(\mathbf{x})$ using our variational formulation. We then apply FIR filters to the coefficients of the approximation to construct a scalar field representing the approximate divergence of the smoothed normal field (Figure 3c.) We then use an appropriate discretization of the inverse Laplacian operator to solve the Poisson equation (1), and obtain a smooth function that approximates the original indicator function (Figure 3c). Our code is also freely available for download.

A. Obtaining $\vec{V}(\mathbf{x})$

We define the vector field as $\vec{V}(\mathbf{x}) := \mathbf{L}\,[\upsilon_1(\mathbf{x})\ \upsilon_2(\mathbf{x})\ \upsilon_3(\mathbf{x})]^T$ where $\mathbf{L} := [\mathbf{b_1}\ \mathbf{b_2}\ \mathbf{b_3}]$ is a basis for $\mathbb{R}^3$ that contains the principal lattice directions of the chosen lattice. Each υ_i is a scalar function representing the projection of the smoothed normal field onto each respective principal lattice direction $\mathbf{b}_i$. This may seem unintuitive at first, however, using a non-canonical basis to represent $\vec{V}$

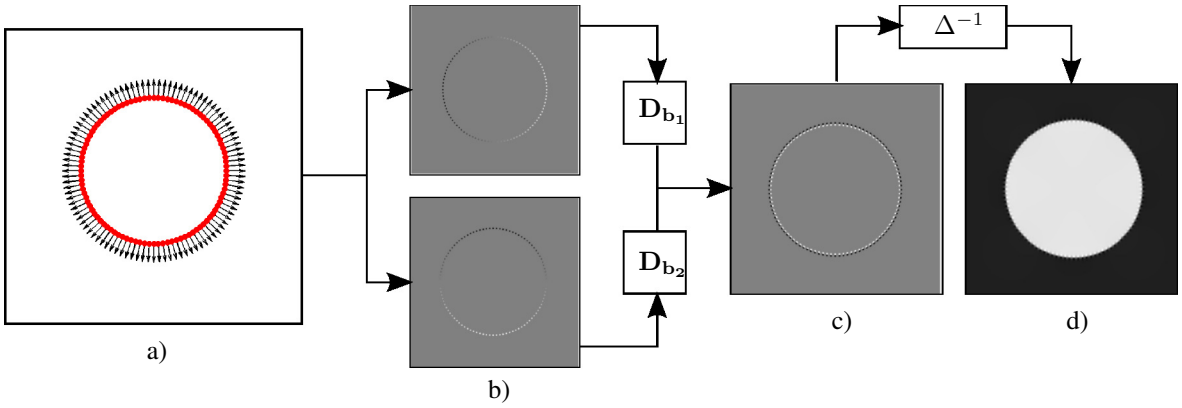


Fig. 3. A high level overview of our surface reconstruction pipeline in 2D. On the far left, a) shows the initial point-set. b) is the result of the smoothed variational reconstruction of the gradient of the indicator, the top image is the component in basis, the bottom is the y component. The approximate divergence of the smoothed indicator function is c), and d) is the resulting indicator function. [[XXX UA: Please fix this.]]

will be convenient in two ways. Firstly, it will allow us to approximate the divergence of $\vec{V}$ by taking directional derivatives of $\vec{V}$ in the principal lattice directions. This is motivated by the fact that it is more natural to approximate derivatives in the principal lattice directions on non-Cartesian lattices. Secondly, it will allow us to construct divergence of the smoothed normal field from more than three principal lattice directions. We also allow each function υ_i to potentially be represented in different function spaces, i.e.

$$\upsilon_i(\mathbf{x}) = \sum_{k=0}^{N} \mathrm{v}_k i \varphi_k^i(\mathbf{x}) \in \mathbb{V}(., \mathcal{L}\varphi^i) \tag{8}$$

The coefficient vector of υ_i is denoted as $\mathbf{v}_i := \begin{bmatrix}\mathrm{v}_1^i & \cdots & \mathrm{v}_N^i\end{bmatrix}$.We use the unfortunate notation $\mathrm{v}_j^i, \varphi_k^i$ to denote that those objects belong to $\upsilon_i(x)$, not to denote exponentiation. We consider each component of $\vec{V}(\mathbf{x})$ separately, thus, it suffices to describe the process at one υ_i. We simply find the υ_i that minimizes

$$\sum_{\forall(\mathbf{x},\mathbf{n})\in P}^{M} (\upsilon_i(\mathbf{x}) - (\mathbf{L}^{-1})_i \cdot \mathbf{n})^2 + E(\upsilon_i). \tag{9}$$

Where $(\mathbf{L}^{-1})_i$ is the i^{th} column of $(\mathbf{L}^{-1})$. As before, this provides us with a parametrized way of controlling the smoothness of the resulting approximation, without the need to explicitly choose a smoothing kernel. These $\mathbf{v}_i$ can be obtained using the procedure of Section III-H.

B. Divergence of $\vec{V}(\mathbf{x})$

A simple way to obtain the divergence of $\vec{V}(\mathbf{x})$ is to consider its analytical partial derivatives which, due to the linearity of the expression, requires one to only take the analytic partial derivatives of the underlying kernel functions. However, this reduces the smoothness of the solution [28], and there is also no conclusive evidence that suggests the analytic derivative of the kernel provides a better approximation to true underlying derivative. With this in mind, we choose to take a discrete filtering approach and approximate the divergence of $\vec{V}$ with a set of discrete FIR filters. While these filters do not strictly facilitate the best approximation scheme for derivatives, they provide a good compromise between compactness versus accuracy, even more so when the reconstruction spaces are shifted [2].

1) Spaces Spanned by a Single Generating Function: When each space is identical (when $\upsilon_i \in \mathbb{V}(\varphi, \mathcal{L}_h)$) the divergence can be found by taking directional derivatives of $\vec{V}$ along the vectors in $\mathbf{L}$ as follows:

$$\begin{aligned}\nabla \cdot \vec{V}(\mathbf{x}) &= \nabla \cdot \mathbf{L}\,[\upsilon_1(\mathbf{x})\ \upsilon_2(\mathbf{x})\ \upsilon_3(\mathbf{x})]^T \\ &= [(\nabla \cdot \mathbf{b_1})\ (\nabla \cdot \mathbf{b_2})\ (\nabla \cdot \mathbf{b_3})] \cdot [\upsilon_1\ \upsilon_2\ \upsilon_3]^T \\ &= \mathbf{D_{b_1}}\upsilon_1 + \mathbf{D_{b_2}}\upsilon_2 + \mathbf{D_{b_3}}\upsilon_3.\end{aligned} \tag{10}$$

This allows us to approximate the divergence of $\vec{V}$ by simply convolving the coefficient vector corresponding to υ_i with an appropriate 1D derivative filter $d[n]$, then sum the coefficients. The expression for the resulting coefficient vector of the divergence is then

$$\mathbf{f} := \sum_{i=1}^{3} \mathbf{v}_i * \mathbf{d}_i \tag{11}$$

where $\mathbf{d}_i$ is a directional derivative filter obtained by orienting the 1D filter d along the direction $\mathbf{b}_i$.

2) Shifted Reconstruction Spaces: To reconstruct within a shifted space, the input point-set is shifted by $\frac{1}{2}\mathbf{b_i}$ then reconstructed within $\mathbb{V}(\varphi, L)$. This is clearly equivalent to reconstructing in a shifted basis space, however, care must be taken to ensure that the data are brought back to a centered representation. Our choice of a shifted derivative filter $s[n]$ accounts for this and brings the resulting approximation back to the unshifted space. Both the shifted and unshifted filters we utilize are found in Table III and are chosen so that they provide a fourth-order discretization of the 1D derivative operator [2].

3) Four Direction Divergence on the BCC Lattice: It is also possible to consider a basis $\mathbf{L}$ in which we choose more than three principal directions to reconstruct the divergence. Thus, we may choose $\mathbf{L} = \mathbf{X_B}$ as the basis for our function. To accommodate for this, we redefine $\vec{V}(\mathbf{x}) := \mathbf{L}\,[\upsilon_1\ \upsilon_2\ \upsilon_3\ \upsilon_4]^T$

Since this matrix is not square, it is not possible to take its inverse in (9). Instead, the least norm inverse $\mathbf{L^T}(\mathbf{L}\mathbf{L}^T)^{-1}$ is used. It is easy to see that, by the same reasoning as in equation (10), $\nabla \cdot \vec{V} = \mathbf{D_{b_1}}\upsilon_1 + \mathbf{D_{b_2}}\upsilon_2 + \mathbf{D_{b_3}}\upsilon_3 + \mathbf{D_{b_4}}\upsilon_4$. Thus, we may use the same idea as in the three direction case, approximating each direction separately, and summing the coefficients to obtain an approximation to the divergence (the shifted case is also similar.)

V. Poisson's Equation

Before detailing how to specifically obtain an indicator function from the divergence of $\vec{V}(\boldsymbol{x})$, we discuss a general framework for solving Poisson's equation on a rectangular domain. We provide an analysis in s-dimensions, as this generalization comes at little cost. Explicitly, we seek an approximation of the function $\chi(\boldsymbol{x})$ that satisfies the Poisson equation with homogeneous Dirichlet boundary conditions, i.e.

$$\begin{aligned} \Delta\,\chi &= f, \;\; \text{in} \;\; \mathcal{C}_o^s, \\ \chi &= 0, \;\; \text{on} \;\; \partial\mathcal{C}^s, \end{aligned} \tag{12}$$

where Δ is the s-dimensional Laplace operator and $\partial\mathcal{C}^s$ denotes the boundary of $\mathcal{C}^s$, i.e. $\partial\mathcal{C}^s \cup \mathcal{C}_o^s = \mathcal{C}^s$.

A. Greens Function

The Poisson equation is an example of a well-studied partial differential equation. When the domain of interest is the unit cube $\mathcal{C}^s$, the solution can be analytically expressed in terms of a Green's function $\mathsf{G}(\boldsymbol{x},\boldsymbol{y})$ that represents the potential due to a point source placed at the location $\boldsymbol{x}$ inside $\mathcal{C}^s$. In other words $\Delta_{\boldsymbol{x}}\,\mathsf{G}(\boldsymbol{x},\boldsymbol{y}) = \delta(\boldsymbol{x}-\boldsymbol{y})$. The Green's function G has the Fourier Sine series expansion

$$\mathsf{G}(\boldsymbol{x},\boldsymbol{y}) = \sum_{\boldsymbol{m}\in\mathbb{Z}_+^s} \frac{\prod_{i=1}^s \sin(m_i\pi x_i)\sin(m_i\pi y_i)}{-\pi^2\|\boldsymbol{m}\|^2}, \tag{13}$$

and the solution χ is given by

$$\chi(\boldsymbol{x}) = \int_{\mathcal{C}^s} f(\boldsymbol{y})\mathsf{G}(\boldsymbol{x},\boldsymbol{y})d\boldsymbol{y}. \tag{14}$$

B. Fourier Interpretation

Even though there has been much effort on finding alternate rapidly convergent series representations of the Green's function (see e.g. Marshall [**?**]), it turns out that the form of the Green's function given in (13), is ideally suited for our needs as it allows us to easily express the solution in the Fourier domain in terms of the Fourier coefficients of f.

Suppose we are given an $L_2(\mathcal{C}^s)$ function $\rho(\boldsymbol{x})$ that is defined on the open unit cube $\mathcal{C}_o^s$. In order to obtain a Fourier sine series that converges to ρ almost everywhere in $\mathcal{C}_o^s$, we need to extend ρ periodically such that it is odd with respect to each variable. Particularly, let us extend the domain of ρ so that, within the interval $\mathcal{T}^s$, it satisfies

$$\rho(\boldsymbol{x}) := \begin{cases} \big(\prod_{i=1}^s \operatorname{sgn}(x_i)\big)\rho(|x_1|,\ldots,|x_s|) & \text{if } \forall i\ |x_i| \neq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{15}$$

while outside $\mathcal{T}^s$, it is $\mathcal{T}^s$-periodic, i.e. $\rho(\boldsymbol{x}+2\boldsymbol{k}) = \rho(\boldsymbol{x})$ for $\boldsymbol{k}\in\mathbb{Z}^s$. The extended function ρ can be developed into a multidimensional Fourier sine series, the coefficients of which are given by

$$\tilde{\rho}[\boldsymbol{m}] = 2^s\big\langle \rho(\boldsymbol{x}), \textstyle\prod_{i=1}^s \sin(m_i\pi x_i)\big\rangle_{\mathcal{T}^s}, \text{ for } \boldsymbol{m}\in\mathbb{Z}_+^s. \tag{16}$$

The coefficient sequence $\tilde{\rho}[\boldsymbol{m}]$ can also be seen as a special case of the more general multidimensional Fourier series. In fact, with an odd extension of the sequence $\tilde{\rho}$, the function ρ can be expressed as a Fourier series. In particular, if we define the Fourier series coefficients $\hat{\rho}[\boldsymbol{m}]$ (for $\boldsymbol{m}\in\mathbb{Z}^s$) as

$$\hat{\rho}[\boldsymbol{m}] := \begin{cases} \frac{1}{(2\imath)^s}\prod_i \operatorname{sgn}(m_i)\tilde{\rho}[|m_1|,\ldots,|m_s|] & \text{if } \forall i\ |m_i| \neq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{17}$$

then ρ is also given by the Fourier series $\rho(\boldsymbol{x}) = \sum_{\boldsymbol{m}\in\mathbb{Z}^s}\hat{\rho}[\boldsymbol{m}]\exp(\imath\pi\boldsymbol{m}\cdot\boldsymbol{x})$. Thus, we use the notation $\tilde{\rho}[\cdot]$ and $\hat{\rho}[\cdot]$ to distinguish between the Fourier sine series and Fourier series coefficients of the periodic function ρ.

Since the Green's function G is defined in $\mathcal{C}_o^s$ and has a Fourier sine series representation, it is natural to seek a solution that can be developed into a Fourier sine series as well. Consequently, for the remainder of this section, we shall only be dealing with Fourier sine series. Therefore, it suffices to consider only the coefficients in $\mathbb{Z}_+^s$.

Using the analytic solution (14) and the sine series representation of the Green's function (13), it is easy to show that the solution is given by

$$\tilde{V}[\boldsymbol{m}] = -\frac{\tilde{f}[\boldsymbol{m}]}{\pi^2\|\boldsymbol{m}\|^2}, \tag{18}$$

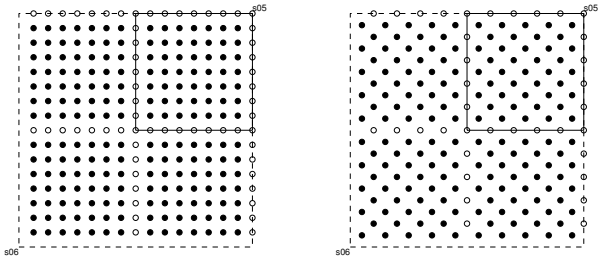
where $\boldsymbol{m}\in\mathbb{Z}_+^s$ and $\tilde{f}[\cdot]$ represents the Fourier sine series coefficients of the odd extension of f. To facilitate subsequent discussions, let us introduce the solution operator $\Delta^{-1} \Leftrightarrow \frac{1}{-\pi^2\|\boldsymbol{m}\|^2}$ ($\boldsymbol{m}\in\mathbb{Z}_+^s$), where the symbol $\Leftrightarrow$ represents how the Fourier sine series coefficients are affected by the operator. The self-adjoint operator Δ^{-1} is the inverse of the Laplace operator. Self-adjointness can be easily verified with the aid of Parseval's relation, which states that $\langle a,b\rangle_{\mathcal{T}^s} = \sum_{\boldsymbol{m}\in\mathbb{Z}_+^s}\tilde{a}[\boldsymbol{m}]\tilde{b}[\boldsymbol{m}]$ for any $\mathcal{T}^s$-periodic functions a and b that are in $L_2(\mathcal{T}^s)$ and odd.

C. Approximate Solution

We are interested in the scenario where the function f is only known through its point samples. Specifically, we assume that the samples of f reside on the sites of the lattice $\mathcal{L}_h$. For the purpose of enumerating the samples that are contained within $\mathcal{T}^s$, let us define the point set

$$\mathcal{T}_h := \underbrace{\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{2^sN}\}}_{\mathcal{I}_h} \cup \underbrace{\{\boldsymbol{x}_{2^sN+1},\ldots,\boldsymbol{x}_{2^sN+M}\}}_{\mathcal{B}_h}, \tag{19}$$

where $\mathcal{I}_h := \mathcal{T}_o^s \cap \{\boldsymbol{x}+\boldsymbol{m} : \boldsymbol{x}\in\mathcal{L}_h\cap\mathcal{C}_o^s, \boldsymbol{m}\in\mathbb{Z}^s\}$ consists of interior lattice points, while $\mathcal{B}_h$ consists of extended boundary points, i.e. $\mathcal{B}_h := (\mathcal{L}_h\cap\partial\mathcal{T}^s\cap(-1,1]^s)\cup(\mathcal{L}_h\cap\mathcal{T}_o^s\backslash\mathcal{I}_h)$. Two dimensional illustrations are shown in Figure 4. We



(a) Cartesian (b) Quincunx

Fig. 4. Illustration of the point set $\mathcal{T}_h$ on the two dimensional Cartesian ($\mathbf{L} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $h = \frac{1}{8}$) and Quincunx ($\mathbf{L} = \left(\begin{smallmatrix} 1 & -1 \\ 1 & 1 \end{smallmatrix}\right)$, $h = \frac{1}{10}$) lattices. The interior points ($\bullet$) belong to the set $\mathcal{I}_h$ while the extended boundary points ($\circ$) belong to $\mathcal{B}_h$.

note that with the above definition of $\mathcal{I}_h$, the points $\boldsymbol{x}_j$ for $j \in \{1, \ldots, N\}$ are contained in the open unit cube $\mathcal{C}_o^s$ whereas the remaining points ($\boldsymbol{x}_j$ for $j \in \{N+1, \ldots, 2^s N\}$) lie outside. Furthermore, we assume that the lattice $\mathcal{L}$ and the sampling rate h are such that the set $\{\boldsymbol{x} + 2\boldsymbol{m} : \boldsymbol{x} \in \mathcal{T}_h, \boldsymbol{m} \in \mathbb{Z}^s\} = \mathcal{L}_h$. We remark that this requirement for $\mathcal{L}_h$ is also satisfied by integration lattices that are commonly used to devise quadrature rules within the unit cube $\mathcal{C}^s$ [**?**].

We denote the interior samples of f as $f[\boldsymbol{x}_j] := f(\boldsymbol{x}_j)$ where $\boldsymbol{x}_j \in \mathcal{I}_h$. Note that $f[\boldsymbol{x}_j]$ only needs to be known in $\mathcal{C}_o^s$ ($j \in \{1, \ldots, N\}$), the other samples can be inferred from oddity.

We wish to use the samples of f to seek an approximation V_{app} of the function V that solves the Poisson equation (12). Since V is a periodic function, we follow the recipe of Jacob *et al.* [**?**] and seek an approximation that lies in a space generated by a periodic reconstruction function. Specifically, we are interested in the case where V_{app} lies in the space $\mathbb{V}(\varphi_p, \mathcal{L}_h) := \mathrm{span}_{\boldsymbol{x}_j \in \mathcal{T}_h}\{\varphi_p(\frac{\boldsymbol{x}-\boldsymbol{x}_j}{h})\}$ that is spanned by the scaled and translated versions of a periodic function φ_p. Our sought-after approximation is given by

$$V_{\mathsf{app}}(\boldsymbol{x}) = \sum_{\boldsymbol{x}_j \in \mathcal{T}_h} c[\boldsymbol{x}_j] \varphi_p\big(\frac{\boldsymbol{x} - \boldsymbol{x}_j}{h}\big), \tag{20}$$

where $c[\boldsymbol{x}_j]$ is an unknown coefficient sequence defined on the point set $\mathcal{T}_h$ that is to be determined from the samples of f. The function φ_p is a periodized version of a generating function φ and is defined as

$$\varphi_p(\boldsymbol{x}) := \sum_{\boldsymbol{m} \in \mathbb{Z}^s} \varphi\big(\boldsymbol{x} - \frac{2}{h}\boldsymbol{m}\big). \tag{21}$$

It is easy to verify that, with this definition of φ_p, V_{app} is also $\mathcal{T}^s$-periodic. Even though we are only interested in the behavior of V_{app} inside $\mathcal{C}^s$, extending V_{app} to the entire domain $\mathcal{T}^s$ using (20) allows us to use the Fourier domain error kernel proposed by Jacob *et. al* [**?**] to quantify the approximation error $\|V - V_{\mathsf{app}}\|_{L_2(\mathcal{T}^s)}$.

Another simplification is in order here. Since the solution V to the Poisson problem (12) is odd with respect to each variable, we look for an approximate solution $V_{\mathsf{app}} \in \mathbb{V}(\varphi_p, \mathcal{L}_h)$ that is also odd in each variable. This can be achieved by setting the boundary coefficients to zero and by requiring that the resulting sequence be odd. With $c[\boldsymbol{x}_j] = 0$ for $\boldsymbol{x}_j \in \mathcal{B}_h$, the approximation (20) simplifies to

$$V_{\mathsf{app}}(\boldsymbol{x}) = \sum_{j=1}^{2^s N} c[\boldsymbol{x}_j] \varphi_p(\frac{\boldsymbol{x} - \boldsymbol{x}_j}{h}). \tag{22}$$

If the generator φ is even and the sequence $c[\cdot]$ is odd, the resulting approximation V_{app} is odd and satifies zero Dirichlet boundary conditions (see Appendix **??**). With this simplification, the coefficient sequence $c[\boldsymbol{x}_j]$ only needs to be determined for $j \in \{1, \ldots, N\}$. The rest can be inferred from oddity.

D. Error Analysis

1) Error Kernel for Periodic Functions: In many approximation problems involving periodic functions, one is interested in seeking an approximation z_{app} of a periodic function z from its measurements that lie on some sampling lattice. In light of the notations introduced earlier, suppose that $z \in L_2(\mathcal{T}^s)$ and is also $\mathcal{T}^s$-periodic. Additionally, suppose that the measurements are made at the locations $\boldsymbol{x}_j \in \mathcal{T}_h$ so that one can obtain an approximation z_{app} that belongs to $\mathbb{V}(\varphi_p, \mathcal{L}_h)$ and — in a manner similar to the expansion (20) — is given by $z_{\mathsf{app}}(\boldsymbol{x}) = \sum_{\boldsymbol{x}_j \in \mathcal{T}_h} \zeta[\boldsymbol{x}_j] \varphi_p(\frac{\boldsymbol{x}-\boldsymbol{x}_j}{h})$, where the coefficient sequence $\zeta[\cdot]$ is obtained through the discrete measurements

$$\zeta[\boldsymbol{x}_j] = \big\langle z(\boldsymbol{x}), \bar{\varphi}_p(\frac{\boldsymbol{x} - \boldsymbol{x}_j}{h})\big\rangle_{\mathcal{T}^s} \tag{23}$$

made with the scaled and shifted versions of a periodic analysis function $\bar{\varphi}_p(\boldsymbol{x})$ (the periodization operation is defined by (21)). The main result of Jacob *et al.* [**?**] states that the mean square approximation error $\|z - z_{\mathsf{app}}\|_{L_2(\mathcal{T}^s)}$ at scale h can be predicted according to

$$\sqrt{\sum_{\boldsymbol{m} \in \mathbb{Z}^s} \|\hat{z}[\boldsymbol{m}]\|^2 E(\frac{h}{2}\boldsymbol{m})} \tag{24}$$

where $\hat{z}[\boldsymbol{m}]$ are the Fourier series coefficients of z and $E(\cdot)$ is the error kernel of Blu and Unser [**?**] introduced earlier (4). Remarkably, the same error kernel can be used to predict the error for periodic and non-periodic functions alike. The only change that needs to be made is in the prediction equation. For functions in $L_2(\mathbb{R}^s)$, the error is predicted according to the integral in (5). On the other hand, for periodic functions that belong to $L_2(\mathcal{T}^s)$, the prediction equation turns into the summation given in (24). We refer the reader to Jacob *et al.* [**?**] for details.

2) Extension to Linear Operators: We are interested in approximating from the discrete measurements of z, not the function z itself but a function Γz that is obtained by applying the linear operator Γ to z. We would like to extend the error kernel formulation (4) so that it will allow us to quantify the error incurred in approximating Γz from the measurements of z. Towards this end, we assume that the approximation $(\Gamma z)_{\mathsf{app}}$ lies in a shift-invariant space spanned by the generator

φ_p, where the coefficients are obtained from the measurements through a digital filtering operation. Particularly, we seek an approximation that is given by

$$(\Gamma z)_{\mathsf{app}}(\boldsymbol{x}) = C_h \sum_{\boldsymbol{x}_j \in \mathcal{T}_h} (\zeta \circledast \gamma_h)[\boldsymbol{x}_j] \varphi_p(\frac{\boldsymbol{x}-\boldsymbol{x}_j}{h}), \qquad (25)$$

where ζ denotes the discrete measurements of z obtained through (23), γ is a digital filter defined on the lattice $\mathcal{L}$ and γ_h is its corresponding scaled version, i.e. $\gamma_h[\boldsymbol{x}_j] := \gamma[\frac{\boldsymbol{x}_j}{h}]$ where $\frac{\boldsymbol{x}_j}{h} \in \mathcal{L}$, and C_h is an associated scaling constant. The digital filter γ represents a suitable discretization of the operator Γ on the lattice $\mathcal{L}$. It is to be applied to the measurements through a cyclic convolution operation (denoted by $\circledast$) on the lattice $\mathcal{L}_h$ using a periodized version of γ_h. In particular, the convolution operation in (25) is defined as

$$(\zeta \circledast \gamma_h)[\boldsymbol{x}_j] := \sum_{\boldsymbol{x}_k \in \mathcal{T}_h} \big(\sum_{\boldsymbol{m} \in \mathbb{Z}^s} \gamma[\frac{\boldsymbol{x}_k + 2\boldsymbol{m}}{h}] \big) \zeta[\boldsymbol{x}_j - \boldsymbol{x}_k], \quad (26)$$

where $\boldsymbol{x}_j \in \mathcal{L}_h$. In the above definition, $\zeta[\cdot]$ is assumed to be $\mathcal{T}^s$-periodic. Furthermore, the resulting sequence $(\zeta \circledast \gamma_h)[\cdot]$ is also $\mathcal{T}^s$-periodic.

Let $\hat{\Gamma}(\boldsymbol{\omega})$ be the Fourier transform of the operator Γ, and $\hat{G}(\boldsymbol{\omega})$ be the DTFT of the filter γ. Furthermore, let Γ be bounded so that $\Gamma z \in L_2(\mathcal{T}^s)$. A simple modification of the measurement process (23) yields the desired extension of the error kernel formulation (4). Our main result can be summarised as follows.

Theorem 1: Suppose that Γ is self-adjoint and shift-invariant. The Fourier error kernel that predicts the approximation error $\|(\Gamma z)_{\mathsf{app}} - \Gamma z\|_{L_2(\mathcal{T}^s)}$ is given by $E(\boldsymbol{\omega}) := E_{\mathsf{min}}(\boldsymbol{\omega}) + E_{\mathsf{mod}}(\boldsymbol{\omega})$, where the minimum error kernel $E_{\mathsf{min}}(\boldsymbol{\omega})$ is given in (4), while the modified residue error kernel is given by

$$E_{\mathsf{mod}}(\boldsymbol{\omega}) = \hat{A}_{\varphi}(\boldsymbol{\omega}) \big| \frac{\hat{\mathring{\varphi}}(\boldsymbol{\omega}) \hat{G}(\boldsymbol{\omega})}{\hat{\Gamma}(\boldsymbol{\omega})} - \hat{\mathring{\varphi}}(\boldsymbol{\omega}) \big|^2. \qquad (27)$$

The proof is given in Appendix **??**.

Note that, even though the operator Γ acts in the space $L_2(\mathcal{T}^s)$, it is extended to the more general space $L_2(\mathbb{R}^s)$ in this error kernel formulation. Therefore, (27) can be used to quantify the L_2 error in both $L_2(\mathbb{R}^s)$ and $L_2(\mathcal{T}^s)$. Going back to the original problem (12), Theorem 1 gives us a way to design and analyze digital filtering solutions that approximate the analytic solution (18). In particular, the linear operator that we wish to discretize is Δ^{-1}. In the sequel, we show how to use the modified residue error kernel (27) to design filtering schemes that discretize this operator and can be efficiently implemented in the Fourier domain.

In order to characterize the order of accuracy provided by a periodic generating function φ_p, we shall switch to the more general space $L_2(\mathbb{R}^s)$. The link between the approximation properties of φ and those of its periodized version φ_p is established by the error kernel formulation introduced in Section V-D1. Since the same kernel can be used in both cases, henceforth, we shall reason about the approximation properties of the generator φ as the same properties are also applicable to its periodized counterpart φ_p.

In many approximation scenarios, one typically assumes a Dirac point sampling model, i.e. $\bar{\varphi}(\boldsymbol{x}) = \delta(\boldsymbol{x})$. This prevents the direct realization of the minimum approximation scenario. Furthermore, for many functions encountered in practice, the power spectrum is concentrated around $\boldsymbol{\omega} = 0$. For such scenarios, one speaks of an *asymptotically optimal* k-th order approximation scheme if the L_2 error behaves as $O(h^k)$ where $k = o(\mathcal{L}, \varphi)$. Our operator discretization approach is also based on these assumptions. Provided that $V \in W_2^k(\mathcal{T}^s)$, our goal is to design suitable digital filters that yield V_{app} such that $\|V - V_{\mathsf{app}}\|_{L_2(\mathcal{T}^s)} = O(h^k)$. As discussed earlier in Section **??**, the criterion that needs to be satisfied is $E_{\mathsf{mod}}(\boldsymbol{\omega}) = O(\|\boldsymbol{\omega}\|^{2k})$.

E. Relationship to the Galerkin method

The modified residue error kernel (27) can also be analyzed in light of the Galerkin method [**?**] using a weak formulation of the Poisson equation (12), where the trial and test spaces are the same with the notable difference that the spaces are not required to explicitly satisfy any particular boundary conditions. Rather, a zero boundary condition is implicitly obtained by requiring that the solution coefficients $c[\cdot]$ be odd as explained in Section **??**. Here, we establish the connection for the homogeneous case but the analysis also easily extends to the non-homogeneous case, as well as to other types of differential operators.

In its weak form, the solution $V \in L_2(\mathcal{T}^s)$ to the homogeneous Poisson equation (12) satisfies the weak formulation

$$\big\langle V, u \big\rangle_{\mathcal{T}^s} = \mathcal{F}(u), \ \ \forall u \in L_2(\mathcal{T}^s), \qquad (28)$$

where the functions u are suitable test functions. The functional $\mathcal{F}(\cdot)$ is defined as $\mathcal{F}(u) := \big\langle \Delta^{-1} f, u \big\rangle_{\mathcal{T}^s}$, where f is the odd extension of the function that appears on the right hand side of (12), and the operator $\Delta^{-1} \Leftrightarrow \frac{1}{-\pi^2 \|\boldsymbol{m}\|^2}$ has the interpretation of the inverse Laplacian as introduced in (18). We remark that this formulation is slightly stronger than the usual weak formulations based on the Laplacian Δ where the trial and test spaces coincide with the Sobolev space $W_2^1(\mathcal{T}^s)$. In comparison, here the trial and test spaces coincide with the more general space $L_2(\mathcal{T}^s)$. This formulation is also in direct correspondence with our earlier treatment of the problem.

Let us now restrict attention to the finite dimensional space $\mathbb{V}(\varphi_p, \mathcal{L}_h) \subset L_2(\mathcal{T}^s)$. We seek a weak solution $V_{\mathsf{app}} \in \mathbb{V}(\varphi_p, \mathcal{L}_h)$ so that

$$\big\langle V_{\mathsf{app}}, u_h \big\rangle_{\mathcal{T}^s} = \mathcal{F}(u_h), \ \ \forall u_h \in \mathbb{V}(\varphi_p, \mathcal{L}_h), \qquad (29)$$

and the Galerkin residual satifies the orthogonality condition

$$\big\langle V - V_{\mathsf{app}}, u_h \big\rangle_{\mathcal{T}^s} = 0, \ \ \forall u_h \in \mathbb{V}(\varphi_p, \mathcal{L}_h). \qquad (30)$$

From this, it can be deduced that the minimum error approximation V_{app} is the orthogonal projection of V upon $\mathbb{V}(\varphi_p, \mathcal{L}_h)$. The coefficients of the approximation are thus given by

$$\begin{aligned} c[\boldsymbol{x}_j] &= \mathcal{F}(h^{-s} \mathring{\varphi}(\frac{\boldsymbol{x}-\boldsymbol{x}_j}{h})) \\ &= \big\langle f(\boldsymbol{x}), \Delta^{-1}(h^{-s} \mathring{\varphi}(\frac{\boldsymbol{x}-\boldsymbol{x}_j}{h})) \big\rangle_{\mathcal{T}^s}. \end{aligned} \qquad (31)$$

Observe that if we use the analysis function $\bar{\varphi}_p = \Delta^{-1}\,\mathring{\varphi}_p$ (in a distributional sense) in the measurement equation (23), and subsequently employ no filtering (i.e. $\gamma[\boldsymbol{x}_j] = \delta[\boldsymbol{x}_j]$ in (25)), then the measurements obtained will realize the orthogonal projection of V upon $\mathbb{V}(\varphi_p, \mathcal{L}_h)$. In this case, the modified Fourier residue error kernel (27) vanishes, i.e. $E_{\mathsf{mod}}(\boldsymbol{\omega}) = 0$.

On the other hand, if the point samples of f are already available (i.e. $\bar{\varphi} = \delta$) and the orthogonal projection cannot be realized, then the goal of the asymptotically optimal procedure is to find a suitable correction filter γ such that $E_{\mathsf{mod}}(\boldsymbol{\omega}) = O(\|\boldsymbol{\omega}\|^{2k})$ where $k = o(\mathcal{L}, \varphi)$. In other words, our approximation procedure attempts to reproduce the Galerkin orthogonality condition given by (30). This is akin to the quasi-interpolation condition (**??**) introduced in Section **??**.

VI. Recovering the Indicator

Returning to the surface reconstruction approach, once the divergence of the smoothed vector field has been computed, the remaining step is to find a function that satisfies equation (1). As with the computation of a function's derivative, we also seek a filtering solution for the inverse of the Laplacian. We desire a filter $\mathbf{l}^{-1}[\mathbf{m}]$ that provides a fourth-order discretization of the inverse Laplacian operator Δ^{-1}. This can be achieved by first obtaining a 1D filter l that provides a fourth-order discretization of the 1D second-derivative operator. On the Cartesian and BCC lattices, application of this filter along the principal directions followed by a summation of the resulting coefficients, yields a fourth-order discretization of the Laplacian [**?**]. The inverse Laplacian filter $\mathbf{l}^{-1}$ is simply the inverse of the resulting Laplacian filter. Our approximation $\tilde{\chi}(\mathbf{x})$ can now be written as

$$\tilde{\chi}(\mathbf{x}) = \sum_{\mathbf{k}} (\mathbf{f} * \mathbf{l}^{-1})_k \varphi_k(\mathbf{x}) \tag{32}$$

where $\mathbf{f}$ is the coefficient vector obtained in the previous step (cf. equation (11)). The only stipulations to this is that the indicator function must satisfy zero Dirichlet boundary conditions (which is given as part of the problem definition). In practice, the filter $\mathbf{l}^{-1}$ does not need to be computed explicitly as it can be efficiently applied in the Fourier domain via a Discrete Sine Transform (DST) that only depends on the 1D filter l [**?**, Chapter 4]. Whereas the DST on the Cartesian lattice is well-defined, on the BCC lattice, we utilize a modified DST that can be efficiently implemented via 1D DSTs [**?**].

A. Extracting the Iso-Contour

With a true indicator function, the level-set desired is the set of points $\mathbf{x}$ such that $\tilde{\chi}(\mathbf{x}) = \frac{1}{2}$, which can readily be visualized either via marching algorithms, ray tracing, or other volumetric visualization techniques. However, the constant scaling factor of equation (2) offsets the appropriate level set at which to threshold the indicator function. A common approach that has shown to be quite robust in this situation [12], is to average the levelset obtained by evaluating the approximation at the input points $1/|P| \sum_{(\mathbf{x},\mathbf{n})\in P} \tilde{\chi}(\mathbf{x})$. The reasoning for this, is that the iso-value at the input data should be on, or at least near the surface of the model. Once the iso-surface has been determined, the mesh is extracted via marching cubes, with the step density set to half the lattice scaling factor ($\frac{h}{2}$)

VII. Experiments

To test our methodology, we choose comparable basis functions on both the Cartesian and BCC lattices. These functions are comparable in that they both provide the same approximation order and smoothness.

A. Cartesian Lattice

1) Basis function: Choosing $\mathbf{X} = \mathbf{X_{C2}} := [\mathbf{I_3}\ \mathbf{I_3}]$ (the 3×6 matrix created by the union of two identity matrices) generates a triliniear piecewise polynomial. It is easy to see, by definition, when $\mathbf{X_{C2}}$ is used in conjunction with Equation 6 on page 5, the resulting function $M_{\mathbf{X_{C2}}}(\mathbf{x})$ corresponds to the non-centered tensor product extension of the linear B-spline in three dimensions. Likewise, the non-centered cubic tensor product box spline is given by $M_{\mathbf{X_{C4}}}(\mathbf{x})$, where $\mathbf{X_{C4}} := [\mathbf{X_{C2}}\ \mathbf{X_{C2}}]$. A box spline is easily centered by taking $M_{\mathbf{X}}(\mathbf{x} + \mathbf{c_X})$, with $\mathbf{c_X} := \sum_{\xi\in\mathbf{X}} \frac{1}{2}\xi$. We will assume, unless otherwise stated, that $M_{\mathbf{X}}$ defines the centered box spline generated by $\mathbf{X}$. Due to the smoothness requirements of the optimization constraint, we only considered the spline $M_{\mathbf{X}_{C4}}$.

2) Filter Design: Extending the filter in table 1

B. BCC Lattice

1) Basis function: On the BCC lattice, a choice of

$$\mathbf{X_B} := \begin{bmatrix} -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix}$$

generates a piecewise linear polynomial function [1]. The support of this box spline is a rhombic dodecahedron, which incidentally contains the set of first nearest neighbors to the Voronoi region of the lattice at the origin, and is a natural fit for the lattice. The quintic box spline is given by the matrix $\mathbf{X_{B2}} := [\mathbf{X_B}\ \mathbf{X_B}]$, or may also be seen as the convolution of the linear box spline with itself. However, there exist both an efficient piecewise polynomial representation [1], and an efficient implementation [29] of these piecewise quintic polynomial functions. Again, due to the smoothness requirements of the optimization constraint, we only considered the spline $M_{\mathbf{X}_{B2}}$.

2) Filter Design: We now turn to the problem of designing filters that discretize the operator $\Delta^{-1} \leftrightarrow (-4\pi^2(\omega_1^2+\omega_2^2))^{-1}$, and are to be used in conjunction with the tensor-product B-splines. Recall that the space $\mathbb{V}(B^k, \mathbb{Z}_h^2)$ generated by $B^k(x_1, x_2) = \beta^k(x_1)\beta^k(x_2)$ (where β^k is as defined in (**??**)), satisfies the approximation property: $o(\mathbb{Z}^2, B^k) = k + 1$. This result also extends to the Cartesian lattice in higher dimensions. Consequently, we cover the 2D case in detail here. The extension to 3D is left to the reader.

a) Interpolation: For the generator B^k, we are interested in a $(k+1)$-th order asymptotically optimal discretization of the inverse Laplacian Δ^{-1}. According to Proposition **??**, we need the two filters p_{B^k} and λ.

Since B^k is a compact generator, p_{B^k} will also be a compact filter. Furthermore, since B^k is separable, p_{B^k} can be determined by sampling $\beta^k(x)$ to yield the filter $p_{\beta^k}[n] = \beta^k(n)$ where $n \in \mathbb{Z}$. The weights of p_{B^k} are then obtained by a simple tensor product, i.e. $p_{B^k}[n_1, n_2] = p_{\beta^k}[n_1]p_{\beta^k}[n_2] \leftrightarrow \hat{P}_{B^k}(\omega_1, \omega_2) = \hat{P}_{\beta^k}(\omega_1)\hat{P}_{\beta^k}(\omega_2)$.

As for the filter λ that discretizes Δ, we follow a similar approach and first look for a symmetric and compact 1D filter λ_1 that satisfies the 1D analog of (**??**), i.e. $\hat{\Lambda}_1(\omega) = -4\pi^2\omega^2 + O(\omega^{k+3})$. The unknown filter weights are determined by requiring that the Taylor developments of $(\hat{\Lambda}_1(\omega) + 4\pi^2\omega^2)$ be zero up to ω^{k+3}. The filter $\lambda \leftrightarrow \hat{\Lambda}$ is then given by a simple addition, i.e. $\hat{\Lambda}(\omega_1, \omega_2) = \hat{\Lambda}_1(\omega_1) + \hat{\Lambda}_1(\omega_2)$.

The sizes of p_{β^k} and λ_1 obviously depend on the degree k. The filters used in our experiments are tabulated in Table III.

b) Quasi-Interpolation: For the quasi-interpolative model, we need the additional filter q (**??**). Choosing ψ to be the tensor product B-spline B^l where $l > k$, the weights of q are given by the tensor product of the 1D filter $q_1[n] := (\beta^l * \mathring{\beta}^k)(n)$ with itself. Using the recursive definition of the 1D B-splines, q_1 can be further split as $q_1[n] = (w_1 * a^{-1}_{\beta^k})[n]$ where $w_1[n] := \beta^{k+l+1}(n)$ and $a^{-1}_{\beta^k}[\cdot] \leftrightarrow \frac{1}{\hat{A}_{\beta^k}(\cdot)}$. Example filter weights for the case $l = 5$ are provided in Table III.

3) BCC Lattice: It is convenient to introduce a scaled version of the BCC lattice $\mathcal{H}$ generated by the matrix $\mathbf{H} = \sqrt[3]{4}\mathbf{B}$. With respect to $\mathcal{H}$, we denote the order-k ($k \in 2\mathbb{Z}_+$) rhombic-dodecahedral box spline as $\vartheta^k(\boldsymbol{x}) := M_{\boldsymbol{\Theta}^{k/2}}(\boldsymbol{x}/\sqrt[3]{4})$. Note that k now refers to the approximation order of the box-spline rather than its polynomial degree.

a) Interpolation: The operator we wish to discretize is $\Delta^{-1} \leftrightarrow -4\pi^2(\omega_1^2+\omega_2^2+\omega_3^2)^{-1}$. According to Proposition **??**, we need the combined filter $(p_{\vartheta^k} * \lambda)$ where $\lambda \leftrightarrow \hat{\Lambda}$ satisfies (**??**), i.e. $\hat{\Lambda}(\boldsymbol{\omega}) = -4\pi^2\|\boldsymbol{\omega}\|^2 + O(\|\boldsymbol{\omega}\|^{k+2})$. The compact filter p_{ϑ^k} is non-separable and is determined by the samples of ϑ^k, i.e. $p_{\vartheta^k}[\boldsymbol{x}_j] = \vartheta^k(\boldsymbol{x}_j)$ where $\boldsymbol{x}_j \in \mathcal{H}$.

As for the filter λ, observe that

$$\|\boldsymbol{\omega}\|^2 = \frac{1}{4}\sum_{i=1}^{4}(\boldsymbol{\theta}_i \cdot \boldsymbol{\omega})^2, \tag{33}$$

where the (scaled) box spline direction vectors $\boldsymbol{\theta}_i$ ($i \in \{1,2,3,4\}$) correspond to the column vectors $(1,1,-1)^\mathsf{T}$, $(1,-1,1)^\mathsf{T}$, $(-1,1,1)^\mathsf{T}$ and $(-1,-1,-1)^\mathsf{T}$ respectively. Since $\boldsymbol{\theta}_i \in \mathcal{H}$, we can re-use the 1D filter $\lambda_1[n]$ from the previous section and apply it across the directions $\boldsymbol{\theta}_i$ to yield the filter λ. Therefore, $\lambda \leftrightarrow \hat{\Lambda}$ is given by

$$\hat{\Lambda}(\boldsymbol{\omega}) = \frac{1}{4}\sum_{i=1}^{4}\hat{\Lambda}_1(\boldsymbol{\theta}_i \cdot \boldsymbol{\omega}), \tag{34}$$

where the 1D filter $\lambda_1 \leftrightarrow \hat{\Lambda}_1$ satisfies $\hat{\Lambda}_1(\omega) = -4\pi^2\omega^2 + O(\omega^{k+2})$.

Table II summarizes the various filters that are the BCC analogs of Cartesian filters listed in Table III.

C. Shifted Basis Functions

We also consider shifted box splines $M^{\xi_i}_{\mathbf{X}}(\mathbf{x}) := M_{\mathbf{X}}(\mathbf{x} - \frac{1}{2}\xi_i)$, where ξ_i is any direction vector in $\mathbf{X}$. In the univariate case with linear B-splines, it has been shown that introducing a shift can offer an improvement when representing scalar data [30]. More generally and relevant to our purposes, they have also been shown to reduce error when approximating the gradient of scalar fields [2].

VIII. Results and Discussion

For our tests, we restrict ourselves to the function spaces spanned by the Cubic tensor product basis on the Cartesian lattice, and the Quintic basis function on the BCC lattice. To provide a baseline reconstruction with which to compare our results, we consider FFT, Poisson and Screened Poisson Methods. We utilize the test models and point-sets of the Anchor, Daratech, Dancing Children, Gargoyle, and Quasimoto models provided by the Surface Reconstruction Benchmark [15].

Quantitatively, we evaluate whether function spaces on the BCC lattice provide better approximation spaces than those on the Cartesian lattice. In order to reduce any artifacts that may arise from any noise or the non-uniformity of the data, we utilize the ideal reference point-sets of our test models. We measure both the Hausdorff distance, mean distance, and angle deviation from the original shape with the tools provided by Berger et al. [15]. We evaluate the FFT, Poisson, and Screened Poisson surface reconstruction algorithms, as their theory is close to ours. We choose λ_1 and λ_2 so that the reconstructions are comparable to the Screened Poisson results.

Since the Gargoyle model has many interesting high frequency features, we consider reconstructing it at increasing grid resolutions. For the Poisson and Screened Poisson methods, we restrict the depth of the tree so that the finest resolution provided by the oct-tree is equivalent to that of the regular Cartesian, Figure 5. We also seem to be able to improve upon the Screened Poisson method, however, the parameter space needs to be explored further to conclusively assert this. Regardless, it is still a good baseline with which to compare our results.

We fix the resolution of the grid at 128^3, and reconstruct the remaining models (Figure 6.) We choose this resolution as it seems to be a point of contention between the cases with the Gargoyle model. However, there seems to be a consistent advantage in using the BCC lattice over the Cartesian lattice. Moreover, in both cases, we noticed that introducing a shift tends to reduce all error metrics. Unfortunately, reconstructing the divergence of the smoothed vector field from more than three lattice directions produced little difference. Figure 7 shows a visual comparison of the techniques. We manage to outperform the FFT and Poisson techniques by a large margin, and outperform the screened Poisson case.

TABLE I. THE VARIOUS 1D FILTERS TO BE USED IN CONJUNCTION WITH BILINEAR AND BICUBIC APPROXIMATION.

	$\mathbb{V}(B^1,\mathbb{Z}_h^2)$ Interp.	$\mathbb{V}(B^3,\mathbb{Z}_h^2)$ Interp.	$\mathbb{V}(B^3,\mathbb{Z}_h^2)$ Quasi. ($l=5$)
$p_{\beta^k}[n]$	δ_n	$[\frac{1}{6}\ \frac{2}{3}\ \frac{1}{6}]$	$[\frac{1}{120}\ \frac{13}{60}\ \frac{11}{20}\ \frac{13}{60}\ \frac{1}{120}]$
$\hat{P}_{\beta^k}(\frac{\nu}{2\pi})$	1	$\frac{1}{3}(2+\cos(\nu))$	$\frac{1}{60}(33+26\cos(\nu)+\cos(2\nu))$
$\lambda_1[n]$	$[1\ -2\ 1]$	$[\frac{-1}{12}\ \frac{4}{3}\ \frac{-5}{2}\ \frac{4}{3}\ \frac{-1}{12}]$	$[\frac{1}{90}\ \frac{-3}{20}\ \frac{3}{2}\ \frac{-49}{18}\ \frac{3}{2}\ \frac{-3}{20}\ \frac{1}{90}]$
$\hat{\Lambda}_1(\frac{\nu}{2\pi})$	$-2+2\cos(\nu)$	$\frac{2}{3}(-7+\cos(\nu))\sin^2(\frac{\nu}{2})$	$\frac{2}{45}(-111+23\cos(\nu)-2\cos(2\nu))\sin^2(\frac{\nu}{2})$
$w_1[n]$			$[\frac{1}{362880}\ \frac{251}{181440}\ \frac{913}{22680}\ \frac{44117}{181440}\ \frac{15619}{36288}\ \cdots]$
$\hat{W}_1(\frac{\nu}{2\pi})$			$\frac{1}{181440}(78095+88234\cos(\nu)+14608\cos(2\nu)+502\cos(3\nu)+\cos(4\nu))$
$a_{\beta^k}[n]$			$[\frac{1}{5040}\ \frac{1}{42}\ \frac{397}{1680}\ \frac{151}{315}\ \frac{397}{1680}\ \frac{1}{42}\ \frac{1}{5040}]$
$\hat{A}_{\beta^k}(\frac{\nu}{2\pi})$			$\frac{1}{2520}(1208+1191\cos(\nu)+120\cos(2\nu)+\cos(3\nu))$

TABLE II. FILTERS TO BE USED IN CONJUNCTION WITH THE LINEAR AND QUINTIC BOX SPLINES ON THE BCC LATTICE. $\mathrm{c}(\theta)$ IS SHORT FOR $\cos(\theta)$.

	$\mathbb{V}(\vartheta^2,\mathcal{H}_h)$ Interp.	$\mathbb{V}(\vartheta^2,\mathcal{H}_h)$ Interp.	$\mathbb{V}(\vartheta^4,\mathcal{H}_h)$ Quasi. ($l=6$)
$p_{\vartheta^k}[\boldsymbol{y}]$ $(\boldsymbol{y}\in\mathcal{H})$	$\delta_{\boldsymbol{y}}$	$\begin{cases}\frac{2}{5} & \text{if } \boldsymbol{y}=\boldsymbol{0},\\ \frac{1}{20} & \text{if } \|\boldsymbol{y}\|=\sqrt{3},\\ \frac{1}{30} & \text{if } \|\boldsymbol{y}\|=2,\\ 0 & \text{otherwise}\end{cases}$	$\begin{cases}\frac{379}{1680} & \text{if } \boldsymbol{y}=\boldsymbol{0},\\ \frac{1177}{20160} & \text{if } \|\boldsymbol{y}\|=\sqrt{3},\\ \frac{7}{180} & \text{if } \|\boldsymbol{y}\|=2,\\ \frac{43}{10080} & \text{if } \|\boldsymbol{y}\|=\sqrt{8},\\ \frac{17}{20160} & \text{if } \|\boldsymbol{y}\|=\sqrt{11},\\ \frac{1}{4032} & \text{if } \|\boldsymbol{y}\|=\sqrt{12},\\ \frac{1}{10080} & \text{if } \|\boldsymbol{y}\|=4,\\ 0 & \text{otherwise}\end{cases}$
$\hat{P}_{\vartheta^k}(\frac{u}{2\pi},\frac{v}{2\pi},\frac{w}{2\pi})$	1	$\frac{1}{15}\big(6+\mathrm{c}(2u)+\mathrm{c}(2v)+\mathrm{c}(2w)+6\,\mathrm{c}(u)\,\mathrm{c}(v)\,\mathrm{c}(w)\big)$	$\frac{1}{5040}\Big(2\,\mathrm{c}(u)\,\mathrm{c}(v)\,\mathrm{c}(w)\big[34\,\mathrm{c}(2u)+34\,\mathrm{c}(2v)+1143\big]+2\,\mathrm{c}(2w)\big[\mathrm{c}(2u)\big(5\,\mathrm{c}(2v)+43\big)+43\,\mathrm{c}(2v)+196\big]+34\,\mathrm{c}(u)\,\mathrm{c}(v)\,\mathrm{c}(3w)+43\,\mathrm{c}\big(2(u-v)\big)+43\,\mathrm{c}\big(2(u+v)\big)+392\,\mathrm{c}(2u)+\mathrm{c}(4u)+392\,\mathrm{c}(2v)+\mathrm{c}(4v)+\mathrm{c}(4w)+1137\Big)$
$\lambda_1[n]$ $\hat{\Lambda}_1(\frac{\nu}{2\pi})$			see Table III
$w_{\mathsf{b}}[\boldsymbol{y}]$ $\hat{W}_{\mathsf{b}}(\boldsymbol{\omega})$	N/A	N/A	see supplementary material
$a_{\vartheta^k}[\boldsymbol{y}]$ $\hat{A}_{\vartheta^k}(\boldsymbol{\omega})$	N/A	N/A	see supplementary material

Filter	Notation	Filter wieghts
Derivative	$d[n]$	$[\frac{-1}{12}\ \frac{2}{3}\ 0\ \frac{-2}{3}\ \frac{1}{12}]$
Shifted Derivative	$s[n]$	$[\frac{-1}{24}\ \frac{9}{8}\ \frac{-9}{8}\ \frac{1}{24}\ 0]$
Second Derivative	$l[n]$	$[\frac{-1}{12}\ \frac{4}{3}\ \frac{-5}{2}\ \frac{4}{3}\ \frac{-1}{12}]$

TABLE III. FILTERS USED IN OUR EXPERIMENTS. THESE FILTERS ARE APPLIED ALONG THE PRINCIPAL LATTICE DIRECTIONS.

A. Robustness

When the input data contain noise, our method allows us to choose an appropriate smoothing parameter λ_2 to average out said error. However, our method fails to be as robust as the Screened Poisson algorithm when the data are non-uniform. We speculate that this is due to the fact that we employ no heuristics to account for the deviation from the assumption of uniformity (Figure 8.)

B. Complexity

In terms of re-sampling speed, representation memory consumption, and reconstruction time, the BCC approach tends to outperform the Cartesian. However, when compared to similar methods whose function spaces are sparse, our method is limited. Specifically, our memory requirement grows as $O(n^3)$ where n is the grid size, and the matrix solve is quite slow as n is large. While the regularization matrices on the BCC lattice are more sparse, and tend to converge faster, their size is still prohibitive for big n. These limitations prevent us from going to the same resolutions that similar methods allow.

One possible remedy to this, is to construct our solution within a wavelet or multiresolution space defined over the BCC lattice. Since the function we wish to reconstruct only needs a detailed representation near the surface or the model, one can expect much better memory consumption. However, this is still an open problem, and needs further research. Another possibility, would be to break the initial point-set up into

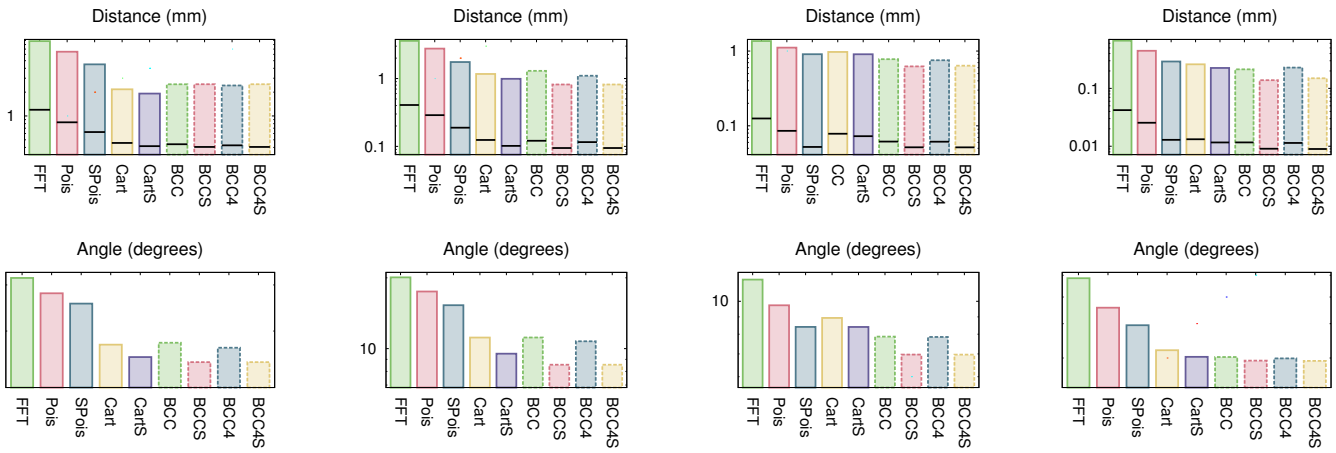


Fig. 5. An error comparison for the Gargoyle model as the grid resolution increases from 32^3 to 256^3 in steps of powers of two. The top row represents the Hausdorff (bar height) and mean distance (black line) to the original baseline model, while the bottom row represents the max angle deviation. $\lambda_1 = 100$, and λ_2, was chosen to be 1×10^{-3}, 1×10^{-4}, 1×10^{-5} and 1×10^{-6} from left to right respectively. Notice that the introduced shift greatly reduces angular error.

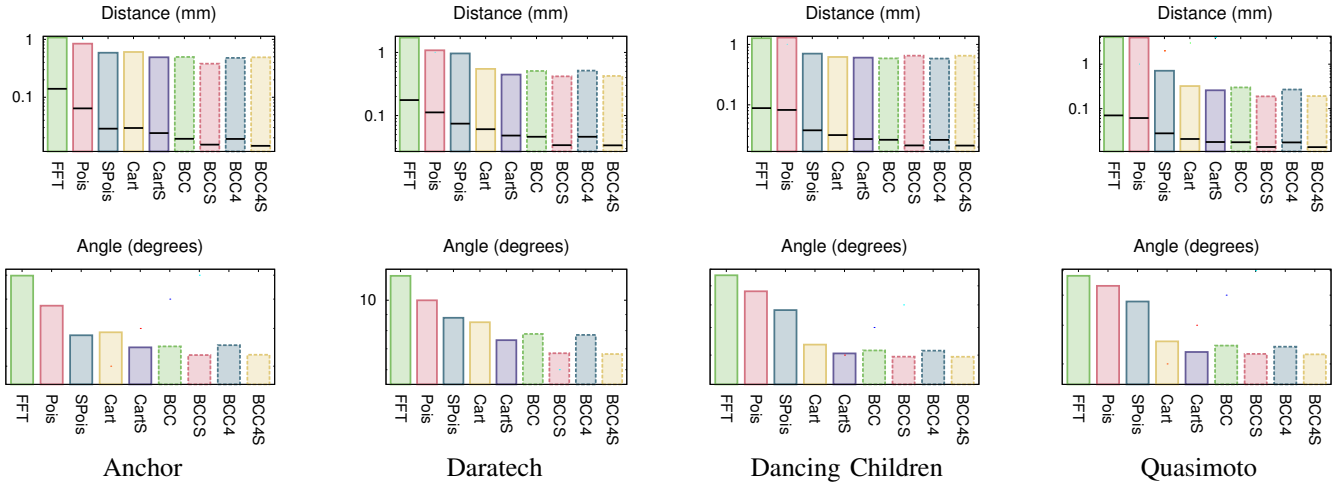


Fig. 6. A similar comparison for the case in which the grid is fixed at a resolution of 128^3 for all models. Here, $\lambda_1 = 1 \times 10^2$, and $\lambda_2 = 5 \times 10^{-5}$. Again, we see the same trend of the BCC lattice outperforming the comparable Cartesian lattices, and the Screened Poisson approach.

smaller sets, reconstruct the surface over those sets, then join the resulting meshes.

C. Parameter Selection

In our experiments, choosing λ_2 to be small reduced smoothing as expected. However, we observed that when λ_2 passed below a threshold, the surface began to show ringing or aliasing artifacts. This is somewhat to be expected, as choosing a very thin smoothing will cause the approximation (2) to break down. Additionally, the choice of λ_1, the compactness constraint, didn't seem to effect the solution much, it's only effect was on the choice of the parameter λ_2. That is, when λ_1 was large, λ_2 had to be large to obtain similar smooth solutions.

The parameterized nature of our solution allows us to control the degree to which the initial data are smoothed. However, this also makes it difficult to conclusively compare methods. More investigation and exploration of the parameter space is required. Moreover, there also appears to be a point at which choosing λ_2 to be too small introduces aliasing artifacts.

IX. Conclusion

The results regarding reconstruction methods are not surprising, they show that the Poisson methods in question are not using the full representation power of their approximation spaces they use. Further, when the we focus on solely on the method presented in this paper, we see an improvement in reconstruction quality. Reconstruction space can play a significant role in the fidelity of surface reconstruction. From the optimality argument, it is not surprising to see that function spaces defined over the BCC lattice tend to reconstruct more details at equivalent resolutions when compared to those

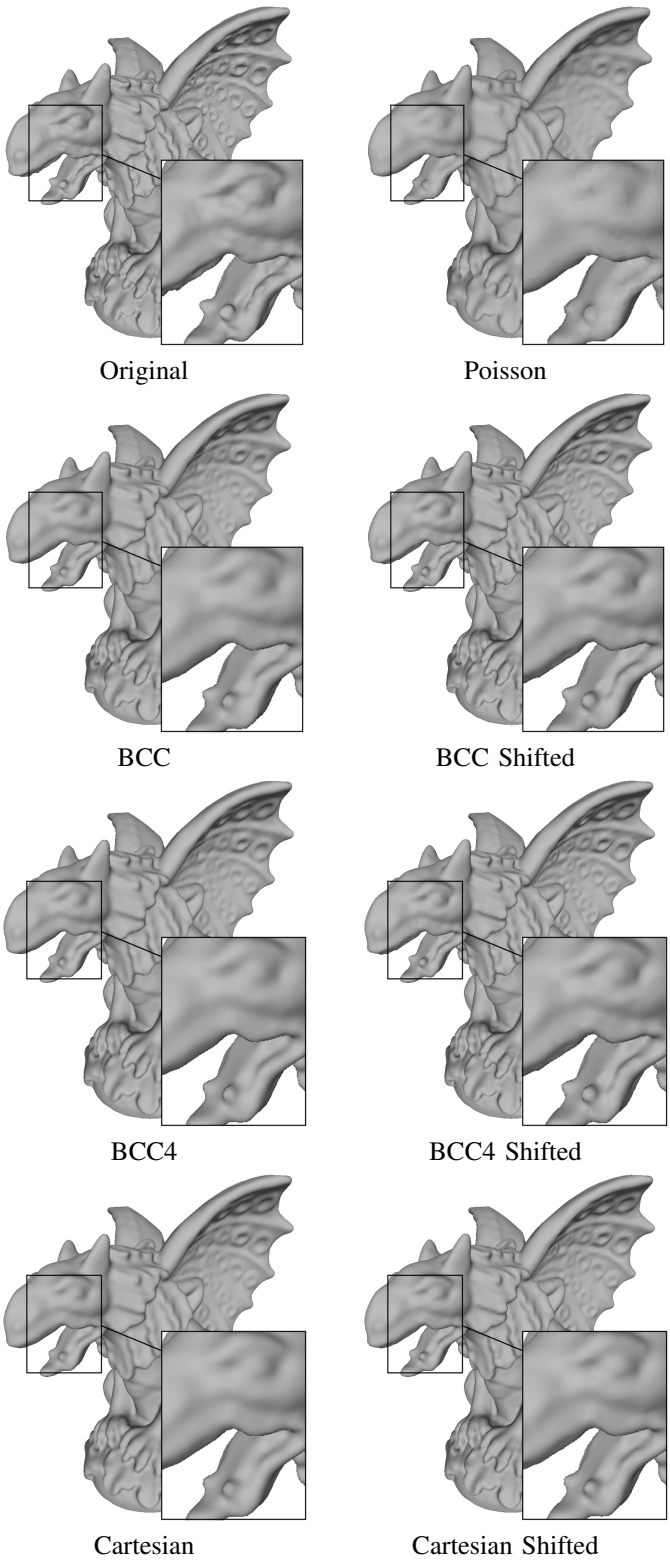


Fig. 7. A set of reconstructions on a 128^3 Cartesian grid, $50 \times 50 \times 100$ BCC grid or an oct-tree depth of 7. Parameters were chosen as $\lambda_1 = 1 \times 10^2$ and $\lambda_2 = 1 \times 10^{-4}$. Comparing between the Cartesian and BCC reconstruction spaces, the BCC lattice seems to consistently outperform the Cartesian lattice beyond a resolution of 128^3, moreover, the advantage of shifting the reconstruction space is apparent, specifically around the teeth and lips of the model.

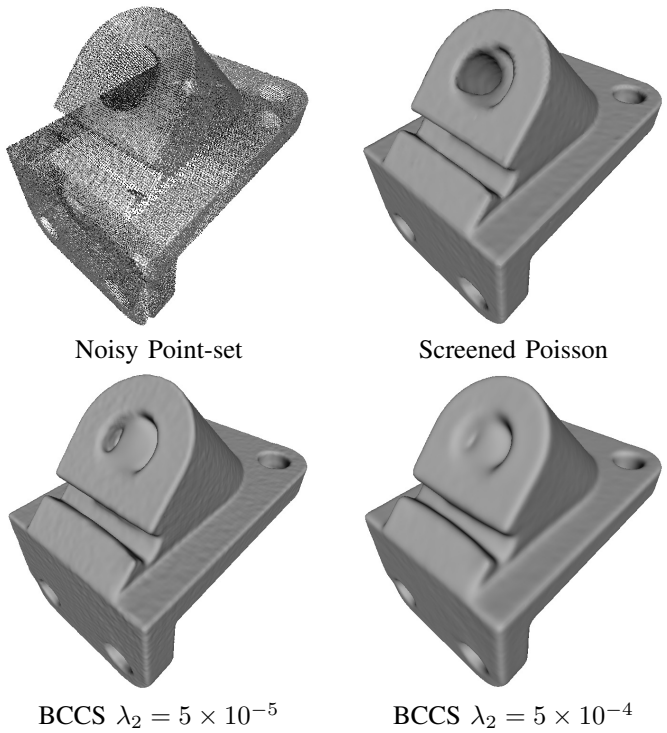


Fig. 8. Varying the smoothing parameter λ_2 allows us to smooth our noise from practical data sets ($\lambda_1 = 1 \times 10^2$). The Screened Poisson method is able to more faithfully reconstruct the center hole of the Anchor model. Our method, however, is unable to infer the importance of the samples on the interior of the Anchor.

defined over the Cartesian lattice. Furthermore, reconstructing within shifted spaces seems to better reconstruct higher frequency details.

The main hindrance for this method its cubic memory requirement, which prevents us from utilizing high density grids, and prevents the method from being used in practical settings verbatim. However, since no multiresolution techniques are currently available on non-Cartesian lattices in 3D, such topics are subject of future work. Additionally, a more complete analysis is required for parameter selection between the two lattices, and research into a faster (perhaps multiresolution) variational algorithm is needed.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] A. Entezari, D. Van De Ville, and T. Möller, "Practical box splines for reconstruction on the body centered cubic lattice," *IEEE Trans. Vis. and Comp. Graph.*, vol. 14, no. 2, pp. 313–328, 2008.

[2] U. R. Alim, T. Möller, and L. Condat, "Gradient estimation revitalized," *IEEE Trans. on Vis. and Comp. Graph.*, vol. 16, no. 6, pp. 1494–1503, November-December 2010.

[3] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, J. A. Levine, A. Sharf, and C. Silva, "State of the art in surface reconstruction from point clouds," *Eurographics STAR (Proc. of EG'14)*, 2014.

[4] F. Cazals and J. Giesen, "Delaunay triangulation based surface reconstruction: Ideas and algorithms," in *Effective computational geometry for curves and surfaces*. Springer, 2006, pp. 231–273.

[5] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust," in *Proc. ACM Symp. Solid Modeling and Applications*, ser. SMA '01. New York, NY, USA: ACM, 2001, pp. 249–266.

[6] N. Amenta, S. Choi, T. K. Dey, and N. Leekha, "A simple algorithm for homeomorphic surface reconstruction," in *Proc. SGP*, ser. SCG '00. New York, NY, USA: ACM, 2000, pp. 213–222.

[7] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. ACM SIGRAPH*, ser. SIGGRAPH '92. New York, NY, USA: ACM, 1992, pp. 71–78.

[8] F. Calakli and G. Taubin, "SSD: Smooth signed distance surface reconstruction," *Computer Graphics Forum*, vol. 30, no. 7, pp. 1993–2002, 2011.

[9] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans, "Reconstruction and representation of 3D objects with radial basis functions," in *Proc. ACM SIGRAPH*, ser. SIGGRAPH '01. New York, NY, USA: ACM, 2001, pp. 67–76.

[10] I. Macedo, J. P. Gois, and L. Velho, "Hermite radial basis functions implicits," *CG Forum*, vol. 30, no. 1, pp. 27–42, 2011.

[11] S. Fuhrmann and M. Goesele, "Floating scale surface reconstruction," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 46:1–46:11, Jul. 2014. [Online]. Available: http://doi.acm.org/10.1145/2601097.2601163

[12] M. Kazhdan, "Reconstruction of solid models from oriented point sets," in *Proc. SGP*, ser. SGP '05. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2005.

[13] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proc. SGP*, ser. SGP '06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 61–70.

[14] M. Kazhdan and H. Hoppe, "Screened Poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 29:1–29:13, Jul. 2013.

[15] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 2, pp. 20:1–20:17, Apr. 2013.

[16] J. Manson, G. Petrova, and S. Schaefer, "Streaming surface reconstruction using wavelets," *Proc. SGP*, vol. 27, no. 5, pp. 1411–1420, 2008.

[17] A. Entezari, R. Dyer, and T. Möller, "Linear and cubic box splines for the body centered cubic lattice," in *Visualization, 2004. IEEE*, 2004, pp. 11–18.

[18] B. Finkbeiner, U. R. Alim, D. V. D. Ville, and T. Möller, "High-quality volumetric reconstruction on optimal lattices for computed tomography," *Proceedings of Eurographics/IEEE-VGTC Symposium on Visualization*, vol. 28, no. 3, 2009, forthcoming.

[19] U. R. Alim, A. Entezari, and T. Mller, "The lattice-boltzmann method on optimal sampling lattices," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 4, pp. 630–641, 2009.

[20] B. Hamilton and S. Bilbao, "On finite difference schemes for the 3-d wave equation using non-cartesian grids," in *Proc. Sound and Music Computing (SMC) Conf., Stockholm, Sweden*, 2013, pp. 592–599.

[21] ——, "Hexagonal vs. rectilinear grids for explicit finite difference schemes for the two-dimensional wave equation," in *Proceedings of Meetings on Acoustics*, vol. 19, no. 1. Acoustical Society of America, 2013, p. 015120.

[22] M. Arigovindan, M. Suhling, P. Hunziker, and M. Unser, "Variational image reconstruction from arbitrarily spaced samples: a fast multiresolution spline solution," *IEEE Trans. on Img. Proc.*, vol. 14, no. 4, pp. 450–460, 2005.

[23] E. Vuçini, T. Möller, and E. Gröller, "On visualization and reconstruction from non-uniform point sets using B-splines," *Computer Graphics Forum (Proc. of Eurographics 2009)*, vol. 28, no. 3, pp. 1007–1014, 2009.

[24] X. Xu, A. Alvarado, and A. Entezari, "Reconstruction of irregularly-sampled volumetric data in efficient box spline spaces," *IEEE Trans. Med. Img.*, vol. 31, no. 7, pp. 1472–1480, 2012.

[25] A. Entezari, M. Mirzargar, and L. Kalantari, "Quasi-interpolation on the body centered cubic lattice," in *Proc. Eurographics / IEEE - VGTC Conf. on Vis.*, ser. EuroVis'09. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2009, pp. 1015–1022.

[26] P. Thevenaz, T. Blu, and M. Unser, "Interpolation revisited," *IEEE Trans. Med. Img.*, vol. 19, no. 7, pp. 739–758, 2000.

[27] C. de Boor, K. Höllig, and S. Riemenschneider, *Box Splines (Applied Mathematical Sciences)*, softcover reprint of hardcover 1st ed. 1993 ed. Springer, 2 2010.

[28] Z. Hossain, U. Alim, and T. Möller, "Toward high-quality gradient estimation on regular lattices," *IEEE Trans. on Vis. and Comp. Graph.*, vol. 17, no. 4, pp. 426–439, 2011.

[29] B. Finkbeiner, A. Entezari, D. Van De Ville, and T. Möller, "Efficient volume rendering on the body centered cubic lattice using box splines," *Comput. Graph.*, vol. 34, no. 4, pp. 409–423, Aug. 2010.

[30] T. Blu, P. Thevenaz, and M. Unser, "Linear interpolation revitalized," *IEEE Trans. Img. Proc.*, vol. 13, no. 5, pp. 710–719, May 2004.