Electronic Transactions on Numerical Analysis.
Volume 59, pp. 295–318, 2023.
Copyright © 2023, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol59s295

FILTERED POLYNOMIAL INTERPOLATION FOR SCALING 3D IMAGES[*]

DONATELLA OCCORSIO[†], GIULIANA RAMELLA[‡], AND WOULA THEMISTOCLAKIS[‡]

Dedicated to Professor Lothar Reichel on the occasion of his 70th Anniversary.

Abstract. Image scaling methods allow us to obtain a given image at a different, higher (upscaling) or lower (downscaling), resolution to preserve as much as possible the original content and the quality of the image. In this paper, we focus on interpolation methods for scaling three-dimensional grayscale images. Within a unified framework, we introduce two different scaling methods, respectively based on the Lagrange and filtered de la Vallée Poussin type interpolation at the zeros of Chebyshev polynomials of the first kind. In both cases, using a non-standard sampling model, we take (via tensor product) the associated trivariate polynomial interpolating the input image. It represents a continuous approximate 3D image to resample at the desired resolution. Using discrete ℓ^∞ and ℓ^2 norms, we theoretically estimate the error achieved in output, showing how it depends on the error in the input and on the smoothness of the specific image we are processing. Finally, taking the special case of medical images as a case study, we experimentally compare the performances of the proposed methods and with the classical multivariate cubic and Lanczos interpolation methods.

Key words. Image resizing, image downscaling, image upscaling, Lagrange interpolation, filtered VP interpolation, de la Vallée Poussin means, Chebyshev nodes

AMS subject classifications. 94A08, 68U10, 41A05, 62H35

1. Introduction.

In many applications, a fundamental task is to reduce a couple of images at the same resolution by upscaling the LR (Low Resolution) or downscaling the HR (High Resolution) image given in input. The resizing can be either the final goal or used as an intermediate step of more complex procedures in many fields such as medicine, communication, remote sensing, agriculture, etc. For instance, it is widely employed in medical diagnostics through image analysis, in particular, for Image Fusion of multimodal medical imaging, or for determining the exact position of an implant in the pre-surgery phase; see, e.g., [1, 5, 6, 20, 26, 27].

Due to the relevance of the problem, numerous scaling methods have been proposed in the literature; see, e.g., [2, 4, 8, 10, 11, 17, 19]. Here we focus on interpolation methods. They have been employed since the early years of computer graphics and image processing; see, e.g., [8].

The various interpolation methods for scaling images are united by two basic steps: first, obtaining an approximate continuous image from the discrete data, then sampling that approximation at the desired scale. In this context, the choices of the sampling system and the approximation tool are both decisive for the success of the scaling interpolation method.

Typically, equidistant nodes in each spatial coordinate are taken to form the sampling grids, and local, rather than global, interpolation is employed to get the approximate image. Here, following the ideas introduced in [12, 13], we propose a novel approach to resizing 3D images by interpolation methods.

More specifically, as a sampling system of nodes, we consider the non-uniform 3D grids generated by Chebyshev zeros of 1st kind of any degree, taken along each spatial coordinate. Moreover, as an approximation tool, we take the global, rather than local, approximation provided by the trivariate interpolation polynomials deduced, via tensor product, from the 1D

[*] Received December 6, 2022. Accepted October 7, 2023. Published online on January 15, 2024. Recommended by M. Spalević.

[†] Department of Mathematics, Computer Science and Economics, viale dell'Ateneo Lucano 10, 85100 Potenza, Italy (donatella.occorsio@unibas.it).

[‡] C.N.R. National Research Council of Italy, Institute for Applied Computing "Mauro Picone", Via P. Castellino, 111, 80131 Naples, Italy ({giuliana.ramella, woula.themistoclakis}@cnr.it).

Lagrange or de la Vallée Poussin (briefly VP) filtered interpolation polynomials at Chebyshev zeros of 1st kind.

We recall that generalizing the trigonometric VP means, VP approximation polynomials are discrete versions of certain delayed arithmetic means of the Fourier–Chebyshev partial sums, which can be equivalently obtained by applying a VP filter to Fourier sums of higher degree. Such an approximation has been introduced as an alternative to classical Lagrange interpolation to ensure uniform convergence for any continuous function at a near-best approximation rate. One of the main peculiarities of VP interpolation is the dependence on two positive integer parameters: one, say n, determining the number of nodes, and the other, say m, determining the action ray of the VP filter. It is known that, by keeping fixed the former and suitably modulating the latter, we do not loose the interpolation property but may improve the local approximation and strongly attenuate the Gibbs phenomenon, occurring in the presence of isolated singularities. In addition, we have uniformly bounded Lebesgue constants with a calculable upper bound. For more details on VP approximation, we refer the interested reader to [14, 23, 24] and the references therein.

In the following, we briefly denote by LCI3 and VPI3 the scaling interpolation methods, respectively, employing the previous trivariate Lagrange and VP interpolating polynomials.

In the paper, we give a unified treatment of the two proposed methods that will be studied from both theoretical and experimental points.

Through some numerical experiments on medical images, we analyze and compare the performances of LCI3 and VPI3, also to the classic multivariate cubic [7] and Lanczos [9] interpolation methods. It turns out that, on average, VPI3 provides better performance than the other methods, generally with small differences in upscaling and a much larger gap in downscaling, especially if the image does not have higher frequency details. These numerical results confirm the trend already observed in [12, 13] dealing with scaling of 2D color images. As a novelty, in this paper, we include the comparison with the Lanczos scaling method. Moreover, we also focus on the case of resizing along only one of the three dimensions, which is a task often required in medical imaging.

From the theoretical side, we have to take into account that for increasing degrees $N < n$, the sets of zeros of the Chebyshev polynomials of the first kind are one included in the other one in all cases that the ratio n/N is odd. Consequently, as in [12, 13], in all downscaling with odd scale factors, LCI3 and VPI3 coincide and are a simple subsampling of the input image. In this case, the theoretical estimates of the MSE (Mean Squared Error) stated in [12, Proposition 1] and [13, Proposition 1] can be easily extended to LCI3 and VPI3 for all couple of input/output 3D images whose resolution satisfies the ideal limit of Nyquist–Shannon sampling theorem. In addition, we prove new error estimates in both the discrete ℓ_∞ and ℓ_2 norms, valid for any scale factor in downscaling as well as in upscaling.

In conclusion, we remark that our discussion is limited to 3D gray-scale images, but the generalization to resizing 3D color images can be easily obtained by applying the proposed LCI3 and VPI3 methods to each component of the RGB color space.

The outline of the paper is the following. In Section 2, we recall some basics on Lagrange and VP filtered interpolation in the univariate and trivariate cases. In Section 3, we expose the proposed resizing methods, providing computational details. In Section 4, the theoretical error estimates are given, and, finally, Section 5 deals with the numerical tests and comparison with other methods.

2. Lagrange and VP filtered interpolation.

This section is divided into two subsections concerning the Lagrange and VP polynomials interpolating at Chebyshev nodes in $[-1, 1]$, and their 3D extension to $[-1, 1]^3$ via tensors.

Throughout the paper, the notation $k = 1 : m$ will shortly denote $k \in \{1, 2, \dots, m\}$, for any $m \in \mathbb{N} = \{1, 2, 3, \dots\}$.

2.1. The univariate case. As usual, $\mathbb{P}_n$ denotes the set of all algebraic polynomials of degree at most n, for all $n \in \mathbb{N}_0 = \{0, 1, 2, \dots\}$. Denoting by $w(x) = (1 - x^2)^{-1/2}$ the Chebyshev weight of first kind, let $\{p_n(x)\}_{n\in\mathbb{N}_0}$ be the system of the corresponding orthonormal polynomials given by

$$p_n(x) = \begin{cases} \sqrt{\dfrac{2}{\pi}} \cos[nt], & n \geq 1, \\ \dfrac{1}{\sqrt{\pi}}, & n = 0, \end{cases} \qquad |x| \leq 1, \quad t = \arccos x.$$

Setting $K_n(x, y) := \sum_{j=0}^{n} p_j(x)p_j(y)$ the n-th Darboux kernel, and denoted by

$$\xi_k^n := \cos t_k^n, \qquad \text{with} \qquad t_k^n := \frac{(2k-1)\pi}{2n}, \quad k = 1 : n \tag{2.1}$$

the zeros of $p_n(x)$, the Christoffel numbers of order n related to w are

$$\lambda_k^n = \frac{1}{K_n(\xi_k^n, \xi_k^n)} = \frac{\pi}{n}.$$

Letting $L_n f \in \mathbb{P}_{n-1}$ the Lagrange polynomial interpolating a given function f at the nodes in (2.1), i.e.,

$$L_n f(\xi_k^n) = f(\xi_k^n), \quad k = 1 : n,$$

with

$$L_n f(x) = \sum_{k=1}^{n} f(\xi_k^n)\ell_{n,k}(x), \quad \ell_{n,k}(x) = \frac{p_n(x)}{p_n'(\xi_k^n)(x - \xi_k^n)},$$

we recall that the fundamental Lagrange polynomials $\{\ell_{n,k}(x)\}_{k=1}^n$ can be represented in terms of the Darboux kernel as

$$\ell_{n,k}(x) = \lambda_k^n K_n(x, \xi_k^n), \qquad k = 1 : n. \tag{2.2}$$

For any pair of positive integers $m < n$, the VP approximation polynomial of f, $V_n^m f(x)$, is defined similarly to $L_n f(x)$ but using different fundamental polynomials. More precisely, we have [23]

$$V_n^m f(x) := \sum_{k=1}^{n} f(\xi_k^n)\Phi_{n,k}^m(x),$$

where the fundamental VP polynomials $\Phi_{n,k}^m$ are given by the following delayed arithmetic mean of Darboux kernels

$$\Phi_{n,k}^m(x) = \frac{\lambda_k^n}{2m} \sum_{r=n-m}^{n+m-1} K_r(\xi_k^n, x), \qquad k = 1 : n,$$

i.e., more explicitly

$$\Phi_{n,k}^m(x) := \lambda_k^n \sum_{j=0}^{n+m-1} \mu_{n,j}^m p_j(\xi_k^n) p_j(x), \qquad k = 1:n,$$

where $\mu_{n,j}^m$ are the VP filtering coefficients defined as

$$\mu_{n,j}^m := \begin{cases} 1, & \text{if } 0 \leq j \leq n-m, \\ \dfrac{n+m-j}{2m}, & \text{if } n-m < j < n+m. \end{cases}$$

Moreover, we recall the following useful representation of the fundamental VP polynomials [3, 23]

$$\Phi_{n,k}^m(x) = \lambda_k^n \sum_{r=0}^{n-1} p_r(\xi_k^n) q_{m,r}^n(x), \quad k = 1:n, \tag{2.3}$$

where

$$q_{n,j}^m(x) := \begin{cases} p_j(x), & \text{if } 0 \leq j \leq n-m, \\ \gamma_{n,j}^m p_j(x) - \gamma_{n,2n-j}^m p_{2n-j}(x), & \text{if } n-m < j < n, \end{cases} \tag{2.4}$$

with $\gamma_{n,j}^m := \frac{m+n-j}{2m}$. Comparing Lagrange and VP polynomials, we recall that both of them need only the n function values $\{f(\xi_k^n)\}_{k=1}^n$ for their computation. Moreover, they both are interpolating polynomials at the same set of nodes, being (see [23] for the VP case)

$$V_n^m f(\xi_k^n) = f(\xi_k^n) = L_n f(\xi_k^n), \quad k = 1:n, \qquad 0 < m < n. \tag{2.5}$$

However, the polynomial $V_n^m f$ depends on the free parameter $m = 1 : n-1$ which can arbitrarily vary without losing the interpolation property (2.5); see [23].

As shown in Figure 2.1, the additional parameter m modulates the localization degree of the fundamental VP polynomials.

Moreover, note that in the limiting case $m = 0$ we get the fundamental Lagrange polynomials, i.e.,

$$\Phi_{n,k}^0(x) = \ell_{n,k}(x), \qquad \forall |x| \leq 1, \qquad k = 1:n,$$

while, at the opposite limiting case $m = n$, we get the following Fejer means

$$\Phi_{n,k}^n(x) = \lambda_k^n \left[\frac{1}{2n} \sum_{r=0}^{2n-1} K_r(\xi_k^n, x) \right], \qquad \forall |x| \leq 1, \qquad k = 1:n.$$

Finally, as regards Lagrange and VP operators, i.e., $L_n : f \to L_n f$ and $V_n^m : f \to V_n^m f$, we recall that the former is a projection on $\mathbb{P}_{n-1}$ while the latter is a projection on the polynomial space [23]

$$\mathrm{S}_m^n := \text{span}\{\Phi_{n,k}^m : \ k = 1:n\} = \text{span}\{q_{n,j}^m : \ j = 0:(n-1)\},$$

which is nested between $\mathbb{P}_{n-m}$ and $\mathbb{P}_{n+m-1}$ and has the same dimension of $\mathbb{P}_{n-1}$.

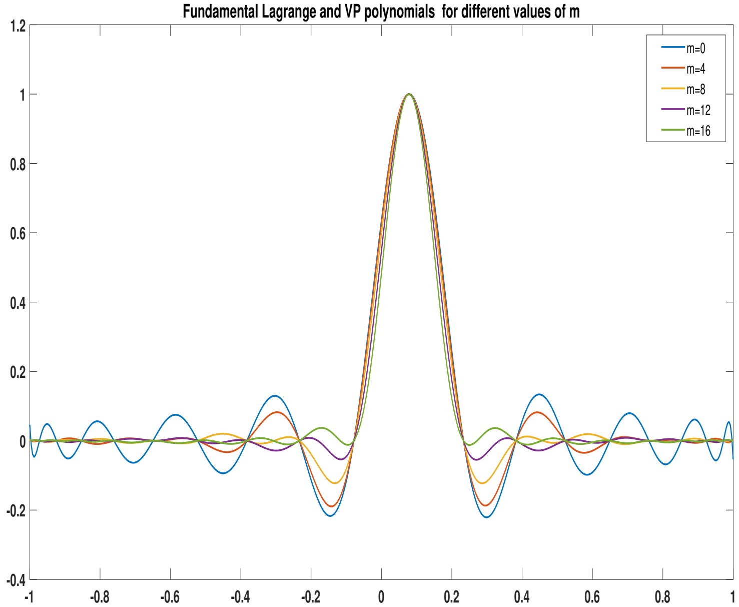


FIG. 2.1. *Fundamental* $\Phi^m_{n,k}$*, for* $n = 20,\ k = 11$ *and different values of* m*.*

2.2. Trivariate Lagrange and VP interpolation polynomials. Let $\mathcal{D} := [-1, 1]^3$, and use the notation $\xi := (x, y, z)$ to denote an arbitrary point of $\mathcal{D}$. For any multi-index $\mathbf{n} := (n_1, n_2, n_3) \in \mathbb{N}^3$, let $\mathbf{P_n}$ denote the space of trivariate polynomials of degree n_i in the i-th variable, for $i = 1, 2, 3$. Moreover, indicate by

$$X_{n_1} := \{\xi^{n_1}_k\}_{k=1:n_1}, \quad X_{n_2} = \{\xi^{n_2}_k\}_{k=1:n_2}, \quad X_{n_3} = \{\xi^{n_3}_k\}_{k=1:n_3}$$

the set of zeros of Chebyshev polynomials of the 1st kind of degree n_1, n_2, n_3, respectively (see (2.1)) . These zeros generate the 3D grid $X_\mathbf{n} := X_{n_1} \times X_{n_2} \times X_{n_3}$, and we denote an arbitrary node of such a grid by

$$\xi^\mathbf{n}_{ijk} = (\xi^{n_1}_i, \xi^{n_2}_j, \xi^{n_3}_k), \qquad i = 1 : n_1, \quad j = 1 : n_2, \quad k = 1 : n_3. \tag{2.6}$$

For any function g defined on the cube $\mathcal{D}$, we recall that the trivariate Lagrange polynomial $\mathbf{L_n} g \in \mathbb{P}_{\mathbf{n}-1}$ interpolating g at the grid $\mathrm{X}_\mathbf{n}$, i.e.,

$$\mathbf{L_n} g(\xi^\mathbf{n}_{ijk}) = g(\xi^\mathbf{n}_{ijk}), \qquad \forall \xi^\mathbf{n}_{ijk} \in \mathrm{X}_\mathbf{n},$$

has the following form

$$\mathbf{L_n} g(\xi) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} g(\xi^\mathbf{n}_{ijk}) \ell_{n_1,i}(x) \ell_{n_2,j}(y) \ell_{n_3,k}(z), \qquad \xi = (x, y, z) \in \mathcal{D}.$$

Similarly, introducing the additional multi-index $\mathbf{m} = (m_1, m_2, m_3) \in \mathbb{N}^3$ s.t. $m_i < n_i$, $i = 1 : 3$ (briefly $\mathbf{m} < \mathbf{n}$), the trivariate VP polynomial of g based on the nodes $X_\mathbf{n}$ and the parameter $\mathbf{m}$ can be written as follows

$$\mathbf{V^m_n} g(\xi) := \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} g(\xi^\mathbf{n}_{i,j,k}) \Phi^{m_1}_{n_1,i}(x) \Phi^{m_2}_{n_2,j}(y) \Phi^{m_3}_{n_3,k}(z), \qquad \xi = (x, y, z) \in \mathcal{D}.$$

For any choice of $\mathbf{m} < \mathbf{n}$, $V_{\mathbf{n}}^{\mathbf{m}} g \in \mathbb{P}_{\mathbf{n}+\mathbf{m}-1}$ and it coincides with g at the grid $\mathrm{X}_{\mathbf{n}}$, i.e.,

$$\mathbf{V}_{\mathbf{n}}^{\mathbf{m}} g(\xi_{ijk}^{\mathbf{n}}) = g(\xi_{ijk}^{\mathbf{n}}), \qquad \forall \xi_{ijk}^{\mathbf{n}} \in \mathrm{X}_{\mathbf{n}}.$$

3. Resampling by interpolation. For all $\mathbf{n} = (n_1, n_2, n_3) \in \mathbb{N}^3$, let $\mathcal{I}_{\mathbf{n}}$ be any 3D gray-scale digital image of size $n_1 \times n_2 \times n_3$. From the mathematical point of view, $\mathcal{I}_{\mathbf{n}}$ is a 3D array whose entries are nonnegative integers giving the gray levels at each voxel. According to the sampling model introduced in [12], such values can be regarded as the values attained at the Chebyshev grid $X_{\mathbf{n}}$ (cf. (2.6)) by a function $f : \mathcal{D} \to \mathbb{R}$ representing the image at a continuous scale, namely

$$\mathcal{I}_{\mathbf{n}}(i,j,k) = f(\xi_{ijk}^{\mathbf{n}}), \qquad i = 1:n_1, \quad j = 1:n_2, \quad k = 1:n_3. \tag{3.1}$$

Moreover, denoting by $P_{\mathbf{n}} f$ the Lagrange or VP polynomial interpolating f at $\mathrm{X}_{\mathbf{n}}$, by (2.5) we have

$$\mathcal{I}_{\mathbf{n}}(i,j,k) := P_{\mathbf{n}} f(\xi_{ijk}^{\mathbf{n}}), \qquad i = 1:n_1, \quad j = 1:n_2, \quad k = 1:n_3.$$

Now, the scaling problem consists in resizing the image $\mathcal{I}_{\mathbf{n}}$ to a different resolution, say $N_1 \times N_2 \times N_3$, being $\mathbf{N} = (N_1, N_2, N_3)$ and $\mathbf{N} \neq \mathbf{n}$. Let $\mathrm{X}_{\mathbf{N}} = X_{N_1} \times X_{N_2} \times X_{N_3}$ the associated Chebyshev grid (denser or coarser than $\mathrm{X}_{\mathbf{n}}$), this problem can be formulated as the construction of a new digital image $\mathcal{I}_{\mathbf{N}}$ such that

$$\mathcal{I}_{\mathbf{N}}(i,j,k) = f(\xi_{ijk}^{\mathbf{N}}), \qquad i = 1:N_1, \quad j = 1:N_2, \quad k = 1:N_3. \tag{3.2}$$

Since $\mathcal{I}_{\mathbf{N}}$ is unknown, the idea of the method is to recover the image $\widetilde{\mathcal{I}}_{\mathbf{N}}$ by evaluating, instead of f, $P_{\mathbf{n}} f$ at the grid X_N to obtain

$$\widetilde{\mathcal{I}}_{\mathbf{N}}(i,j,k) = P_{\mathbf{n}} f(\xi_{ijk}^{\mathbf{N}}), \qquad i = 1:N_1, \quad j = 1:N_2, \quad k = 1:N_3.$$

This formula provides the output resized image of the method that we denote by LCI3 and VPI3 depending on whether it is $P_{\mathbf{n}} = L_{\mathbf{n}}$ or $P_{\mathbf{n}} = V_{\mathbf{n}}^{\mathbf{m}}$. More explicitly, providing $I_{\mathbf{n}}$ as input image, for $i = 1:N_1,\ j = 1:N_2,\ k = 1:N_3$, we have

- LCI3 output image:

$$\begin{aligned}\widetilde{\mathcal{I}}_{\mathbf{N}}(i,j,k) &= L_{\mathbf{n}} f(\xi_{ijk}^{N}) \\ &= \sum_{h=1}^{n_1} \sum_{r=1}^{n_2} \sum_{s=1}^{n_3} \mathcal{I}_{\mathbf{n}}(h,r,s) \ell_{n_1,h}(\xi_i^{N_1}) \ell_{n_2,r}(\xi_j^{N_2}) \ell_{n_3,s}(\xi_k^{N_3}),\end{aligned}$$

- VPI3 output image:

$$\begin{aligned}\widetilde{\mathcal{I}}_{\mathbf{N}}(i,j,k) &= V_{\mathbf{n}}^{\mathbf{m}} f(\xi_{ijk}^{N}) \\ &= \sum_{h=1}^{n_1} \sum_{r=1}^{n_2} \sum_{s=1}^{n_3} \mathcal{I}_{\mathbf{n}}(h,r,s) \Phi_{n_1,h}^{m_1}(\xi_i^{N_1}) \Phi_{n_2,r}^{m_2}(\xi_j^{N_2}) \Phi_{n_3,s}^{m_3}(\xi_k^{N_3}).\end{aligned}$$

By these formulae, different numerical procedures can be performed. In particular, introducing the Vandermonde-like matrices

$$\mathbf{L}_k := \left[\ell_{n_k,i}(\xi_j^{N_k})\right]_{i,j} \in \mathbb{R}^{n_k \times N_k} \quad \text{and} \quad \mathbf{V}_k := \left[\Phi_{n_k,i}^{m_k}(\xi_j^{N_k})\right]_{i,j} \in \mathbb{R}^{n_k \times N_k} \qquad k = 1:3,$$

the output 3D array $\widetilde{\mathcal{I}}_{\mathbf{N}}$ can be computed from the input array $\mathcal{I}_{\mathbf{n}}$ by one of the following formulae

$$\widetilde{\mathcal{I}}_{\mathbf{N}} = \begin{cases} \mathbf{L}_3^T \boldsymbol{\Upsilon}_{\mathbf{L}}, & LCI3 \\ \mathbf{V}_3^T \boldsymbol{\Upsilon}_{\mathbf{V}}, & VPI3 \end{cases} \tag{3.3}$$

with

$$\boldsymbol{\Upsilon}_V = [\mathbf{V}_1 \mathbf{I}^{(1)} \mathbf{V_2}^T, \dots, \mathbf{V}_1 \mathbf{I}^{(n_3)} \mathbf{V_2}^T]^T \qquad \boldsymbol{\Upsilon}_L = [\mathbf{L}_1 \mathbf{I}^{(1)} \mathbf{L_2}^T, \dots, \mathbf{L}_1 \mathbf{I}^{(n_3)} \mathbf{L_2}^T]^T$$

block vectors of length n_3, where $\{\mathbf{I}^{(k)}\}_{k=1}^{n_3}$ are the slices of the input image of dimensions $n_1 \times n_2$.

Note that in case a lot of images of initial equal dimension have to be resized to the same final size, formula (3.3) allows to pre-compute once the matrices $\mathbf{L}_k$ or $\mathbf{V}_k$, with the possibility of using a parallel algorithm.

We also observe that fast computation of the matrices $\mathbf{L}_k$ and $\mathbf{V}_k$ can be achieved by using the more convenient trigonometric forms of the fundamental Lagrange and VP univariate polynomials, that yield both such polynomials in terms of the Inverse Discrete Cosine Transform (IDCT) of the respective orthogonal basis polynomials. More precisely, denote by n, N, m the generic components of the vectors $\mathbf{n}, \mathbf{N}, \mathbf{m}$, respectively. About the fundamental Lagrange polynomials, by (2.2), we have

$$\ell_{n,i}(\xi_k^N) = \frac{\sqrt{\pi}}{n}\left[p_0(\xi_k^N) + \sqrt{2}\sum_{r=1}^{n-1} \cos\left[\frac{(2i-1)r\pi}{2n}\right] p_r(\xi_k^N)\right], \quad k = 1:N, \quad i = 1:n. \tag{3.4}$$

Similarly, for the fundamental VP polynomials, by (2.3)–(2.4) we get for $k = 1 : N$ and $i = 1 : n$

$$\Phi_{n,i}^m(\xi_k^N) = \frac{\sqrt{\pi}}{n}\left[q_{m,0}^n(\xi_k^N) + \sqrt{2}\sum_{r=1}^{n-1} \cos\left[\frac{(2i-1)r\pi}{2n}\right] q_{m,r}^n(\xi_k^N)\right], \tag{3.5}$$

where the orthogonal basis polynomials $q_{m,r}^n$ take the following trigonometric form

$$q_{m,r}^n(\xi_k^N) = \begin{cases} \cos\left[\dfrac{(2k-1)r\pi}{2N}\right], & \text{if } 0 \le r \le n-m, \\[2ex] \dfrac{n+m-r}{2m} \cos\left[\dfrac{(2k-1)r\pi}{2N}\right] & \\ \qquad + \dfrac{n-m-r}{2m} \cos\left[\dfrac{(2k-1)(2N-r)\pi}{2N}\right], & \text{if } n-m < r < n. \end{cases}$$

On the other hand, recall that the IDCT of any data set

$$\mathcal{A} = \{a_{r,k} \;\; r = 0 : (n-1), \; k = 1 : N\}$$

is defined as (see e.g. [16])

$$IDCT(\mathcal{A}) = \left\{ b_{h,k} = \sum_{r=0}^{n-1} w_r a_{r,k} \cos\left[\frac{(2h-1)r\pi}{2n}\right] \; : \; h = 1:n, \; k = 1:N \right\}$$

where $w_r = \sqrt{1/n}$ if $r = 0$, and $w_r = \sqrt{2/n}$ otherwise.

Hence, the matrices $L = \left[\ell_{n,h}(\xi_k^N)\right]_{h,k}$ and $V = \left[q^m_{n,h}(\xi_k^N)\right]_{h,k}$ can be computed by

$$L = \sqrt{\frac{\pi}{n}} IDCT[p_r(\xi_k^N)], \qquad V = \sqrt{\frac{\pi}{n}} IDCT[q^n_{m,r}(\xi_k^N)].$$

In conclusion, let us examine two particular cases where the computation of the output image $\widetilde{I}_N$ can be achieved by simpler formulas.

3.1. Downscaling with odd scale factors. We point out that when downscaling with odd scale factors, the implemented algorithm is based on the decimation (subsampling) of the original image. This comes from the fact that if $n = sN$ with s odd then the zeros of the first kind Chebyshev polynomial of degree N are also zeros of the 1st kind Chebyshev polynomial of degree n , i.e., by (2.1) we get

$$\xi_j^N = \cos\frac{(2j-1)\pi}{2N} = \cos\frac{s(2j-1)\pi}{2n} = \xi^n_{\frac{s(2j-1)+1}{2}}, \qquad j = 1 : N, \tag{3.6}$$

where we remark that s odd implies that $s(2j-1)+1$ is even for all $j \in \mathbb{N}$.

Hence, in the case that, for $i = 1 : 3$, the ratio $s_i := n_i/N_i$ is an odd integer, as regards the associated 3D Chebyshev grids of dimension $\mathbf{N} = (N_1, N_2, N_3)$ and $\mathbf{n} = (n_1, n_2, n_3)$, we get $\mathrm{X}_{\mathbf{N}} \subseteq \mathrm{X}_{\mathbf{n}}$. Consequently, by the interpolation property (2.5), LCI3 and VPI3 coincide and their output LR image is achieved by the following reassignment of the input voxels values

$$\widetilde{\mathcal{I}}_{\mathbf{N}}(i,j,k) = \mathcal{I}_{\mathbf{n}}\left(\frac{s_1(2i-1)+1}{2}, \frac{s_2(2j-1)+1}{2}, \frac{s_3(2k-1)+1}{2}\right),$$

with $i = 1 : N_1$, $j = 1 : N_2$, and $k = 1 : N_3$.

3.2. Downsampling along only one direction. In the special case that $N_1 = n_1$, $N_2 = n_2$, and $N_3 \neq n_3$, we are resizing along only the third dimension of the input image. In this case the output resized image is given by (3.3) but we simply have

$$\boldsymbol{\Upsilon}_V = [\mathbf{I}^{(1)}, \ldots \mathbf{I}^{(n_3)}]^T \qquad \boldsymbol{\Upsilon}_L = [\mathbf{I}^{(1)}, \ldots, \mathbf{I}^{(n_3)}]^T,$$

where, for $k = 1 : n_3$, $\mathbf{I}^{(k)}$ are the following 2D slices of the input image

$$\mathbf{I}^{(k)} = [I_{\mathbf{n}}(i,j,k)]_{i=1:n_1, j=1:n_2}, \qquad k = 1 : n_3.$$

An analogous formula also holds for resizing along only the 1st or the 2nd direction ($N_i \neq n_i$ only for $i = 1$ or $i = 2$, respectively) .

4. Error analysis. In this section, we provide several theoretical estimates for the error when we have in resampling by LCI3 and VPI3 methods.

Let us first focus on the univariate case that corresponds to resizing along only one dimension. Maintaining the previous notations, in the univariate setting, we suppose that

$$I_n(j) = f(\xi_j^n),\ j = 1 : n, \qquad \text{and} \qquad I_N(j) = f(\xi_j^N),\ j = 1 : N,$$

are the vectors representing the target function $f : [-1, 1] \to \mathbb{R}$ at two different sizes $n \neq N$, and we aim to approximate I_N starting from I_n. To consider also the case of corrupted input data, we suppose our starting data are the following

$$\widetilde{I}_n(j) = \tilde{f}(\xi_j^n),\ j = 1 : n,$$

where $\tilde f : [-1,1] \to \mathbb{R}$ represents a more or less corrupted version of f.

Hence, we compute the following approximation of I_N

$$\widetilde{I}_N(j) = L_n \tilde f(\xi_j^N) \qquad \text{or} \qquad \widetilde{I}_N(j) = V_n^m \tilde f(\xi_j^N) \tag{4.1}$$

depending on whether we are using Lagrange or VP interpolation.

In both cases, as it is well-known, the conditioning is measured by the associated Lebesgue function $\Lambda_n(x)$ and Lebesgue constant $\overline{\Lambda}_n$, defined as

$$\overline{\Lambda}_n = \sup_{|x|\le 1} \Lambda_n(x), \quad \text{with} \quad \Lambda_n(x) = \sum_{k=1}^{n} \begin{cases} |\ell_{n,k}(x)|, & \text{Lagrange interpolation,} \\ |\Phi_{n,k}^m(x)|, & \text{VP interpolation.} \end{cases}$$

In fact, setting

$$\|f\|_{\ell^\infty(X_n)} = \max_{1\le k\le n} |f(\xi_k^n)|, \qquad n \in \mathbb{N}$$

it is well known that

$$\|P_n f\|_{\ell^\infty(X_N)} \le \overline{\Lambda}_n \|f\|_{\ell^\infty(X_n)}, \qquad P_n \in \{L_n, V_n^m\} \tag{4.2}$$

and

(4.3)
$$|P_n f(\xi_j^N) - P_n \tilde f(\xi_j^N)| \le \Lambda_n(\xi_j^N) \|f - \tilde f\|_{\ell^\infty(X_n)}, \qquad P_n \in \{L_n, V_n^m\} \qquad j = 1:N.$$

Note that, starting from (3.4) and (3.5), by means of well-known trigonometric identities, we get that the values $\Lambda_n(\xi_j^N)$, $j = 1:N$ can be explicitly computed by the following formulas (see [15] for VP case)

$$\Lambda_n(\xi_j^N) = \begin{cases} \dfrac{1}{2n} \displaystyle\sum_{k=1}^{n} \left| \dfrac{\sin[(2n-1)(t_j^N - t_k^n)/2]}{\sin[(t_j^N - t_k^n)/2]} + \dfrac{\sin[(2n-1)(t_j^N + t_k^n)/2]}{\sin[(t_j^N + t_k^n)/2]} \right|, & P_n = L_n, \\[2ex] \dfrac{1}{4nm} \displaystyle\sum_{k=1}^{n} \left| \dfrac{\sin[m(t_j^N - t_k^n)] \sin[n(t_j^N - t_k^n)]}{\sin^2[(t_j^N - t_k^n)/2]} \right. \\ \qquad\qquad \left. + \dfrac{\sin[m(t_j^N + t_k^n)] \sin[n(t_j^N + t_k^n)]}{\sin^2[(t_j^N + t_k^n)/2]} \right|, & P_n = V_n^m. \end{cases}$$

Moreover, we recall that (see, e.g., [15, 25])

$$\overline{\Lambda}_n \le \begin{cases} \dfrac{2}{\pi} \log n + 0.52125\ldots, & P_n = L_n, \\ 2\sqrt{\dfrac{n}{m}} \left[1 + \dfrac{2\pi(n+m)}{n}\right], & P_n = V_n^m. \end{cases}$$

These bounds can be applied to the next theorem where we estimate the maximum error

$$\|I_N - \widetilde{I}_N\|_{\ell^\infty} = \max_{1\le j\le N} |I_N(j) - \widetilde{I}_N(j)|,$$

in terms of the input error

$$\|I_n - \widetilde{I}_n\|_{\ell^\infty} = \max_{1\le j\le n} |I_n(j) - \widetilde{I}_n(j)|,$$

and in terms of the smoothness of the target function f. As an indicator of such smoothness, we introduce the following error of the best polynomial discrete approximation of f

$$\epsilon_r(f)_{\ell^\infty(X_k)} = \inf_{Q_r\in\mathbb{P}_r} \|f - Q_r\|_{\ell^\infty(X_k)}, \quad r = \begin{cases} n-1, & P_n = L_n, \\ n-m, & P_n = V_n^m. \end{cases}$$

THEOREM 4.1. *Under the previous setting, we have*

$$\|I_N - \widetilde{I}_N\|_{\ell^\infty} \le \epsilon_r(f)_{\ell^\infty(X_N)} + \overline{\Lambda}_n \left[\epsilon_r(f)_{\ell^\infty(X_n)} + \|I_n - \widetilde{I}_n\|_{\ell^\infty}\right]. \tag{4.4}$$

Proof. Let $P_n f$ denote $L_n f$ or $V_n^m f$. For any $j = 1:N$, by (4.3) we get

$$\begin{aligned}|I_N(j) - \widetilde{I}_N(j)| = |f(\xi_j^N) - P_n\widetilde{f}(\xi_j^N)| &\le |f(\xi_j^N) - P_n f(\xi_j^N)| + |P_n f(\xi_j^N) - P_n\widetilde{f}(\xi_j^N)| \\ &\le |f(\xi_j^N) - P_n f(\xi_j^N)| + \Lambda_n(\xi_j^N)\, \|f - \widetilde{f}\|_{l^\infty(X_n)}.\end{aligned}$$

Moreover, recalling the invariance property (see [23] for the VP case)

$$P_n Q(x) = Q(x), \quad \forall x \in [-1,1], \qquad \forall Q \in \begin{cases} \mathbb{P}_{n-1}, & P_n = L_n, \\ \mathbb{P}_{n-m}, & P_n = V_n^m, \end{cases} \tag{4.5}$$

by (4.3) we also get

$$\begin{aligned} f(\xi_j^N) - P_n f(\xi_j^N)| &\le |f(\xi_j^N) - Q(\xi_j^N)| + |P_n(f-Q)(\xi_j^N)| \\ &\le |f(\xi_j^N) - Q(\xi_j^N)| + \Lambda_n(\xi_j^N)\, \|f - Q\|_{l^\infty(X_n)}.\end{aligned}$$

Collecting the previous estimates, for any polynomial Q as in (4.5), and for $j = 1:N$, we have

$$|I_N(j) - \widetilde{I}_N(j)| \le |f(\xi_j^N) - Q(\xi_j^N)| + \Lambda_n(\xi_j^N)\left[\|f - Q\|_{l^\infty(X_n)} + \|f - \widetilde{f}\|_{l^\infty(X_n)}\right]$$

and the statement follows by taking the maximum w.r.t $j = 1:N$ and the infimum with respect to the polynomials Q as in (4.5) at both sides. □

REMARK 4.2. Regarding possible estimates of the discrete errors $\epsilon_r(f)$ in (4.4), we note that

$$\|f - Q\|_{\ell^\infty(X_k)} \le \|f - Q\|_\infty = \max_{|x|\le 1} |f(x) - Q_r(x)|, \qquad \forall Q \in \mathbb{P}_r, \quad \forall k \in \mathbb{N}$$

and taking the infimum on $Q \in \mathbb{P}_r$ on both sides, we get

$$\epsilon_r(f)_{\ell^\infty(X_k)} \le E_r(f)_\infty = \inf_{Q\in\mathbb{P}_r} \|f - Q\|_\infty, \qquad \forall k \in \mathbb{N}.$$

Hence, to estimate the discrete errors in (4.4), we can theoretically apply all the results of approximation theory bounding the error of best uniform polynomial approximation $E_r(f)_\infty$

in term of the smoothness of f; see [21] and the references therein. For instance, by the classical Jackson inequalities, $\forall k \in \mathbb{N}$, we get

$$\epsilon_r(f)_{\ell^\infty(X_k)} \leq \mathcal{C}\omega\left(f, \frac{1}{r}\right), \qquad \forall f \in C^0[-1,1],$$
$$\epsilon_r(f)_{\ell^\infty(X_k)} \leq \frac{\mathcal{C}}{r^s}\omega\left(f^{(s)}, \frac{1}{r}\right), \qquad \forall f \in C^s[-1,1],$$

where $\mathcal{C} > 0$ is independent of f, r, k, and

$$\omega(f,\delta) = \max_{|t_1-t_2|<\delta} |f(t_1) - f(t_2)|.$$

In particular, if we know that for some $s \in \mathbb{N}_0$ the underlying function $f \in C^s[-1,1]$ and $f^{(s)}$ is Hölder continuous with exponent $0 < \alpha \leq 1$, we have

$$\epsilon_r(f)_{\ell^\infty(X_k)} = \mathcal{O}\left(\frac{1}{n^{s+\alpha}}\right).$$

Now, note that in image analysis, commonly used quality measures (see, e.g., the PSNR and SSIM quality measures defined in the next section) are based on the Root Mean Squared Error (RMSE), that in our setting is defined as follows

$$\|I_N - \widetilde{I}_N\|_{\ell^2} = \left(\frac{1}{N}\sum_{j=1}^{N} |I_N(j) - \widetilde{I}_N(j)|^2\right)^{\frac{1}{2}}.$$

Taking into account that $\|I_N - \widetilde{I}_N\|_{\ell^2} \leq \|I_N - \widetilde{I}_N\|_{\ell^\infty}$, we can certainly apply the previous estimates with the ℓ^∞-norm.

However, setting

$$\|f\|_{\ell^2(X_k)} = \left(\frac{1}{k}\sum_{j=1}^{k} |f(\xi_j^k)|^2\right)^{\frac{1}{2}}, \qquad k \in \mathbb{N},$$

and

$$\epsilon_r(f)_{\ell^2(X_k)} = \inf_{Q\in\mathbb{P}_r} \|f - Q\|_{\ell^2(X_k)}, \qquad r = \begin{cases} n-1, & P_n = L_n, \\ n-m, & P_n = V_n^m, \end{cases}$$

the next two theorems state the analogues of (4.2) and (4.4) with the ℓ^2-norm at both sides.

THEOREM 4.3. *Under the previous setting, there exists a constant $\mathcal{C} > 0$ depending on the ratio n/N but independent of n, N and f such that*

$$\|P_n f\|_{\ell^2(X_N)} \leq \mathcal{C}\|f\|_{\ell^2(X_n)}, \qquad P_n \in \{L_n, V_n^m\}. \tag{4.6}$$

Proof. By virtue of [15, Lemma 6.2], we know that if $Q(t)$ is an even or odd trigonometric polynomial of degree at most ν, then we have

$$\frac{1}{N}\sum_{j=1}^{N} |Q(t_j^N)| \leq \left(\frac{1}{\pi} + \frac{2\nu}{N}\right)\int_0^\pi |Q(\tau)|d\tau. \tag{4.7}$$

On the other hand, for all $k = 1 : n$, it is easy to check that $\ell_{n,k}(\cos t)$ and $\Phi^m_{n,k}(\cos t)$ are even trigonometric polynomials of degree $(n-1)$ and $(n+m-1)$, respectively. Hence, we can apply (4.7) to the trigonometric polynomial

$$Q(t) = \left[P_n f(\cos t)\right]^2, \qquad \nu = \deg(Q) = \begin{cases} 2(n-1), & P_n = L_n, \\ 2(n+m-1), & P_n = V_n^m, \end{cases}$$

obtaining

$$\begin{aligned}\|P_n f\|_{\ell^2(X_N)} &= \left(\frac{1}{N}\sum_{j=1}^N |P_n f(\cos t_j^N)|^2\right)^{\frac12} \leq \sqrt{\frac{1}{\pi}+\frac{2\nu}{N}}\left(\int_0^\pi |P_n f(\cos\tau)|^2 d\tau\right)^{\frac12} \\ &= \sqrt{\frac{1}{\pi}+\frac{2\nu}{N}}\left(\int_{-1}^1 |P_n f(x)|^2 \frac{dx}{\sqrt{1-x^2}}\right)^{\frac12}.\end{aligned}$$

However, in the case $P_n = L_n$, it is known that (see, e.g., [25])

$$\left(\int_{-1}^1 |L_n f(x)|^2 \frac{dx}{\sqrt{1-x^2}}\right)^{\frac12} = \sqrt{\pi}\|f\|_{\ell^2(X_n)},$$

and, similarly, in the case $P_n = V_n^m$, it has been proved that [3, Theorem 2.4]

$$\left(\int_{-1}^1 |V_n^m f(x)|^2 \frac{dx}{\sqrt{1-x^2}}\right)^{\frac12} \sim \|f\|_{\ell^2(X_n)},$$

where $a \sim b$ means that $\mathcal{C}^{-1}a \leq b \leq \mathcal{C}\, a$ holds with $\mathcal{C} > 0$, independent of n, m and f.

Hence, continuing the previous estimate for $P_n \in \{L_n, V_n^m\}$, we can state that there exists a constant $\mathcal{C} > 0$, independent of n, m and f such that

$$\|P_n f\|_{\ell^2(X_N)} \leq \sqrt{\frac{1}{\pi}+\frac{2\nu}{N}}\left(\int_{-1}^1 |P_n f(x)|^2 \frac{dx}{\sqrt{1-x^2}}\right)^{\frac12} \leq \mathcal{C}\|f\|_{\ell^2(X_n)}\sqrt{\frac{1}{\pi}+\frac{2\nu}{N}}.$$

Then, the statement follows by taking into account that $\nu/N \leq 4n/N$ implies that $\sqrt{\frac{1}{\pi}+\frac{2\nu}{N}}$ depends on the (fixed) scale factor n/N but is independent of n and N. □

THEOREM 4.4. *Under the previous setting, we have*

$$\|I_N - \widetilde{I}_N\|_{\ell^2} \leq \epsilon_r(f)_{\ell^2(X_N)} + \mathcal{C}\left[\epsilon_r(f)_{\ell^2(X_n)} + \|I_n - \widetilde{I}_n\|_{\ell^2}\right], \tag{4.8}$$

where $\mathcal{C} > 0$ is a constant depending on n/N but independent of n, N and $f, \widetilde{f}$.

Proof. Similarly to the proof of Theorem 4.1, for any polynomial Q as in (4.5), and for $j = 1 : N$, we have

$$\begin{aligned}|I_N(j) - \widetilde{I}_N(j)| &= |f(\xi_j^N) - P_n\widetilde{f}(\xi_j^N)| \\ &\leq |f(\xi_j^N) - Q(\xi_j^N)| + |P_n(Q-f)(\xi_j^N)| + |P_n(f-\widetilde{f})(\xi_j^N)|.\end{aligned}$$

Hence, applying (4.6), we obtain

$$\|I_N-\widetilde{I}_N\|_{\ell^2} \le \|f-Q\|_{\ell^e(X_N)} + \mathcal{C}\|f-Q\|_{\ell^e(X_n)} + \mathcal{C}\|f-\widetilde{f}\|_{\ell^e(X_n)}$$

that yields the statement by taking the infimum on Q at both sides. □

So far we have considered arbitrary input and output sizes, but we point out that in the special case

$$n = sN, \qquad \text{with} \quad s\in\mathbb{N} \quad \text{odd} \tag{4.9}$$

both the Lagrange and VP interpolation yield, by (4.1), the same output $\widetilde{I}_N$. Moreover, in such a special case the error estimates (4.4) and (4.8) are simplified as follows.

THEOREM 4.5. *If the input and output sizes, n and N respectively, satisfy (4.9), then we have*

$$\|I_N-\widetilde{I}_N\|_{\ell^\infty} \le \|I_n-\widetilde{I}_n\|_{\ell^\infty}, \tag{4.10}$$

and

$$\|I_N-\widetilde{I}_N\|_{\ell^2} \le \sqrt{s}\|I_n-\widetilde{I}_n\|_{\ell^2}. \tag{4.11}$$

Proof. The proof easily follows from the fact that, if (4.9) holds then (3.6) holds. Consequently, we have

$$I_N(j) = f\left(\xi^n_{\frac{s(2j-1)+1}{2}}\right) = I_n\left(\frac{s(2j-1)+1}{2}\right), \qquad j=1:N,$$

and, due to the interpolation property, we also have

$$\widetilde{I}_N(j) = \widetilde{f}\left(\xi^n_{\frac{s(2j-1)+1}{2}}\right) = \widetilde{I}_n\left(\frac{s(2j-1)+1}{2}\right), \qquad j=1:N.$$

By these identities (4.10) becomes trivial, and we also deduce (4.11) as follows

$$\|I_N-\widetilde{I}_N\|^2_{\ell^2} = \frac{s}{n}\sum_{j=1}^{N}\left|I_n\left(\frac{s(2j-1)+1}{2}\right) - \widetilde{I}_n\left(\frac{s(2j-1)+1}{2}\right)\right|^2 \le s\|I_n-\widetilde{I}_n\|^2_{\ell^2}. \qquad \square$$

In conclusion, let us extend the previous results to the general case of 3D rescaling from the size $\mathbf{n}=(n_1,n_2,n_3)$ to $\mathbf{N}=(N_1,N_2,N_3)$. To this end, henceforth, let us assume that the functions $f,\widetilde{f}:[-1,1]^3\to\mathbb{R}$ represent the image and its "corrupted" version, respectively. Moreover, let I_n, I_N be defined by sampling f as in (3.1), (3.2), and let

$$\widetilde{I}_{\mathbf{n}}(i,j,k)) = \widetilde{f}(\xi^{\mathbf{n}}_{ijk}), \qquad i=1:n_1, \quad j=1:n_2, \quad k=1:n_3,$$

be the input image that yields the following output

$$\widetilde{I}_{\mathbf{N}}(i,j,k)) = P_{\mathbf{n}}\widetilde{f}(\xi^{\mathbf{N}}_{ijk}), \qquad i=1:N_1, \quad j=1:N_2, \quad k=1:N_3,$$

where, as usual, we assume $P_{\mathbf{n}}=L_{\mathbf{n}}$ for LCI3 method and $P_{\mathbf{n}}=V^{\mathbf{m}}_{\mathbf{n}}$ for VPI3 method.

For simplicity, we continue to use the same notation of the univariate case also for the norm

$$\|I_{\mathbf{N}}\|_{\ell^p} = \begin{cases} \max\limits_{i,j,k}|I_N(i,j,k)|, & p=\infty, \\ \left(\dfrac{1}{N_1N_2N_3}\displaystyle\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3}|I_N(i,j,k)|^2\right)^{\frac{1}{2}}, & p=2. \end{cases}$$

Moreover, for any 3D-grid $X_{\mathbf{k}} = X_{k_1} \times X_{k_2} \times X_{k_3}$, $\mathbf{k} = (k_1, k_2, k_3) \in \mathbb{N}^3$, we set

$$\|f\|_{\ell^p(X_{\mathbf{k}})} = \begin{cases} \displaystyle\max_{\xi_{ijk} \in X_{\mathbf{k}}} |f(\xi_{ijk})|, & p = \infty, \\ \left(\displaystyle\frac{1}{k_1 k_2 k_3} \sum_{\xi_{ijk} \in X_{\mathbf{k}}} |f(\xi_i jk)|^2 \right)^{\frac{1}{2}}, & p = 2, \end{cases}$$

and

$$\mathcal{E}_{\mathbf{r}}(f)_{\ell^p(X_{\mathbf{k}})} = \inf_{Q \in \mathbb{P}_{\mathbf{r}}} \|f - Q\|_{\ell^p(X_{\mathbf{k}})} \quad \mathbf{r} = \begin{cases} \mathbf{n} - \mathbf{1}, & P_{\mathbf{n}} = L_{\mathbf{n}}, \\ \mathbf{n} - \mathbf{m}, & P_{\mathbf{n}} = V_{\mathbf{n}}^{\mathbf{m}}. \end{cases}$$

The following theorem can be proved using the same arguments as for the univariate case.

THEOREM 4.6. *Under the previous setting, for all input and output sizes,* $\mathbf{n} = (n_1, n_2, n_3)$ *and* $\mathbf{N} = (N_1, N_2, N_3)$*, respectively, we have*

$$\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^\infty} \leq \mathcal{E}_{\mathbf{r}}(f)_{\ell^\infty(X_{\mathbf{N}})} + \overline{\Lambda}_{n_1} \overline{\Lambda}_{n_2} \overline{\Lambda}_{n_3} \left[\mathcal{E}_{\mathbf{r}}(f)_{\ell^\infty(X_{\mathbf{n}})} + \|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^\infty} \right],$$

and

$$\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^2} \leq \mathcal{E}_{\mathbf{r}}(f)_{\ell^2(X_{\mathbf{N}})} + \mathcal{C} \left[\mathcal{E}_{\mathbf{r}}(f)_{\ell^2(X_{\mathbf{n}})} + \|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^2} \right]$$

where $\mathcal{C} > 0$ *is a constant independent of* $\mathbf{n}, \mathbf{N}$ *and* $f, \widetilde{f}$.

Moreover, if $\mathbf{n}$ *and* $\mathbf{N}$ *are such that*

$$n_i = s_i N_i, \quad \textit{with} \quad s_i \in \mathbb{N} \quad \textit{odd} \qquad i = 1:3,$$

then we get

$$\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^\infty} \leq \|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^\infty}$$

and

$$\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^2} \leq \sqrt{s_1 s_2 s_3}\, \|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^2}.$$

Finally, we remark that in the 3D case the error can be also estimated by using the results of the 1D case applied to the following univariate functions

$$f_{y,z}^{(1)}(x) = f(x, y, z), \qquad f_{x,z}^{(2)}(y) = f(x, y, z), \qquad f_{x,y}^{(3)}(z) = f(x, y, z).$$

More precisely, we observe that the trivariate VP interpolation error can be, for instance, decomposed as follows

$$\begin{aligned} f(x, y, z) - V_{\mathbf{n}}^{\mathbf{m}} f(x, y, z) &= f_{y,z}^{(1)}(x) - V_{n_1}^{m_1} f_{y,z}^{(1)}(x) \\ &+ \sum_{i=1}^{n_1} \left[f_{\xi_i^{n_1}, z}^{(2)}(y) - V_{n_2}^{m_2} f_{\xi_i^{n_1}, z}^{(2)}(y) \right] \Phi_{n_1, i}^{m_1}(x) \\ &+ \sum_{j=1}^{n_2} \sum_{i=1}^{n_1} \left[f_{\xi_i^{n_1}, \xi_j^{n_2}}^{(3)}(z) - V_{n_3}^{m_3} f_{\xi_i^{n_1}, \xi_j^{n_2}}^{(3)}(z) \right] \Phi_{n_1, i}^{m_1}(x) \Phi_{n_2, j}^{m_2}(y), \end{aligned}$$

and a similar decomposition holds for the trivariate Lagrange interpolation error too.

Consequently, setting $P_{n_i} = L_{n_i}$ for LCI3 and $P_{n_i} = V_{n_i}^{m_i}$ for VPI3, $i = 1 : 3$, we easily get

$$\begin{aligned}\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^\infty} \leq & \overline{\Lambda}_{n_1}\overline{\Lambda}_{n_2}\overline{\Lambda}_{n_3}\|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^\infty} + \max_{y\in X_{N_2},\, z\in X_{N_3}} \|f^{(1)}_{y,z} - P_{n_1} f^{(1)}_{y,z}\|_{\ell^\infty(X_{n_1})} \\ & + \overline{\Lambda}_{n_1} \max_{x\in X_{n_1},\, z\in X_{N_3}} \|f^{(2)}_{x,z} - P_{n_2} f^{(2)}_{x,z}\|_{\ell^\infty(X_{n_2})} \\ & + \overline{\Lambda}_{n_1}\overline{\Lambda}_{n_2} \max_{x\in X_{n_1},\, y\in X_{n_3}} \|f^{(3)}_{x,y} - P_{n_3} f^{(3)}_{x,y}\|_{\ell^\infty(X_{n_3})}\end{aligned}$$

and

$$\begin{aligned}\|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^2} \leq & \mathcal{C}\|I_{\mathbf{n}} - \widetilde{I}_{\mathbf{n}}\|_{\ell^2} + \max_{y\in X_{N_2},\, z\in X_{N_3}} \|f^{(1)}_{y,z} - P_{n_1} f^{(1)}_{y,z}\|_{\ell^2(X_{n_1})} \\ & + \mathcal{C} \max_{x\in X_{n_1},\, z\in X_{N_3}} \|f^{(2)}_{x,z} - P_{n_2} f^{(2)}_{x,z}\|_{\ell^2(X_{n_2})} \\ & + \mathcal{C} \max_{x\in X_{n_1},\, y\in X_{n_2}} \|f^{(3)}_{x,y} - P_{n_3} f^{(3)}_{x,y}\|_{\ell^2(X_{n_3})}\end{aligned}$$

with $\mathcal{C} > 0$ independent of $\mathbf{n}$, N and $f, \widetilde{f}$.

5. Quality measures and tests.

In this section, we describe some experiments to show the performance of LCI3 compared with VPI3. We also comparewith the multivariate cubic interpolation method (shortly BIC) and the Lanczos interpolation method, implemented by the Matlab built-in function IMRESIZE3, with the option 'cubic' and 'lanczos3', respectively.

For the quantitative analysis of the performance, we use the standard quality measures PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index Measure) computed by Matlab (see, e.g., [18, 28]). Denoted by $\mathcal{I}_{\mathbf{N}}$ the target image and $\widetilde{\mathcal{I}}_N$ the output resized image, we recall that PSNR is defined as

$$\mathrm{PSNR}(\mathcal{I}_{\mathbf{N}}, \widetilde{\mathcal{I}}_{\mathbf{N}}) = 20\log_{10}\frac{\mathrm{L}_{\max}}{\sqrt{\mathrm{MSE}(\mathcal{I}_{\mathbf{N}}, \widetilde{\mathcal{I}}_{\mathbf{N}})}},$$

where

$$MSE(\mathcal{I}_{\mathbf{N}}, \widetilde{\mathcal{I}}_{\mathbf{N}}) = \|I_{\mathbf{N}} - \widetilde{I}_{\mathbf{N}}\|_{\ell^2} = \frac{1}{N_1N_2N_3}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3}\left[\mathcal{I}_{\mathbf{N}}(i,j,k) - \widetilde{\mathcal{I}}_{\mathbf{N}}(i,j,k)\right]^2,$$

and L_{max} is the maximum possible value of the image, that in our experiments can be

$$L_{max} = \begin{cases} 255, & 8-bits \\ 65535, & 16-bits \end{cases} \text{images representation.}$$

PSNR values vary in $(0, +\infty)$, being infinity the best possible value (corresponding to a null MSE). Moreover, SSIM is defined as

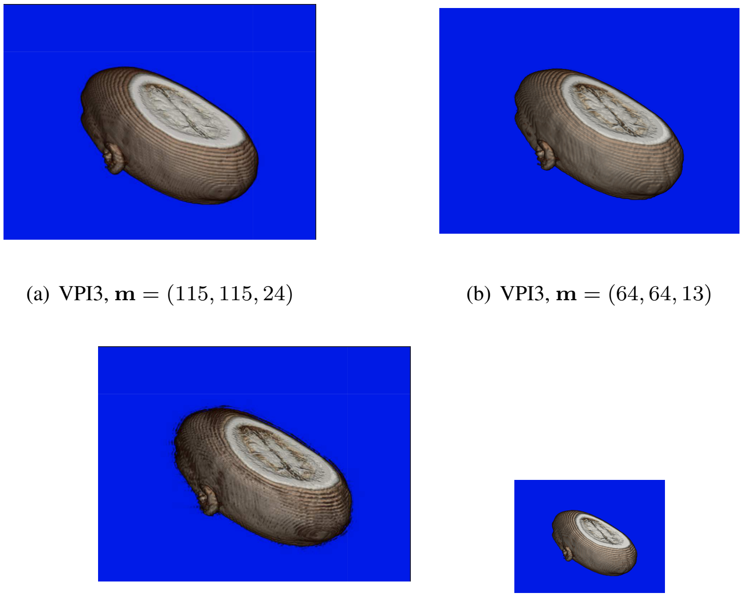
$$\mathrm{SSIM}(\mathcal{I}_{\mathbf{N}}, \widetilde{\mathcal{I}}_{\mathbf{N}}) = \frac{\left[2\mu(\mathcal{I}_{\mathbf{N}})\mu(\widetilde{\mathcal{I}}_{\mathbf{N}}) + \mathrm{c}_1\right]\left[2\mathrm{cov}(\mathcal{I}_{\mathbf{N}},\, \widetilde{\mathcal{I}}_{\mathbf{N}}) + \mathrm{c}_2\right]}{\left[\mu^2(\mathcal{I}_{\mathbf{N}}) + \mu^2(\widetilde{\mathcal{I}}_{\mathbf{N}}) + \mathrm{c}_1\right]\left[\sigma^2(\mathcal{I}_{\mathbf{N}}) + \sigma^2(\widetilde{\mathcal{I}}_{\mathbf{N}}) + \mathrm{c}_2\right]},$$

where $\mu(A), \sigma(A)$ and cov (A, B) indicate the average, variance and covariance, respectively, of the arrays A, B, and c_1, c_2 are constants usually fixed as $c_1 = (0.01 \times L_{max})$, $c_2 = (0.03 \times L_{max})$. SSIM values vary in $[0, 1]$, being $SSIM = 1$ the best result.

In Tests 2,3, and 4 where the reference image (or Ground truth) is available we have computed PSNR and SSIM values. We outline that we have implemented VPI3 method with the additional parameter $\mathbf{m} := \lfloor \mathbf{n}\theta \rfloor$, with $\theta \in \zeta := \{0.1, 0.2, \dots, 0.9)$, and selected the best PSNR and SSIM values attained on ζ.

Our numerical tests concern the following three types of images:

- an MRI of a human cranium available in the MatLab library (uint8 format) and consisting of 27 Slices of dimensions 128×128, shortly denoted **'BRAIN'**;
- a DICOM (Digital Imaging and Communications in Medicine) format image [29], which is a study in the abdomen about the arterielle system acquired in modality CT (361 Slices of dimensions 512×512, uint16 format), and shortly denoted **'DICOM-Study1'**;
- the synthetic 3D Shepp-Logan phantom image, widely used in the literature for testing image reconstruction [22]. It is generated by the Matlab function PHANTOM3D and shortly denoted **'PHANTOM'**.



(a) VPI3, $\mathbf{m} = (115, 115, 24)$

(b) VPI3, $\mathbf{m} = (64, 64, 13)$

(c) LPI3

(d) Initial image

FIG. 5.1. *Test 1:BRAIN upscaling* $x2$ *by VPI3 (two choices of* $\mathbf{m}$*) and by LCI3.*

5.1. Test 1. This test aims to highlight the improvement achieved by filtered VP interpolation vs Lagrange interpolation, with the aid of the additional parameter $\mathbf{m}$. The image BRAIN has been taken as an input image to be upscaled, with a factor 2, by LCI3 and by VPI3 implemented with two different choices of $\mathbf{m}$. The volume reconstructions in Figure 5 display the respective output images (54 slices of dimensions 256×256) and the original cranium data. We can note some artifacts produced along the contour of the image resized by LCI3, while in both the images realized by VPI3 such effects are not visible. In particular, image (b) upsampled by VPI3 with $\mathbf{m} = \lfloor \mathbf{n}/2 \rfloor$ appears a little bit fair.

5.2. Test 2. In this second test, the image BRAIN has been taken as the target image, namely as $\mathcal{I}_{\mathbf{N}}$ where the size $\mathbf{N}$ is taken according to our previous notation. We want to examine how the performance results (either in upscaling or in downscaling) depend on different methods used to get the input image from the target one. To be more precise, we first generate the input image with size $\mathbf{n}$, say it $\widetilde{\mathcal{I}}_{\mathbf{n}}$, by applying one of the comparison methods LCI3, VPI3, BIC, LANCZOS, and also the following methods

1. trilinear interpolation ('linear'),
2. nearest-neighbor interpolation ('nearest'),

implemented in Matlab by IMRESIZE3 with the option 'linear' and 'nearest', respectively.

Then, for each case, we apply BIC, LANCZOS, LCI3, and VPI3 (in upscaling and downscaling, for the scale factors 2,3,4) to resize $\widetilde{\mathcal{I}}_{\mathbf{n}}$ at the resolution $\mathbf{N}$, and, being available the target image, we compute the respective PSNR and SSIM values. The results are given in Table 5.1, 5.2, 5.3, 5.4, 5.5, and 5.6, each one corresponding to the specific method used to generate the input image. The best results, also in cases of ex-aequo, are evidenced in bold.

TABLE 5.1
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by BIC.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	30.0587	0.8969	27.6775	0.8093	26.3989	0.7414
LANCZOS	30.4403	0.9057	27.9418	**0.8197**	26.5929	**0.7507**
LCI3	30.5884	0.8992	28.0777	0.8103	26.7148	0.7350
VPI3	**30.6006**	**0.9058**	**28.0816**	0.8194	**26.7168**	0.7486
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	38.3195	0.9862	38.6428	0.9873	38.5650	0.9871
LANCZOS	39.9393	0.9905	40.1922	0.9911	40.1292	0.9910
LCI3	53.5674	0.9993	**Inf**	**1**	56.1893	**0.9996**
VPI3	**53.9381**	**0.9994**	**Inf**	**1**	**58.2314**	**0.9996**

As a global evaluation of Tables 5.1, 5.2, 5.3, 5.4, 5.5, and 5.6, VPI3 provides a small improvement in upscaling w.r.t the other interpolation methods, while the gap is significantly high in downscaling. In particular, note that in the case of downsampling with scale factor 3, according to what is said in Section 3.1, VPI3 and LCI3 coincide and provide the best quality measure, i.e., PSNR = Inf and SSIM = 1. However, we caution the reader that this excellent result depends on the method used to generate the input image and is unlikely to be obtained in real cases. For further details, we refer the reader to the bivariate case studied in [12, 13], as similar observations can be replicated. Finally, concerning the other downscaling cases, as well as in all upscaling with any scale factor, we observe that LCI3 appears less stable than VPI3 which, even changing the input image generation method, always detects the highest quality measure, except one case; see Table 5.2.

5.3. Test3. Upscaling along only one direction. In this test we consider a task often required in Medical Imaging, namely the image upscaling along only in one of the three directions (see Section 3.2), in the particular case of the scale factor 2. To this aim, we take the DICOM-Study1 as target image $\mathcal{I}_{\mathbf{N}}$, with size $\mathbf{N} = (N_1, N_2, N_3)$, and we construct the

TABLE 5.2
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by imresize3 with option 'linear'.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	29.2248	0.8719	27.2332	0.7883	26.0561	0.7221
LANCZOS	29.5914	0.8826	27.4862	0.7997	26.2362	0.7323
LCI3	29.7131	0.8825	27.5951	0.7970	**26.3339**	0.7262
VPI3	**29.7141**	**0.8850**	**27.5963**	**0.8011**	26.3333	**0.7335**
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	35.0536	0.9696	36.2980	0.9778	35.5792	0.9735
LANCZOS	35.9314	0.9755	37.2817	0.9824	36.4741	0.9786
LCI3	39.3188	**0.9893**	**Inf**	**1**	44.4675	0.9967
VPI3	**39.3200**	**0.9893**	**Inf**	**1**	**44.4794**	**0.9968**

TABLE 5.3
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by imresize3 with option 'nearest'.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	27.9837	**0.8661**	26.5746	**0.8019**	25.0829	**0.7406**
LANCZOS	27.7093	0.8611	26.2405	0.7909	24.7449	0.7252
LCI3	27.2964	0.7971	25.7339	0.6893	24.3418	0.5906
VPI3	**27.9854**	0.8640	**26.6819**	0.7974	**25.2910**	0.7285
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	41.0014	0.9929	40.1348	0.9913	39.8174	0.9905
LANCZOS	42.8631	0.9955	41.7072	0.9940	41.3504	0.9934
LCI3	31.0828	0.9613	**Inf**	**1**	37.6482	0.9783
VPI3	**54.3092**	**0.9997**	**Inf**	**1**	**64.7407**	**1**

input image $\mathcal{I}_{\mathbf{n}}$, with size $\mathbf{n} = (n_1, n_2, n_3)$ such that two of the three components of $\mathbf{n}$ and $\mathbf{N}$ remain unchanged while the third ones, say N_i, n_i, are such that $n_i = \left\lfloor \frac{N_i+1}{2} \right\rfloor$.

To be more precise, $\mathcal{I}_{\mathbf{n}}$ is obtained from $\mathcal{I}_{\mathbf{N}}$ by eliminating all the voxel-values that have an even index along the i-th dimension we want to change. For instance, in upscaling along the third dimension, Figure 5.2 illustrates, in the case of 3 slices ($N_i = 3$), the idea of the reconstruction by interpolation of the teal slice starting from the known values of the purple and pink slices. In this case, the axis z represents the direction along which we need to interpolate.

More generally, in Matlab code, the input image is given by

$$\mathcal{I}_{\mathbf{n}} := \mathcal{I}_{\mathbf{N}}(1 : N_1, 1 : N_2, 1 : 2 : N_3)$$

if we are upscaling along the third dimension, and similarly we proceed along the second and third dimensions.

To upscale such an input image $\mathcal{I}_{\mathbf{n}}$ at the original size $\mathbf{N}$, we apply BIC, LANCZOS, LCI3, VPI3, and, having available the target images, we compute the quality measures not only

TABLE 5.4
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by LCI3.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	29.6151	0.8736	26.7880	**0.8079**	25.4077	**0.7287**
LANCZOS	29.3398	0.8664	26.4753	0.7965	25.0565	0.7101
LCI3	28.6644	0.8004	25.9753	0.6971	24.6387	0.5871
VPI3	**29.6244**	**0.8747**	**26.8521**	0.8020	**25.5946**	0.7158
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	40.3143	0.9704	40.7449	0.9875	40.6215	0.9834
LANCZOS	43.0964	0.9775	43.4821	0.9906	43.3594	0.9875
LCI3	50.9347	**0.9918**	**Inf**	**1**	56.9852	0.9988
VPI3	**50.9605**	0.9910	**Inf**	**1**	**58.2415**	**0.9991**

TABLE 5.5
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by VPI3 with $\mathbf{m} = (90,\ 90,\ 19)$.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	30.3954	0.9079	27.0742	**0.8178**	25.9585	**0.7578**
LANCZOS	30.5376	0.9103	26.8316	0.8099	25.7134	0.7461
LCI3	30.3439	0.8803	26.4077	0.7187	25.3867	0.6334
VPI3	**30.5354**	**0.9088**	**27.0826**	0.8147	**26.0207**	0.7542
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC3	37.5644	0.9833	37.7054	0.9842	37.6764	0.9840
LANCZOS	38.8275	0.9876	38.8799	0.9880	38.8651	0.9879
LCI3	57.4818	**0.9997**	**Inf**	**1**	60.3868	0.9998
VPI	**57.8748**	**0.9997**	**Inf**	**1**	**62.8055**	**0.9999**

TABLE 5.6
Test 2: PSNR and SSIM values attained in upscaling (x) and downscaling (:) on MRI image, with input image generated by imresize3 with option 'lanczos3'.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	30.3148	0.9035	27.8514	0.8154	26.5175	0.7441
LANCZOS	30.6713	**0.9104**	28.1089	**0.8235**	26.7151	0.7523
LCI3	**30.8592**	0.9026	**28.2809**	0.8127	**26.8744**	0.7351
VPI3	30.8331	**0.9104**	28.2794	0.8228	26.8725	**0.7530**
	:2		:3		:4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC3	39.7173	0.9901	39.9892	0.9908	39.9044	0.9906
LANCZOS	41.7136	0.9935	41.8582	0.9938	41.8012	0.9937
LCI3	52.1988	0.9992	Inf	1.0000	55.8549	0.9996
VPI3	**52.7651**	**0.9992**	**Inf**	**1.0000**	**57.6055**	**0.9998**

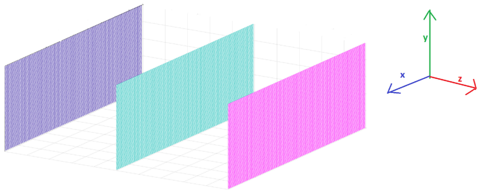


FIG. 5.2. *Test 3: The idea of the interpolation between slices: the teal slice unknown values are obtained by interpolating purple and pink slices. The axis z represents the direction along which we interpolate.*

TABLE 5.7
Test 3: Global 3D PSNR and SSIM values (top of the table), and the mean of 2D PSNR and SSIM values referred to the neglected slices (bottom of the table) in upscaling ×2.

3D quality measures

	1-st		2-nd		3-rd	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	**68.2220**	**0.9996**	**71.0322**	**0.9998**	78.4908	**1**
LANCZOS	68.2047	0.9996	71.0087	0.9998	78.5010	1
LCI3	68.0357	**0.9996**	70.8964	**0.9998**	78.2296	**1**
VPI3	68.2205	**0.9996**	71.0259	**0.9998**	**78.5067**	**1**

Mean 2D quality measures

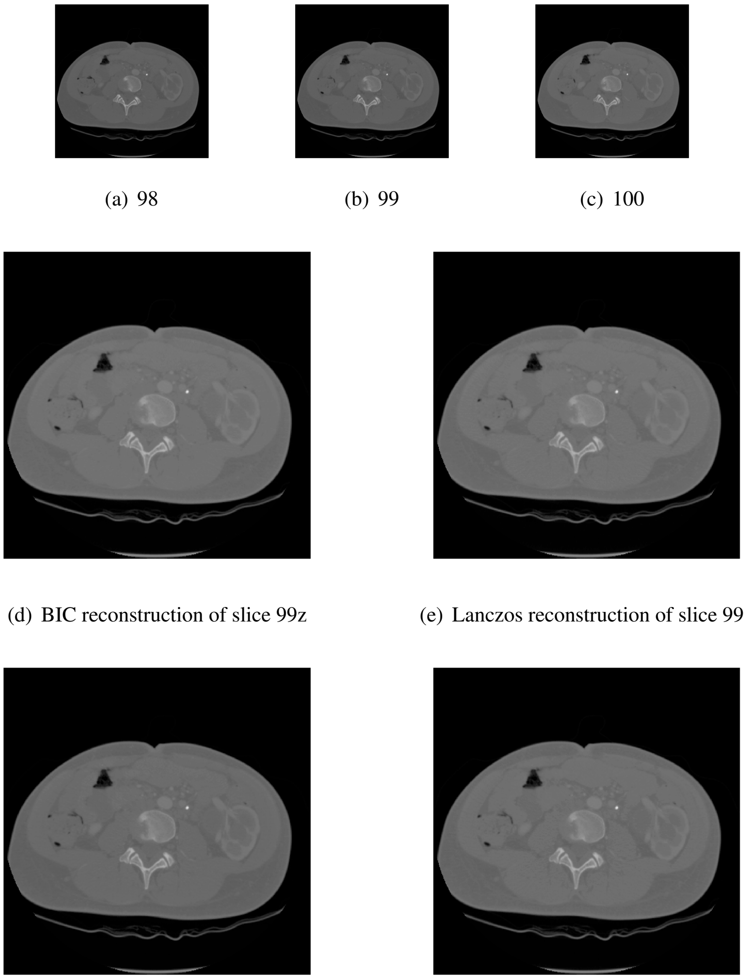
	1-st		2-nd		3-rd	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	73.5158	0.9993	**76.6248**	**0.9998**	77.5766	1
LANCZOS	73.8288	0.9994	76.5749	**0.9998**	77.8335	1
LCI3	73.3102	**0.9994**	76.3945	**0.9998**	77.5402	1
VPI3	**73.8356**	**0.9994**	76.5992	**0.9998**	**77.8708**	1

for the 3D images but also for the 2D slices representing the reconstruction of the neglected ones. More precisely, we compute the usual PSNR and SSIM between the output 3D images and the target one represented by DICOM-Study1 (named "3D quality measures"). Moreover, we also compute the averages of the bivariate PSNR and SSIM values concerning all the neglected 2D images along the direction to resize (named "mean 2D quality measures"). The results of both 3D and 2D measurements are given in Table 5.7, where the 1st, 2nd, and 3rd columns correspond to the choice of the size n_i to upscale, with $i = 1, 2$, and 3, respectively.

Comparing the performance results in Table 5.7, we observe that all the values are almost comparable among them, even though the mean of the PSNR values of the 2D images, shows a better performance of VPI3 with respect to the other methods, confirming in some sense, the better local approximation provided by VPI3 interpolation.

Figure 5.3 illustrates the pointwise qualitative results of the previous experiment focusing on the reconstruction of slice $99z$ displayed in the central position of the first row upscaled along the axial direction (parallel to the plane $z = 0$, "from top to down"), in our notation $N_3 = 2n_3 + 1,\ N_1 = n_1,\ N_2 = n_2$.

By the quantitative results displayed in Table 5.8, which reports the pointwise 2D PSNR and SSIM values taking as target 2D image $250x$, $162y$, and $99z$, respectively, we do not



(a) 98

(b) 99

(c) 100

(d) BIC reconstruction of slice 99z

(e) Lanczos reconstruction of slice 99z

(f) LCI3 reconstruction of slice 99z

(g) VPI3 reconstruction of slice 99z

FIG. 5.3. *Test 3: Original slices along* z *direction* $98z, 99z, 100z$ *(first row), Reconstructed slice* $99z$ *by BIC, LANCZOS, LCI3, VPI3, (second row).*

detect differences among the methods, being all the quality measures very close among them. However, at a visual level, the reconstructed slice $99z$ by VPI3 presents a little bit of major contrasts with respect to the reconstructions realized by BIC, better evidencing the contours of some tissues, appearing more flat in the BIC images.

5.4. Test 4. The synthetic image Shepp-Logan Phantom. This test is performed on the synthetic image PHANTOM. Let us assume that the target image $\mathcal{I}_{\mathbf{N}}$, with $\mathbf{N} = (\mu, \mu, \mu)$ is generated by the Matlab function PHANTOM3D with size parameter μ.

About the input image, let us first consider the case when it is also generated by PHANTOM3D. Table 5.9 displays the PSNR values achieved by BIC, LANCZOS, LCI3, and VPI3 in both upscaling and downscaling with scale factor 2. According to the previous experiments, VPI3 almost always provides slightly better quality measures in both up and down resizing.

TABLE 5.8
Test 3: 2D PSNR and SSIM values for slice $250x$ (left), slice $162y$ (middle), slice $99z$ (right).

	250x		156y		99z	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BIC	**65.1103**	**0.9999**	76.4455	**0.9999**	81.7798	**1**
LANCZOS	64.9646	**0.9999**	76.5420	**0.9999**	81.8273	**1**
LCI3	64.6527	**0.9999**	76.4781	**0.9999**	80.8667	**1**
VPI3	65.0356	**0.9999**	**76.6517**	**0.9999**	**81.8585**	**1**

However, for all downscaling cases, we note that the large gap displayed for the MRI image (cf. Tables 5.1-5.6) is now much more attenuated.

TABLE 5.9
Test 4: PSNR for PHANTOM (used for both input/output) in upscaling and downscaling with scale factor 2.

	PSNR in upscaling x2				PSNR in downscaling :2			
μ	BIC	Lanczos	LCI3	VPI3	BIC	Lanczos	LCI3	VPI3
50	62.9890	63.0275	63.1405	**63.1405**	66.5665	66.4048	65.9866	**66.8365**
51	63.1743	63.0526	63.0863	**63.6257**	66.7644	66.6604	66.4442	**67.1229**
90	64.8480	64.9271	65.0157	**64.9591**	69.1358	69.0202	68.8398	**69.2855**
91	65.4179	65.3064	65.1224	**65.4420**	69.1220	69.0459	68.9613	**69.3494**
130	66.5688	66.6704	**66.6959**	66.6885	70.7026	70.6444	70.5571	**71.1462**
131	67.1639	67.1205	66.9828	**67.1834**	70.8402	70.7747	70.6225	**71.1289**
170	67.9832	67.9921	**68.0832**	68.0255	71.9770	72.0270	71.6893	**72.5645**
171	68.5279	68.5127	68.3925	**68.5344**	72.1082	72.0526	71.6990	**72.6223**
200	68.7833	68.7594	68.7810	**68.7929**	72.6611	72.6737	72.3772	**73.2203**
201	69.2559	69.2147	69.0722	**69.3393**	72.6382	72.6735	72.3432	**73.2663**

Now, let us focus on upscaling x2 and consider the case that the input image is not generated by PHANTOM3D but it is obtained by decimating the target image $I_{\mathbf{N}}$, with step 2, simultaneously along all the three directions. Briefly, in Matlab code, we take the following input image

$$\mathcal{I}_{\mathbf{n}} := \mathcal{I}_{\mathbf{N}}(1:2:\mu, 1:2:\mu, 1:2:\mu). \tag{5.1}$$

By BIC, LCI3, and VPI3 we resize $\mathcal{I}_{\mathbf{n}}$ at the size $\mathbf{N}$, and in Table 5.10 we show the respective PSNR and SSIM values for increasing values of μ. In this case, we observe that the best performance results are achieved by VPI3, followed in order by LCI3, Lanczos and BIC. Moreover, we note that the gap between the PSNR values of Lanczos and BIC with respect to VPI3 is larger for odd μ. To evidence such behavior, in Figure 5.4 we have plotted the PSNR values for increasing odd values of μ varying from 51 to 201. Finally, comparing Table 5.10 and Table 5.9 we see that VPI3 maintains a stable performance even when changing the input image, which is generated by decimation in the former table and by PHANTOM3D in the latter. Moreover, VPI3 attains a superior performance in Table 5.10 with a greater gap over BIC and Lanczos methods.

TABLE 5.10
Test 4: PSNR and SSIM for PHANTOM with input image given by (5.1).

	PSNR				SSIM			
μ	BIC	Lanczos	LCI3	VPI3	BIC	Lanczos	LCI3	VPI3
50	62.6222	62.6222	62.5842	**62.6634**	**0.9993**	0.9993	0.9992	**0.9993**
51	62.3104	62.2457	62.9820	**63.4528**	0.9992	0.9993	0.9992	**0.9994**
90	64.4500	64.4500	64.6126	**64.6651**	**0.9995**	0.9995	**0.9995**	**0.9995**
91	63.3737	63.4163	64.8595	**65.0094**	0.9993	0.9995	0.9993	**0.9996**
130	65.8861	65.8861	66.2534	**66.2736**	**0.9996**	0.9996	**0.9996**	**0.9996**
131	64.7870	64.8017	66.8349	**67.0706**	0.9995	0.9995	**0.9997**	**0.9997**
170	67.0740	67.0740	67.5461	**67.5504**	**0.9997**	0.9997	**0.9997**	**0.9997**
171	65.9378	65.9378	68.2842	**68.4395**	0.9995	0.9995	**0.9998**	**0.9998**
200	67.7811	67.7811	68.2382	**68.2520**	**0.9997**	0.9997	**0.9997**	**0.9997**
201	66.6180	66.6180	69.0730	**69.1504**	0.9996	0.9996	**0.9998**	**0.9998**

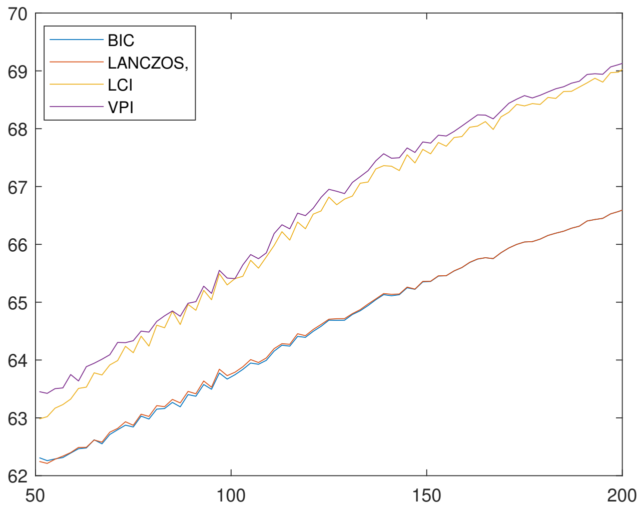


FIG. 5.4. *Test 4: PSNR values attained for odd values of* $\mu = 51 : 2 : 201$.

Code and supplementary materials The source code implementing LCI3 and VPI3 is openly available at the following link: https://github.com/ImgScaling/3D_VPIscaling.

Acknowledgements The work has been accomplished within RITA (Research ITalian network on Approximation) and UMI (Unione Matematica Italiana) research groups: TAA-UMI (Approximation Theory and Applications), AI&ML&MAT-UMI (Mathematics for Artificial Intelligence and Machine Learning). It has been partially supported by GNCS-INdAM. The authors thank the anonymous reviewers for their helpful remarks that allowed to improve the quality of the paper.

REFERENCES

[1] R. Archibald, and A. Gelb, *A method to reduce the Gibbs ringing artifact in MRI while keeping tissue boundary integrity*, IEEE Trans. Med. Imaging, 21 (2002), pp. 305–319.

[2] H. Amri, M. K. Abdmouleh, A. Khalfallah, J. C. Lapayre, and M. S. Bouhlel, *Transmission and archiving of reduced MRI medical images*, Int. J. Med. Eng. Inf., 12 (2020), pp. 19–51.

[3] M. R. Capobianco and W. Themistoclakis, *Interpolating polynomial wavelets on* $[-1, 1]$, Adv. Comput. Math., 23 (2005), pp. 353–374.

[4] S. Gao, C. Zhang, Y. Zhang, and Y. Zhou, *Medical image zooming algorithm based on bivariate rational interpolation*, in International Symposium on Visual Computing, Lecture Notes in Computer Science, 5359 LNCS, Springer, Berlin, 2008, pp. 672–681.

[5] D. Gottlieb, B. Gustafsson, and B. Forssen, *On the direct Fourier method for computed tomography*, IEEE Trans. Med. Imaging, 19 (2000), pp. 223–233.

[6] Y. Gulzar, Y. Hamid, A. B. Soomro, A. A. Alwan, and L. Journaux, *A convolution neural network-nased seed classification system*, Symmetry, 12 (2020), 2018 (18 pages).

[7] R. Keys, *Cubic convolution interpolation for digital image processing*, IEEE Trans. Acoust. Speech Signal Process., 29 (1981), pp. 1153–1160.

[8] T. M. Lehmann, C. Gönner, and K. Spitzer, *Survey: interpolation methods in medical image processing*, IEEE Trans. Med. Imaging, 18 (1999), pp. 1049–1075.

[9] C. Lanczos, *Applied Analysis*, Dover Publications, New York, 1988.

[10] L. Juelin, X. Guoliang, and Z. Yongjie, *Medical image interpolation based on multi-resolution registration*, Comput. Math. Appl., 66 (2013), pp. 1–18.

[11] T. Moraes, P. Amorim, J.V. Da Silva, and H. Pedrini, *Medical image interpolation based on 3D Lanczos filtering*, Comput. Methods Biomech. Biomed. Engin. Imaging Vis., 8 (2020), pp. 294-300.

[12] D. Occorsio, G. Ramella, and W. Themistoclakis, *Lagrange-Chebyshev Interpolation for image resizing*, Math. Comput. Simulation, 197 (2022), pp. 105–126.

[13] ———, *Image scaling by de la Vallée-Poussin filtered interpolation*, J. Math. Imaging Vision, 65 (2023), pp. 513–541.

[14] D. Occorsio and W. Themistoclakis, *On the filtered polynomial interpolation at Chebyshev nodes*, Appl. Numer. Math., 166 (2021), pp. 272–287.

[15] ———, *Some remarks on filtered polynomial interpolation at Chebyshev nodes*, Dolomites Res. Notes Approx., 14 (2021), pp. 68–84.

[16] G. Plonka, D. Potts, G. Steidl, and M. Tasche, *Numerical Fourier Analysis, Ser. Applied and Numerical Harmonic Analysis*, Birkhäuser Springer Nature Switzerland AG, 2018.

[17] D. Poggiali, D. Cecchin, C. Campi, and S. De Marchi, *Oversampling errors in multimodal medical imaging are due to the Gibbs effect*, Mathematics, 9 (2021), 1348 (18 pages).

[18] G. Ramella, *Evaluation of quality measures for color quantization*, Multimed. Tools. Appl. 80 (2021), pp. 32975–33009.

[19] G. Ramesh and T. A. Prasath, *An aphoristic study on different interpolation techniques for medical image scaling and its comparative analysis*, in International Conference on Computer Communication and Informatics, ICCCI, 2021 Article number 9402675.

[20] N. Rathee, S. Pahal, and D. Sheoran, *Evaluating the uncertainty of classification due to image resizing techniques for satellite image classification*, Mapan-J. Metrol. Soc. India., 36 (2021), pp. 243–251.

[21] T. Rivlin, *An Introduction to the Approximation of Functions*, Dover, New York, 1981.

[22] M. Schabel, *3D Shepp-Logan phantom*, MATLAB Central File Exchange, 2022.
https://www.mathworks.com/matlabcentral/fileexchange/9416-3d-shepp-logan-phantom

[23] W. Themistoclakis, *Uniform approximation on* $[-1, 1]$ *via discrete de la Vallée Poussin means*, Numer. Algorithms, 60 (2012), pp. 593–612.

[24] W. Themistoclakis and M. Van Barel, *Generalized de la Vallée Poussin approximations on* $[-1, 1]$, Numer. Algorithms, 75 (2017), pp. 1–31.

[25] L. N. Trefethen, *Approximation Theory and Approximation Practice: Extended Edition*, Society for Industrial and Applied Mathematics, Philadelphia, 2019.

[26] Y. A. Üncü, T. Mercan, G. Sevym, and M. Canpolat, *Interpolation applications in diffuse optical tomography system*, in 2017 21st National Biomedical Engineering Meeting (BIYOMUT), IEEE, 2018.

[27] J. K. Udupa and G. T. Herman, *3D Imaging in Medicine*, 2nd ed., CRC Press, Boca Raton, 1999.

[28] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, *Image quality assessment: from error visibility to structural similarity.* IEEE Trans. Image Process., 13 (2004), pp. 600–612.

[29] *DICOM Library*, https://www.dicomlibrary.com.